# 認知的負荷を軽減する描画方式の提案と実装

**Reduction of Cognitive Workload on Drawing Editor**

河内谷幸子　五十嵐健夫　松岡聡　田中英彦*

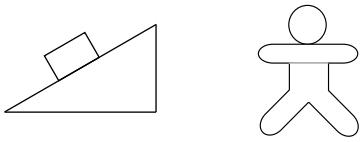
**Summary.** Current computer-assisted drawing editors have sufficient capabilities to draw figures with required precision of a user. However, it takes much time to draw intended figures. First, we analyze the operations of conventional drawing systems using the verbal protocol method and the video records. The experiments have resulted that there exists the cognitive processing in traditional drawing editor. We propose a new drawing editor which reducts the cognitive workload. This system is an interactive beautification system, which extracts various constraints from input strokes, and generates desired diagrams automatically. The user has only to select from multiple candidates, which frees users of the cognitive processing. Results show that our method can reduce cognitive workload and improve the efficiency of drawing editor.

## 1　はじめに

現在広範に普及しているCADシステムは、すべてメニュー選択方式のオブジェクト指向型である。idrawやMacDrawなどの描画エディタもこれに含まれる。これらの描画エディタによって、定規やコンパスを用いた製図と比べて短時間で正確な図を描くことが可能になった。しかし、実際の描画時間は手書きの場合に比べて膨大な時間がかかっているのが現状であり、円に接する直線のように一見単純な関係を実現するのに非常に苦労することがある。より効率的な描画を行うためのアプローチとして、機能を増やしたり、より多くの制約やマクロを描けるようにしたり、操作履歴を利用した補助を行ったり、電子ペンを利用するなどの研究が行われている。しかし、そのほとんどが表層的な機能の改善にとどまっており、描画過程に内在する問題に対する認知的な考察はあまり行われてこなかった。

そこで、我々は描画エディタの持つ基本的な問題点を明らかにするために、発話プロトコル法[2]による描画過程の認知的解析[1]を試みた。本稿の前半では、描きたい絵を描画エディタの機能に変換する過程すなわちプランニングに多くの時間がかかっていることを実験の結果として提示する。その原因としては、描画エディタの機能というのは目に見えない内部表現であるため認知的に負荷が大きいことが考えられる。

* Sachiko Kawachiya, 東京大学大学院 理学系研究科 情報科学専攻, Takeo Igarashi, 東京大学大学院 工学系研究科 情報工学専攻 , Satoshi Matsuoka, 東京工業大学大学院 情報理工学研究科 数理・計算科学専攻, Hidehiko Tanaka, 東京大学工学部



(a) (b)

**図 1.** 実験に用いた絵

次に我々はプランニング過程を描画エディタが負担するような、新しいアプローチを提案する。具体的には、ユーザが描きたい絵をスケッチすると、システムが従来の描画エディタが機能として持っている変換を組み合わせ、考えられる整形変換候補を visible に表示する。この方式によれば、ユーザは提示される候補から目的の図形を直接選ぶだけでよく、視覚的イメージからエディタ操作への変換から解放される。本稿の後半では、このような新しいアプローチと、それに基づいたスケッチの対話的整形システムの実装を紹介する。

なお、本稿で対象とする図形のクラスはビジネスグラフィクス図形であり、スケッチ画や、数値的な正確さを要求される CAD 図面は対象としない。

## 2 既存の描画ツールはなぜ時間がかかるのか？

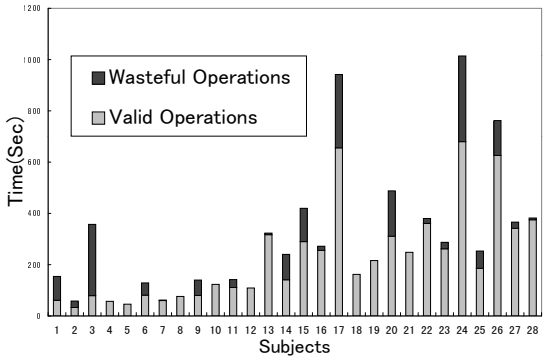
ここでは既存のオブジェクト指向型描画エディタの問題点を実験により明らかにする。

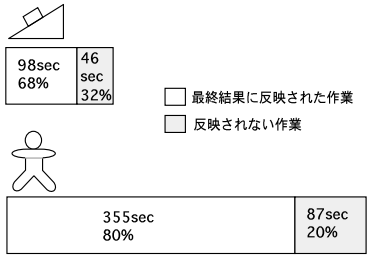
### 2.1 描画実験

**＜実験方法＞**
図 1のような 2 つの絵を、オブジェクト指向型描画エディタの中で広範に利用されている MAC 上の描画エディタ CANVAS (MacDraw の高機能版) を用いて被験者に描いてもらう。被験者は描画エディタを 3 年以上利用している者 14 名。あらかじめ CANVAS の機能について説明を行う。描画中に操作意図を発話してもらい、その様子をビデオに録画して分析する。

