予測 / 例示インタフェース

増井俊之
ソニーコンピュータサイエンス研究所
masui@csl.sony.co.jp

計算機操作の問題点

- つまらない繰り返し操作
  - (例 1) 連続する 100 行の字下げ
    - プログラム? マクロ? 特殊機能を探す?
  - (例 2) パラメタを変えて再計算 / コンパイル
    - プログラムを作るべき? Makefile を使うべき?
  - (例 3) 図形レイアウト
    - ルーチンワークとあきらめる?

解決法

- 機能強化
  - アプリケーションが肥大
- マクロ作成 / 使用
  - 一般に面倒
- プログラム作成
  - 誰もがプログラマになる必要がある
  - プログラムを簡単に作れない場合がある

予測 / 例示インタフェースによる解決

- 操作履歴から次の操作を予測
  - 繰返し操作の検出
  - パラメタの予測
    - 1,2,3 の後に 4 を予測
- 例示によるマクロ作成

例からのプログラミング

- **目標 = 例データからプログラムを自動的に作成**
- Programming by Example (PBE)
- Programming by Demonstration (PBD)

予測 / 例示インタフェース手法の利点

- 繰返し作業を楽にする
  - マクロを簡単に作成
  - 複雑な機能を用意する必要がない
- プログラミングが不要
  - プログラマでででなくても使える
  - 具体的操作だけすればよいので抽象的思考が不要
  - 結果が常に目に見えているのでわかりやすい

エンドユーザプログラミングへの応用

- テキストでプログラミングする必要がない
- (例) メールの仕分けプログラムの作成
  - プログラムを書く場合
    - `if(/From: masui.*/){ ...`
  - 例示インタフェースの場合
    - 実際の仕分け操作から推論

GUI プログラミングへの応用

- **実際の** GUI **操作から** GUI **プログラムを生成**
- **画面の表示やふるまいとプログラムのギャップを解消**
- e.g. **インタフェースビルダ**
  - ◇ **画面表示とプログラムのギャップを解消**

予測 / 例示インタフェースの分類 [Myers]

4 種類に分類

- 知的処理を行なうかどうか
- プログラムの作成を支援するかどうか

単純 / 複雑

- 単純なもの
  - 操作記憶 / 再実行
  - アプリケーション知識からの予測
  - 統計情報などからの予測
  - 繰返しパタンの検出
- 複雑なもの
  - 自動適応
  - 機械学習
    - Learning from Example (LFE)
    - Dialog-based Learning (DBE)
  - 「エージェント」

プログラミングの有無

- **プログラミングが主目的のもの**
  - **自動プログラミング**
  - Programming by Demonstration (PBD)
  - Programming by Example (PBE)

予測に使用されるデータ

- 無意識的なもの
  - 操作履歴 (計算トレース)
  - 予測実行時のコンテクスト
  - 別のテキスト
  - 辞書 (他人による例示)
- 明示的に指定
  - プログラミングに近い
- 両者の組み合わせ
  - 自動適応 + 意識的に例を与えて「教育」

例示インタフェースでよく使われる手法

- 操作履歴, コンテクスト, 辞書の利用
- アプリケーション依存のアドホックな知識の利用
- ユーザの介入による推論誤りの修正
- 例示データの生成の自動化
- ユーザ操作やプログラムのビジュアルな表現

単純な予測インタフェースの実例

- UNIX **シェル**
- Reactive Keyboard
- **操作予測電卓**
- Dynamic Macro
- Eager
- Smart Make
- **履歴利用電卓**

UNIX シェル、 Emacs

- **省略形による再実行**
  - !! (**直前コマンドの再実行**)
  - !c ("c"**で始まる直前コマンドの再実行**)
- **補完** (completion)
  - **ファイル名などの一部分のみ指定**
- Dynamic abbreviation
  - **既出の文字列を静的 / 動的辞書として利用**

Reactive Keybord [Darragh]

- シェルのキーストローク列の頻度情報から次の入力文字列を予測
  - e.g. "ls -l" を繰り返し起動すると、 "l"を入力しただけで残りを予測
- 打鍵が非常に遅い場合に有効
- PPM (Prediction by Partial Match) 法

PPM 法

- 文字出現頻度情報から次の出現文字を予測
- テキスト圧縮に有効
  - ◇ 予測が正しいほど高圧縮率
- e.g. "abracadabra"に続く文字の予測

  "a"の後には "b"が 2 回, "c", "d"が 1 回ずつ出現した実績があり，

  "ra"の後には "c"が 1 回出現した実績があり，

  "bra"の後にも "c"が 1 回出現した実績があるといった出現頻度の荷重和をとって次の文字の予想出現確率を計算

テキスト圧縮

- **辞書式** (LZ **系**)
  - ◇ **出現した文字列を辞書に格納**
  - ◇ **再び出現した文字列は辞書へのポインタで表現**
- **予測 + 符号化**
  - ◇ PPM **法**
  - ◇ **ハフマン符号化 / 算術符号化**

