

キャプテン画像入力装置における ユーザインタフェース

近藤 隆志 森 和宏
(株)東芝 情報通信システム技術研究所

我々が開発したキャプテン画像入力装置におけるユーザインタフェースについて報告する。この装置は、ビデオ・カメラで撮影した原稿に会話編集をしてキャプテン仕様の画像を作成するものであるが、メニューと画像のピックだけで全ての操作ができるようにしている。まず、アプリケーションから独立したメニュー管理部の機能を紹介したあとで、その機能範囲についての考察を述べる。次に、画像編集におけるキャンセル処理の外部仕様と内部での実現方法について、この装置での処理を例として説明しながら考察を述べる。

USER INTERFACE IN A CAPTAIN IMAGE INPUT TERMINAL

Takashi KONDO and Kazuhiro MORI

Information & Communication Systems Laboratory,
TOSHIBA Corporation
70, Yanagicho, Saiwai-ku, Kawasaki, 210 Japan

We have developed a CAPTAIN image input terminal. It allows users to capture original colour images via video-camera, edit them, convert them automatically, and consequently to create CAPTAIN images in a short time. Users can use this terminal only by menu-picking & image-pointing. Menu management subsystem of this terminal is introduced, and its desirable specification is discussed. Operation cancelling, one of the most important facilities in user interface, is discussed by studying the terminal's case.

1. はじめに

我々はキャプテン画面の作成を容易にするために、ビデオカメラで撮影した画像を自動変換・会話編集してキャプテン画面を作成する装置を開発した^{1) 2) 3)}。この装置では画面の下側にメニューを表示して、カーソルとタブレットによる画面ピックだけですべての操作ができるようにしている。

ここでは装置の概要を述べたあとで、ユーザインタフェースについて、メニュー管理部の機能とキャンセル処理に分けて報告する。

2. 装置の概要

図1にこの装置の全体構成を示す。この装置は大きく2つの部分に分けられる。1つは画像入力部で、ここではこの部分のユーザ・インタフェースについて報告する。もう1つは文字入力部で、画像入力部から送られてきた画像に文字列を重ねてからセンターに登録する。表1に画像入力部の機能を示し、以下その概要を述べる。

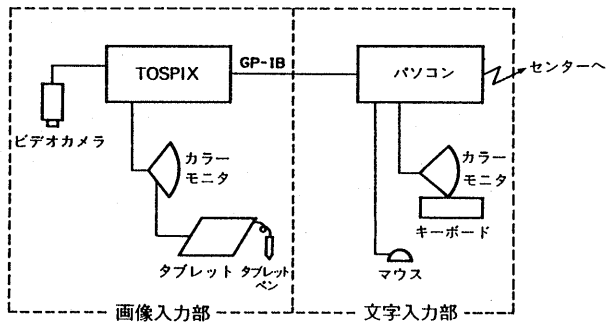


図1. 本装置の構成図

(1) 入力

ビデオ・カメラで原稿を撮影してキャプテン仕様の15色画像に変換する。原稿の種類に応じてカラー/モノクロ/線画の3モードがある。

(2) 編集

会話的な画像編集で、太・中・細の3種類のペン先での描画、色付け・色変更、切貼りの3モードがある。

(3) その他

キャプテン特有のブロック着色、文字入力部との通信、フロッピーの画像ファイルの管理などがある。

画像編集装置としてのこの装置の特徴は、画像を記述するデータを一切持っていないことで、画素ごとに色コードを有する画像の形のデータを常に処理対象にしている。この条件の下で、アプリケーション・プログラムから独立した画像ズームやキャンセルなどの機能を応答性良く（大部分の処理が5秒以内）実現するために、画像処理専用ハードウェアを活用している。

表1. 画像入力部の機能

画像入力部の機能		
入 力	編 集	そ の 他
カラー/モノクロ/線画の自動入力、雑画除去	描画、色付け/色変更、切貼り/拡大・縮小	ブロック単位の処理 ²⁾ 、FDD管理

3. メニュー管理部の機能

この装置では、メニュー管理部は1つの共通タスクとして独立していて、アプリケーションプログラムに起動される。ただし、描画用タスクだけは自由曲線の連続描画を高速に行うために自タスク内にメニュー管理プログラムを内蔵している。ここでは共通タスクであるメニュー管理部の機能を説明したあとで、開発経験に基づく考察を述べる。