**＜実験結果＞**
**(結果 1) グローバル分析**
録画したビデオをもとに、作業全体のうち無駄に終った作業がどのくらいあるのかを調査した。ここで、いったん描いたが使わずに消してしまった部分を描くのにかかった時間を「試行錯誤の時間」とよぶことにする。その結果を図 2に示す。図 2(a) は各被験者ごとの結果で、Subject の 1 〜 14 は図 1(a) の場合、15 〜 28 は図 1(b) の場合である。図 2(b) は図 2(a) の平均である。被験者および各描画作業によって、ばらつきが大きいが、傾向として、2 〜 3 割の時間が結果として無駄になる作業に費やされている。

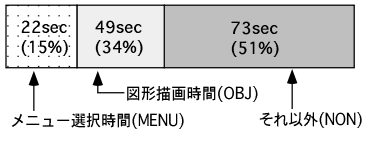
**(結果 2) ローカル解析**
各描画過程において、

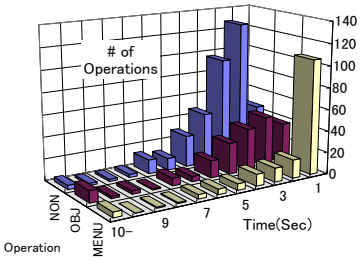



(a) 被験者別分析 (1 〜 14 : 図 1(a),15 〜 28 : 図 1(b)) (b) 14 名の平均

**図 2.** 試行錯誤の時間




(a) 14 名の平均 (b) 1 操作あたりの時間分析

**図 3.** 図 1(a) のメニュー選択 (MENU) ／図形描画 (OBJ) ／それ以外 (NON)

- MENU: メニューからコマンドを選んでいる時間
- OBJ: オブジェクトを直接操作している図形描画時間 (図形の配置、移動など)
- NON: 上記以外の時間

の時間を計測した。図 1(a) の絵を描いた場合についての 14 名の平均を図 3(a) に示す。何もしていない時間 (NON) がかなり長いことがわかる。また、発話内容からメニュー選択時間や図形描画時間の間でも次の作業がなかなか決められすに迷っている時間が長いこともわかった。

次に図 1(a) の場合の全員のデータを、メニュー選択／図形描画／それ以外の 3 種について 1 操作ごとにかかる時間を分析したのが図 3(b) である。 1 操作に 8 秒以上かかる場合がかなり存在することがわかる。

＜考察＞

特に重要な問題点として、次の 2 点があげられる。

- 目的の絵を描くために描画エディタの機能をどう使うかという戦略を考えるのに、多くの時間を要している。

| | 物理的操作時間 (Sec) | | |
|---|---|---|---|
| | 4.0 | 3.0 | 2.0 |
| 図 1(a) | 22.5% | 30.9% | 45.1% |
| 図 1(b) | 17.6% | 26.7% | 41.5% |

**Table 1.** いくつかの物理的操作時間の仮定値を与えた時の認知的操作時間

- まちがった戦略を用いていることに気づくまでに多くの時間がかかる。

例えば図 1(a) の「斜面と物体」の絵を描く場合、斜面の三角形を 3 つの線分として描き、次に長方形を描いて回転し、斜面に接するように移動するという描き方がある。この場合、回転の精度が合わずに斜面の傾きと長方形の傾きを合わせられない場合があるが、このような時に合わない理由がすぐにわからず、しばらく試行錯誤をつづけることが多い。また、グリッドを利用する場合には、あらかじめうまくいくように数を数えてから描画しなければならず、そのために時間がかかってしまう。