ハフマン符号化によるテキスト圧縮

“a”, “b”, “c”, “d” の 4 文字からなるテキストの圧縮

- 出現確率が不明の場合
  - 各文字に 2 ビット必要
- “a” の確率が 50%, “b” が 25%, “c”,“d” が 12.5% の場合
  - “a” を 0, “b” を 10, “c” を 100, “d” を 101 と符号化
  - 1*0.5 + 2 * 0.25 + 2 * 3 * 0.125 = 平均 1.75 ビット / 文字
- 算術符号化 = ハフマン符号化の拡張

操作予測電卓 [Witten]

- PPM **法利用**
- 1 + 2 = , 1 + 3 = , ... **といった操作列から** "1 + ", "=" **などの共通部分を予測**
- (**例**) x * exp(1-x) **の計算**

```
.1 mc m+= +/- + 1 = exp * mr =
.2 mc m+= +/- + 1 = exp * mr =
.3 mc m+= +/- + 1 = exp * mr =
.4 mc m+= +/- + 1 = exp * mr =
```

Dynamic Macro [増井]

- Emacs 上での操作の再実行
- **操作履歴から繰返しパタンを抽出**
- **あらゆる繰返し操作に対して有効**
- **マクロ登録が不要**

Dynamic Macro の動作

- **2 回続けて全く同じ操作列が実行された後で繰り返し実行キーが入力されるとその 1 回分を繰り返す**

“abcabc” + [REP] → “abcabcabc”

- **そのようなパタンが複数存在するときは最長の繰り返しパタンを選択**

“bbabb” + [REP] → “bbabbabb”

Dynamic Macro の動作 (Cont'd)

- **直前の操作列と全く同じ操作列が以前の操作列中に存在するとき、それらの間の処理を実行する**

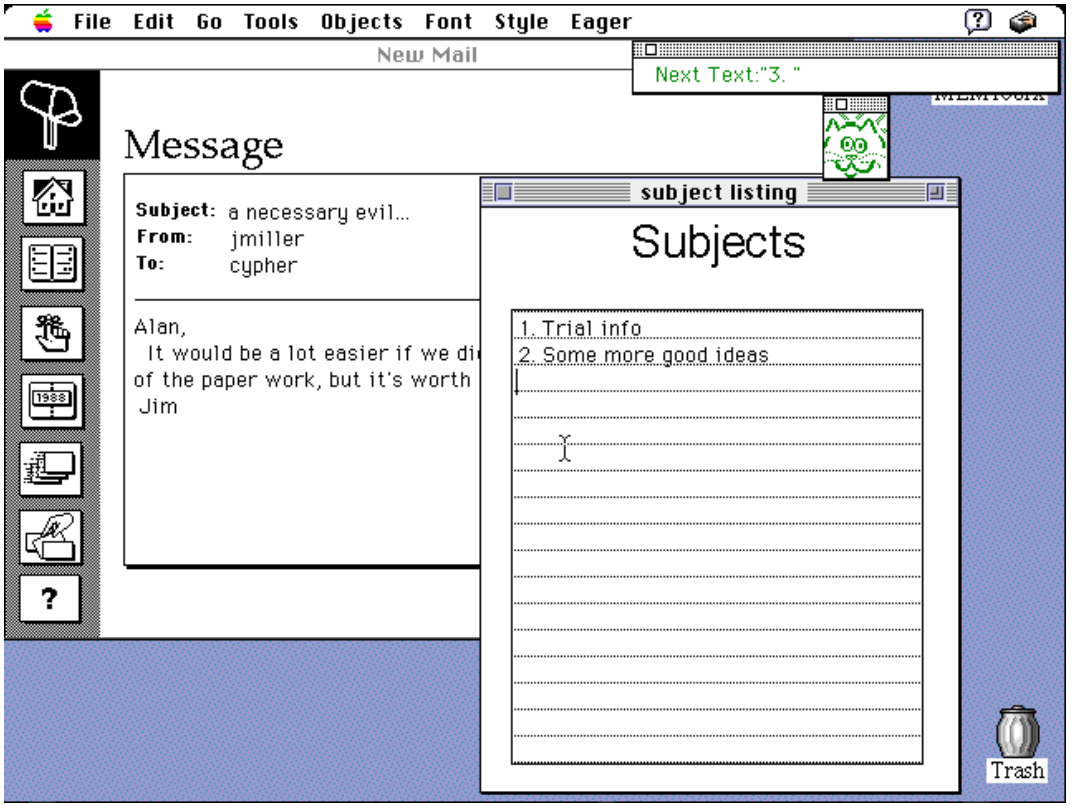
"abcdeab" + [REP] → "abcdeabcde"