3.1 機能の概要

図2に、メニュー管理部とアプリケーション・プログラム、ユーザーとの関係を示す。メニュー管理部は以下のように表示画面作成とユーザからの入力に対する処理を行う。

(1) 表示画面の作成

- ①画像のズーム
- ②メニュー画面合成
- ③モードの表示

(2) ユーザからの入力に対する処理

- ①画像のズームに関する要求の処理
- ②メッセージ出力
- ③アプリケーションプログラムへの応答

以下、それぞれについて説明する。

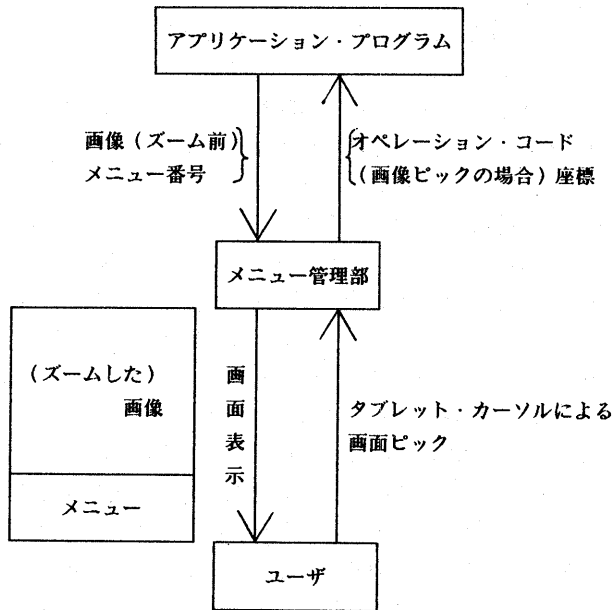


図2 メニュー管理部の外部インタフェース

3.2 表示画面の作成

(1) 画像のズーム

アプリケーション・プログラムが取扱うのはズームしていない元の解像度の画像で、表示のたびにコモン変数に従って画像を拡大する。この処理に約0.2秒かかるので、連続描画プログラムだけがペン先が細い時に拡大画像に直接描込む。そのために、描画タスクだけがメニュー管理プログラムを内蔵している。

(2) メニュー画面の合成

アプリケーション・プログラムから指定されたメニュー・コードに対応するメニュー画面を画像の下に合成する。表示画面は512×480で、キャプテン画像は最大496×384であるので、下側512×96の領域を固定的なメニュー領域としている。

(3) モードの表示

コモン変数の値に基づいて表示を変更する。たとえば色付けメニューには、ピック・全画面・多角形内の3モードがあるが、その時に選ばれているモードだけが色付け表示される。

3.3 ユーザからの入力に対する処理

(1) 画像のズームに関する要求の処理

画像のズームについてはメニュー管理部が管理している。ズームに関する要求、すなわち拡大・縮小・移動・リセットおよびこれらのキャンセルに対しては内部でコモン変数を更新して3.1(1)からの処理をやり直す。

(2) メッセージ出力

次の場合には、メニュー管理部がメッセージを出力する。

①空欄がピックされた時/許されていない画像ピックがされた時

「指定のメニューはありません」

②時間のかかる処理が指定された時 「しばらくお待ち下さい」

このために、下記のオペレーション・コードを定義するテーブルの中で、時間のかかる処理(10秒以上が基準)にはフラグを付けている。

(3) メニュー/画像のピックに応じた処理

表示上、メニュー領域と画像領域は固定しているので、ピックされた座標だけでメニュー/画像のどちらがピックされたのかを決定できる。

画像ズーム以外のメニューがピックされた時には、オペレーション・コードをアプリケーション・プログラムに渡す。メニュー領域内の座標とオペレーション・コードとの対応は、小ブロック単位のテーブルで定義されている。

画像がピックされた時には、それを示すオペレーション・コードを返すと共に、座標をズーム変数で補正して、アプリケーション・プログラムが取扱っている座標に変換してアプリケーション・プログラムに渡す。

3.4 考察

今回開発したソフトウェアを見直すと、領域指定に関する機能を複数のアプリケーション・プログラムが持っている。これらの機能の整理と、メニュー管理部への取込みについての考察をここで行う。

この装置が持っている、領域指定に関する機能には以下のものがある。

(1) 画像に依存しない指定

①幾何図形(円、長方形) --- 描画の中で、内部を塗潰すという意味で領域指定に使っている。

②任意多角形 --- (2)①参照

③自由曲線 --- これは現在、領域指定には使っていないが閉曲線として領域指定に使える。

(2) 画像に依存する指定

①多角形に含まれる連結領域全て --- 色変更と切出しで使っている。

②ピックされた点を含む連結領域 --- 同上

②文字入力部との通信

通信相手をどの状態に戻すか、仕様設定とプログラミング2つの点で困難である。

③フロッピーへの画像書込み

2倍の時間をかけて旧データをセーブしてから書込むか、フロッピーへの書込みを一時保留してフロッピー取外しに制約を加えるかが必要である。

(2) 処理終了前のキャンセル

時間がかかる処理では実行中のキャンセルのニーズはあるが、プログラムが複雑になるので現在行っていない。通信プログラム以外は技術的にはあまり困難ではないと思われる。

(3) キャンセルのキャンセル

通常はあまり意味がないが、画像を自動的に改善する処理、すなわち画像を改善する可能性がある処理に対して意味がある。この装置ではカラー入力での「雑画除去」や限定色表示のための代表色の追加・削除のあとではキャンセルを繰返して処理前後の画像を見比べるようにして効果を上げている。