### 2.2 認知的負荷の計算

上記の実験で、戦略を考えるための認知的負荷が大きいことが明らかになった。戦略を考えるための認知的な負荷の時間は、必ずしも上記の「メニュー選択でも図形描画でもない時間」がすべてであるとはいえない。逆にメニュー選択や図形描画の時間の中にも戦略のための認知的負荷時間が含まれるであろう。正確な認知的負荷時間を計算することは難しいが、ある程度の幅をもって計算をすることは可能である。そのために、我々は戦略を必要としない図で実験をし、その差分をとる実験をした。被験者に図 4(a) のような四角 5 つ描いてもらい、その後で四角をクリック選択して左右反転する作業を左の四角から順番に行ってもらった。ただし、描画方法および編集方法はあらかじめ全部説明した。四角を左右反転しても形状は変化しないが、メニュー選択による編集作業の一例として行った。結果は図 4(b) のようになった。図 3(b) に比べて 1 作業あたりの時間は 4 秒以内になっている。この 1 作業あたりの時間を物理的作業時間と考え、前節での実験での作業時間からひくことによって、認知的作業時間を計算すればよい。ただし、作業時間は作成する図形の種類や作業の個人差などに依存することを考慮し、ある程度の幅をもたせて計算する。

$$\text{認知的操作の時間} = \sum(\max(0, t(k) - \text{threshold})) \quad k \in \text{全ステップ} \qquad \text{[式.1]}$$

ただし、

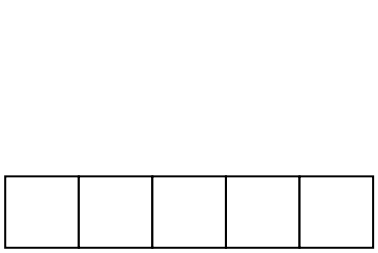
$t(k)$ = ステップ $k$ に費やされた時間,

threshold = 物理的操作に必要な時間

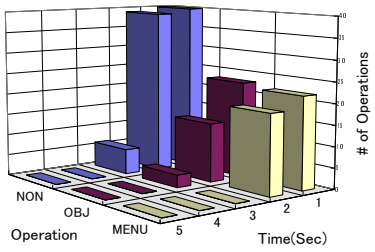
式 1 の threshold として、2,3,4 秒をそれぞれ代入して認知的負荷の時間を計算したのが表 1である。認知的作業に費やされる時間は、楽観的に見た場合でも全体の 17 〜 20 % 、悲観的に見た場合には全体の 41 〜 45 % も占めることがわかる。

### 2.3 描画に時間がかかる理由

以上のように認知的負荷が大きくなる問題の原因として考えられるのは、visibility の欠如である。すなわち、手書きの図では描いたものそのままが図形情報であって、隠れ



(a) 実験に用いた絵




(b) 1 操作あたりの時間分析

**図 4.** 戦略のない絵の実験

た構造がない [8]。これに対して従来の描画エディタでは、目に見えている構造とは別に、線・円・矩形というようなプリミティブや複写・回転というような編集機能の組み合わせとしての内部表現があり、操作に際しては表層の構造とは別にその内部構造をイメージしなくてはならない。目に見える図形表現の階層と描画エディタの内部構造である操作コマンドの階層との間の変換が難しいため、多くの時間を要すると考えられる。我々はこの変換のために必要な戦略をプランニングとよぶことにする。

## 3 描画エディタの操作モデル

### 3.1 モデルの説明

前節の実験結果から従来の描画エディタを用いた場合の処理の流れとして図 5のようなモデルが考えられる。描画エディタで描く場合、描きたい絵 (i) のイメージから描画エディタのプリミティブや編集機能のコマンド列 (ii) へ変換するためのプランニング (A) が行われる。コマンド列 (ii) はエディタの操作 (B) を介して中間図形 (iii) や、描画終了図形 (iv) へと変換される。中間図形と描画終了図形の間では位置合わせなどの画面上の微調整 (C) のくりかえしがある。操作 (A)(B) や (ii) のデータは、表層的な形状とは乖離した抽象的な階層の中にあり、描きたい絵 (i) や中間図形 (iii)、描画終了図形 (iv) と画面上の調整 (C) は具体的な図形表現である。さらに (iv) から (i) へのフィードバックが行われ、対話的に修正される。

### 3.2 従来のアプローチについて

従来の描画エディタに関する研究を上記のモデルで説明する。機能を増やしたり、より多くの制約やマクロを描けるようにするアプローチ [4][5] は (C) の作業をなくし図の精度を高くしようというものであるが、結果として (ii) にあたる記号レベルを与えるためのプランニングの時間 (A) を長くしてしまうものといえる。さらにこの方法では意図した図形を制約充足によって得ることが難しく、試行錯誤に時間をとられるという問題もある。操作履歴を利用した補助、いわゆる PBE の Mondrian[10] や Metamouse[11] は (B) や (C) での単純作業の反復回数を減らそうという試みであるが、最初の作業のためのプランニングをなくすことはできない。 [3] 等いくつかの電子ペンを利用した描画システムの場合、 MacDraw のようなメニュー方式とは表層の操作方法が異なるが、

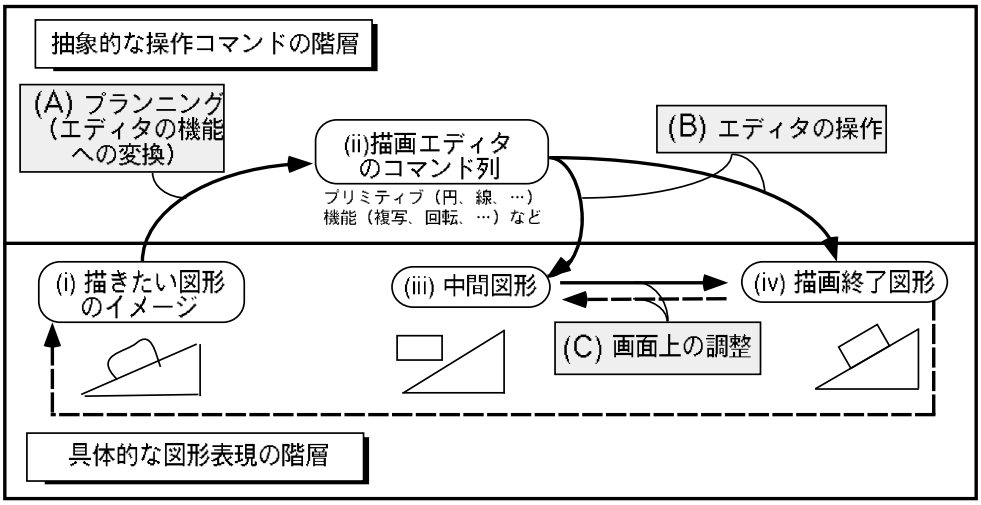



**図 5.** 描画エディタの操作モデル

実際にジェスチャーによって入力されるものはメニュー方式と同一であり、さまざまな制約をみたす図形を描くためには通常の描画エディタと同様な操作 (A)(B) を使う必要がある。また、図形の整形システムも提案されている [7] が、これらのシステムの目的は基本的に位置合わせ、すなわち (C) の作業をなくすだけのものであり、操作の構造自体はあまり変わっていないといえる。

また、ごく最近では効率的な描画手順を教育することにより、(B) の作業回数を減らせるという研究 [9] もある。効率的な描画手順とはすなわちプランニングにほかならない。つまりこの研究は「プランニング」の存在を裏づける研究といえる。しかし、このアプローチはマクロ化された作業のプランニングの負担は軽減できるが、新しい作業では負担は減らない。

以上のように従来のアプローチはすべて、基本的なプランニングは人間が行い、計算機はプランニングに基づいた作業の補助をするというものであるといえる。しかし、前節の実験でも明らかなように、実際にはプランニングする部分 (A) こそがもっとも時間のかかる部分であり、この部分をなくすことこそが本質的な問題の解決になると考えられる。