4. 2 内部での実現について

キャンセルを実現する方法には、状態のセーブ・リストアと逆操作の2つがある。この装置では大部分をセーブ・リストアで実現しているが、一部は画像のセーブを省略して高速化するために逆操作で実現している。

逆操作のほうが複数ステップ遡るキャンセルを実現しやすいと思われるが、前節(2)で述べた処理終了前のキャンセルが不可能になる。

5. おわりに

我々が開発したキャプテン画像入力装置を実例として参照しながら、ユーザインタフェースについて考察した。今後ユーザは進化しないであろうが、計算機は急速に知的になると思われる。その意味でマンマシン系でのマシンの責任はますます大きくなる。そのような状況では、ユーザは最適に呈示されたメニューの中から選択するだけで、計算機はそれでも起こるユーザの間違いへの対応に注力することになるであろう。そのために、ユーザインタフェース管理システム、特にメニュー管理システムと、ユーザの間違いへの対応の一つとしてのキャンセル処理はますます重要になると思われる。この報告がそのような方向に向かう研究開発の参考になれば幸いである。

参考文献

- 1) 森、近藤「キャプテン画像入力装置における画像処理(1) 画像の入力・編集」
情処第34全大, 3D-1 (1987春)
- 2) 近藤、森「 同上 (2) ブロック単位の処理」, 同上, 3D-2 (1987春)
- 3) 森、近藤「 同上 (3) 少色化」, 情処第35全大, 2J-3 (1987秋)

ただし、色変更と切出しでピックの使い方が違う。色変更では1点がピックされるたびにその領域の色を変更する。切出しではユーザは複数の領域を連続してピックし、最後に「切出し実行」メニューを指定することでまとめて切出す。これらの仕様を統一するためには、1点がピックされるたびに処理するほうが望ましい。ただし切出しをそのように変更するためには、原画と切出された画像の2画像同時表示が必要と思われる。

表4. 1編集メニューでの領域指定方法の使用状況

編集メニュー		描 画	色付け/色変更	切 貼 り
領域指定方法				
画依 像存 にし ない	幾何図形	円、四角内塗潰し		
	任意多角形		下記 参照	下記 参照
	自由曲線	(描画だけ)		
画 像 に 依 存 す る	指定領域に 含まれる 連結領域		任意多角形に 含まれる 連結領域	同 左
	指定点を 含む 連結領域	個別	有	
		一括		

ただし切出しをそのように変更するためには、原画と切出された画像の2画像同時表示が必要と思われる。

以上の指定方法の、3つの編集メニューでの使用状況を表2に示す。

この表にあげた領域指定方法は全て、メニュー管理部（あるいはユーザインタフェース管理システム）が管理すべきものと思われる。

4. キャンセル処理

会話型のシステムではユーザは必ず間違いを冒すものと考えなければならない。ユーザの間違いに対する装置側の対策は次の2つに分けられる。

(イ) 回復困難な処理の前には、1ステップ確認操作を入れる。

(ロ) 間違っただけの場合のキャンセルを許す。

ユーザの個性に依存する可能性もあるが、冗長な確認操作は省いて間違っただけの場合にキャンセルをするほうが操作性が良いと思われる。この装置ではその考え方に基づいて、ほとんど全ての操作を直後にだけではあるがキャンセル可能にした。ここでの例を挙げながらキャンセル処理の外部仕様と内部での実現方法について考察する。

4. 1 外部仕様について

(1) キャンセルが困難な処理

この装置で、要求も強くないため直後にもキャンセル不可能になっている3種類の処理について考察する。

①メイン・メニューへのリターン

これは原理的には困難ではない。しかし、全てのアプリケーション・タスクでキャンセルによる再入を可能にするために、全ての状態変数を保存するというプログラムの複雑さだけが問題である。

討 論

4. キャプテン画像入力装置におけるユーザインタフェース

近藤（東芝）

守屋：アプリケーションとメニュー管理部の作業の分担はどうなっているのですか。

近藤：メニュー管理部の仕事は、基本的には表示画面の作成とユーザからの入力に対する処理（本文参照）だけです。

例えば、画像の一部を切り出す仕事を考えると、モード切り換えはメニューで行い、点のピック、領域の指定、領域の切り出しの一連の作業はアプリケーションで行います。つまり、画像ズーム以外のほとんどの仕事（キャンセルも）はアプリケーションが行うわけです。

川合：画像の切り出しが、90%以上の面積を占める連結領域に対して行われるという話がありましたが、それは妥当な数値なのですか。

近藤：領域範囲外の部分が入ることはないので、ほとんどは大丈夫です。

坂下：画像切り出しなどの一般的な操作をメニュー管理部でもってはどうですか。

近藤：アプリケーション側を主導にしたいという考えがありましたので、そうはしませんでした。

坂下：メニュー管理部を柔軟にプログラマブルにすればうまく行くのではないかと思います。

近藤：領域指定などの処理をメニュー管理部でもつことは考えられますが、アプリケーションの機能をうまくモジュール化するのは難しいと思います。