### 3.3 理想的な描画システムとは

以上のような実験結果および提案したモデルから、描画作業を根本的に改善するために必要な要件は次のようにまとめられる。

「描画におけるプランニングおよび試行錯誤の時間を減らすために、図を得る方法が自明であり、複雑な操作の必要がないようにする。」

これはまた、「図 5のモデルにおける抽象的な操作コマンドの階層 (A)(B)(ii) からユーザを解放する。」とも言いかえることができる。

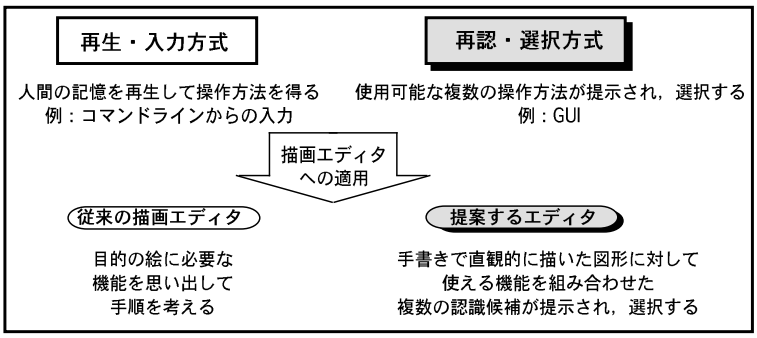



**図 6.** 再認・選択方式の採用

## 4　プランニング時間を削減する描画エディタ

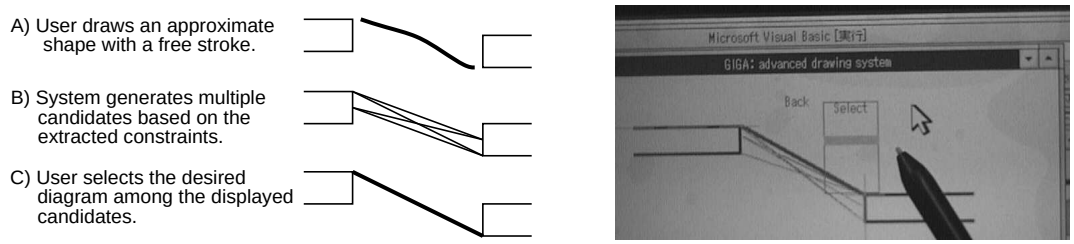
以上のような実験および考察をもとに、「作業の効率の悪さの原因である抽象的操作」に相当する部分を解消し、具体的な図形のみを扱うことによって描画を進める、新しいタイプの描画システムを提案する。

まず、具体的なな図形のみを扱えるように、手書きスケッチを採用する。ただ、これまでのスケッチ認識系描画エディタのように、編集時にはMacDrawと同じような方法をとるのでは抽象的作業が必要になってしまう。そこで認識結果として、交点の処理や対称といった編集まで行ったものを出せば抽象的作業はなくなる。しかし、どのような編集がユーザの意図かを判断するのは非常に難しい。そこで我々は可能な認識結果をすべて提示しユーザがこれを選択すればよいと考える。

Apple社のヒューマン・インタフェース・ガイドラインによれば、GUIのように使用可能な操作がすべて提示されてこれを選択するような再認・選択方式は、コマンドラインからの入力のように人間の記憶を再生して情報を得る再生・入力方式よりも優れたインタフェースである[12]。我々の提案はこのような再認・選択方式を描画エディタのインタフェースに採用するものともいえる。

具体的なインタラクションを説明する。まずユーザは描きたい絵の概形をエディタ上にフリーストロークで直接描く。次にシステムは本来描画エディタが機能として持っていた変換を組み合わせ、考えられる整形変換候補を画面に表示する。例えば図7(a)でAようなスケッチが描かれた場合、可能性のある整形結果としてBのような候補を提示する。デフォルトの候補で満足な場合はそのまま作業をつづけ、気に入らない場合にはその他の候補を選ぶことになるが、すべての候補は画面上に目に見える形で表示されているのでユーザはその中から選ぶだけでよい。候補は好きな場所に開くことのできるスライダーの操作で選ぶ方法を採用した(図7(b))。ほかに候補をじかに選択する方法も考えられるが、隣接する候補の選択を容易にするためにスライダーを用いる方法を選んだ。以上のような候補を選択する方法により、ユーザは内部での表現や抽象的な操作を扱うことなく、目に見える図形だけを扱いながら描画を進めることができる。

従来のペンスケッチによる整形システム(Apple Newton, GO Penpoint, その他市販描画ソフトのスケッチ整形機能)と異なる点は、端点の接続や円への変換といった簡単で局所的な処理だけでなく平行や対称、形状の類似性といった複雑で広範な制約に基づく処理を行う点である。この高次の制約の利用によってプランニングの手間を省きつつ正確な図を手早く描くことができるようになる。例えば、対称な図形を描く場合に




(a) 複数の整形候補の提示と選択　　(b) 動的に開けるスライダーの利用

**図 7.** インタラクションの説明

は、従来は複写→反転→移動というような複数の操作を行わなければならなかったが、本システムでは対称な図形が自動的に生成されるので、ユーザは細かな作業にとらわれず、形状のデザインに集中することができる。

### 4.1 複数の整形候補の生成と提示

以上のようなシステムを実現しようとした場合、問題となるのは、整形結果の不確実性である。すなわち、さまざまな制約を考慮する結果、矛盾した制約が出てきて整形図形に多義性が生じる場合があり、また、整形結果の善し悪しの判断はもともと主観的なものであるために 100% の認識はありえないからである。この問題を解決するために、本システムでは複数の認識候補を制約の組みかえにより自動生成し、その中から対話的に最適な図を選択させる方法をとっている。

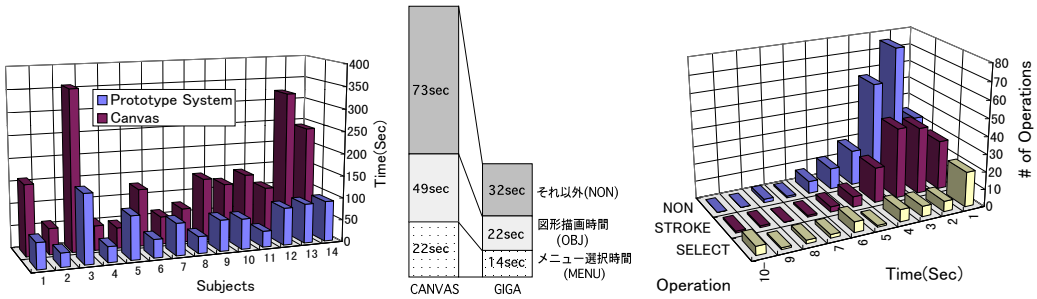
複数候補の生成部では、まず、入力ストロークとすでに描かれている図形との関係を調べて、知覚的制約を重みつきの制約(制約階層)としてとりだす。このとき、見ために明らかな制約を強く、自明でない制約を弱く設定する。次に、この制約階層を強い順に解いて第一候補を生成する。この後、満たされなかった制約を強くして解き直すことを繰り返していくつかの候補を生成する。

生成された候補は画面に提示され、第一候補が棄却された場合にスライダーによってその場で選択できるようにする。この時、ユーザに対して それぞれの候補がどのような制約を満たしているのか(どの部分と垂直でどの部分と接しているか、など)が画面に表示されるので、素早く確実に作業を進めることができる。第一候補は赤、それ以外の候補は青、第一候補に使われた制約の提示は緑、というような色わけ提示も行っている。例えば、ある線 $l1$ を描いた時に、それ以前に描いた別の線 $l2$ と垂直な候補を提示する場合は、$l2$ が緑となり、さらに垂直を示す緑色の十字マークが $l1$ と $l2$ の交点に表示される。

### 4.2 実装環境と評価

上に述べたインタラクションを採用したプロトタイプシステム **GIGA (Graphics with Interactive Geometrical Assistance)** は Windows 上の Visual Basic で開発されており、現在、高速化のために Visual C++ に移行中である。

制約としては、端点接続をはじめとして平行や垂直、xy 座標の揃え、対称性や類似性などをすでに実装している。整形結果を得るまでの反応時間は、図 9程度の図で 1




(a) 被験者別の CANVAS との比較　　(b) (a) の平均　　(c) 1 操作あたりの時間分析

**図 8.** プロトタイプシステム **GIGA** の操作分析 (マウス利用)

| | 物理的操作時間 (Sec) | | |
|---|---|---|---|
| | 4.0 | 3.0 | 2.0 |
| 図 1(a) | 11.8% | 19.3% | 34.3% |

**Table 2.** プロトタイプシスム **GIGA** の認知的操作時間

秒以下であり、対話的な作業が可能である。

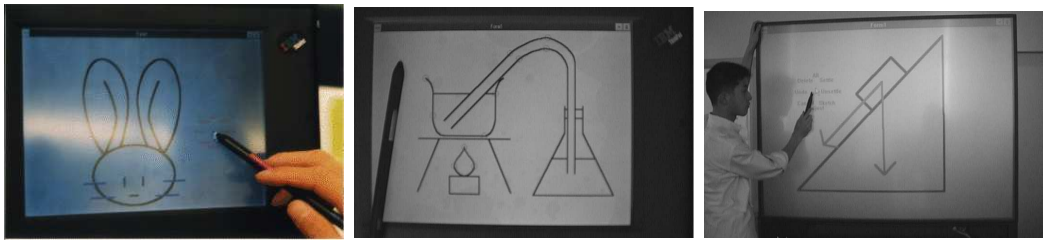
本システムの有効性を確かめるために、プロトタイプシステムを利用して前述の実験に用いた図 1(a) の絵を被験者に描かせる簡単な実験を行った。従来の描画エディタと比較するため、ペンではなくマウスを用いた。被験者それぞれについての CANVAS と GIGA の操作時間の比較を図 8(a) に示す。平均で比較すると 47 % にまで短縮された (図 8(b))。さらに、メニュー選択時間 (MENU) ／図形描画時間 (STROKE) ／それ以外の時間 (NON) の 3 種についての 1 操作あたりの時間 (図 8(c)) も CANVAS の場合 (図 3) に比べて少ない。式 1 に準じて認知的作業時間を計算したところ、表 2のようになった。表 1と比べても認知的作業時間は短縮された。

また、発話内容などから、既存のエディタでの実験でみられたような、根本的に誤っている操作に気づかず無駄な試行錯誤を繰り返す現象を起こすことはなかった。

なお、提案した描画手法およびプロトタイプは、本来入力デバイスに依存しない一般的なものであるが、スケッチによる方法は特にペンによる入力に適していると考えられるので、IBM 社のペンコンピュータ ThinkPad 360P (図 9(a)(b)) および XEROX 社の電子ホワイトボードシステム LiveBoard (図 9(c)) といったシステム上でも積極的にテストを行っている。

## 5　まとめと今後の課題

本稿では描画エディタにおける描画過程の分析を通じて従来の描画システムの問題点を明らかにし、さらにこの問題を解決するシステムを提案した。そして、スケッチによる直観的な描画、知覚的制約の認識、対話的な認識候補の選択を柱とした、スケッチ整形方法とプロトタイプシステム **GIGA** を紹介した。従来の描画エディタでは描きたい絵をエディタの機能の組合せへと変換するためのプランニング時間を多く要していたが、本システムでは変換候補が出されるため、プランニング時間を大きく削減でき、また利



(a) 耳の対称なうさぎ　(b) 化学実験装置　(c) 電子ホワイトボード利用

**図 9.** プロトタイプシステム **GIGA** による描画例

用方法の習得も簡単である。

今後は、さらに優れた知覚的制約認識や変換候補提示のインタラクションを検討したい。また、電子ホワイトボードを利用した会議システムや CAI への応用も考えたい。

**謝辞** 本研究の実装にあたり、XEROX 社の Liveboard を貸してくださった慶應義塾大学環境情報学部の徳田英幸教授に感謝する。

## 参考文献

[1] Jakob, N., "Usability Engineering", Academic Press, 1993.

[2] Ericcson,K.A., Simon,H.A., "Protocol Analysis: Verbal Reports as a Data", MIT Press.

[3] Apte,A., Kimura,D., "A Comparison Study of the Pen and the Mouse in Editing Graphic Diagrams", *Proc. of Visual Languages '93*, pp. 352–357, 1993.

[4] Bier,E.A., Stone,M.C., "Snap Dragging", Computer Graphics, Vol.20, No.4, pp. 233–240, 1986.

[5] Borning,A., "The Programming Language Aspects of ThingLab, A Constraint-Oriented Simulation Laboratory", ACM Trans. on Program. Lang. Syst., Vol.3, No.4, pp.353–387, 1981.

[6] Nelson,G., "Juno, A Constraint-based Graphics System", Computer Graphics, Vol.19, No.3, pp. 235–243, 1985.

[7] Pavlidis,T., VanWyk,C.J., "An Automatic Beautifier for Drawings and Illustrations", Computer Graphics, Vol.19, No.3, Proc. of SIGGRAPH '85, pp. 225–234, 1985.

[8] Saund,E., Moran,T.P., "A Perceptually Supported Sketch Editor", *Proc. of UIST'94*, pp. 175–184, 1994.

[9] Bhavnani,S.K., John,B.E., "Exploring the Unrealized Potential of Computer-Aided Drafting", *Proc. of CHI'96*, pp. 332–339, 1996.

[10] Lieberman,H., "Dominoes and Storyboards: Beyond "Icons on Strings"", *Proc. of VL'92*, pp. 65–71, 1992.

[11] Maulsby,D.L., Witten,I.H, Kittlitz,K.A., "Metamouse: Specifying Graphical Procedures by Example", Computer Graphics, Vol.23, No.3, pp.127–136, 1989.

[12] "Macintosh Human Interface Guidelines", Apple Computer, Inc., 1992.