- **もう一度繰り返し実行キーが押されると繰り返しパタンを実行する**

+ [REP] → "abcdeabcdeabcde"

Eager [Cypher]

- HyperCard 上の繰り返し操作から次の操作を予測
  - システムは常にユーザ操作を監視
- 次の操作を先取りしてユーザに提示
  - 提示された操作と同じ操作をユーザが実行した場合は推論の肯定とみなす
  - 推論を無視して操作を続けた場合は否定とみなす
- 正しい予測を行なっているとユーザが確信した場合は残りを自動実行可能


File Edit Go Tools Objects Font Style Eager
New Mail
Next Text:"3. "
Message
Subject: a necessary evil...
From: jmiller
To: cypher
Alan,
It would be a lot easier if we di
of the paper work, but it's worth
Jim
subject listing
Subjects
1. Trial info
2. Some more good ideas
Trash

Smart Make

- シェル上のコマンド履歴からファイル / コマンド間の依存関係を抽出し Makefile を生成
- コマンド操作 = 暗黙的な依存関係の例示
- 依存関係の抽出法
  - コマンドの字面からだけではわからない
  - ファイルの生成 / アクセス時間から判断

Smart Make 使用例

```
101% grep a /etc/passwd > passwd.a
102% grep b /etc/passwd > passwd.b
103% tar cvf passwd.tar passwd.a passwd.b
a passwd.a 2 blocks
a passwd.b 3 blocks
104% compress < passwd.tar > passwd.tar.Z
105% uuencode passwd.tar.Z passwd.tar.Z > passwd.UU
106% ls
passwd.UU   passwd.a    passwd.b    passwd.tar  passwd.tar.Z
107% mm
passwd.tar.Z: passwd.tar
        compress < passwd.tar > passwd.tar.Z
passwd.tar: passwd.a passwd.b
        tar cvf passwd.tar passwd.a passwd.b
passwd.b:
        grep b /etc/passwd > passwd.b
passwd.a:
        grep a /etc/passwd > passwd.a
```

履歴利用電卓

- Undo **操作を活用**
- **値を変えて再計算**
- **例**: **華氏の計算**

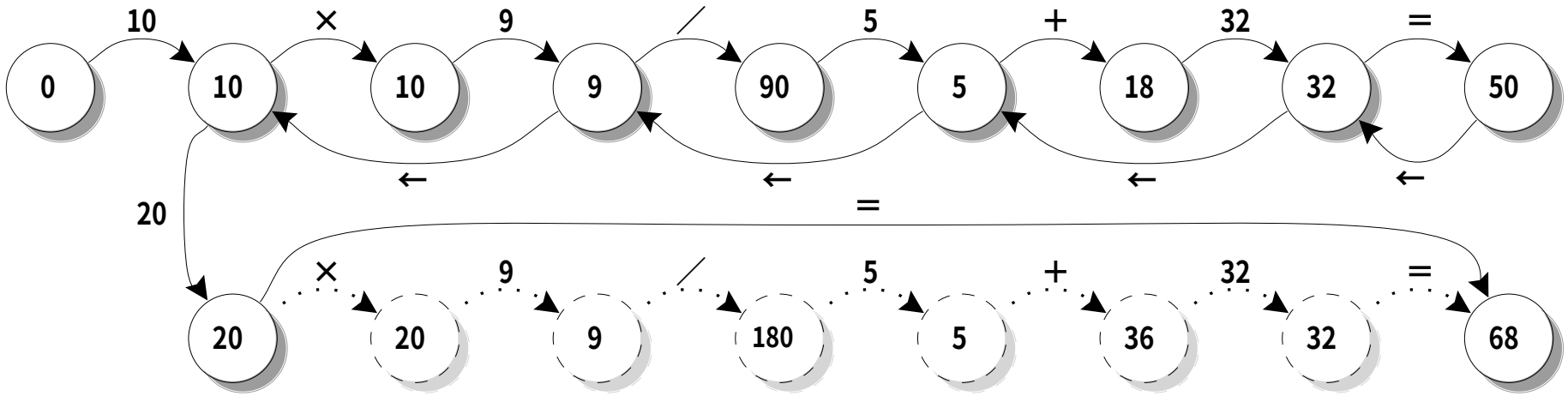
キー入力	表示
10	10
×	10
9	9
／	90
5	5
＋	18
32	32
＝	50

値を変えて再計算

後退キーを Undo に使用

キー入力	表示	
	50	前の表示
後退キー	32	最後に入力されたパラメタが表示される
後退キー	5	さらに前のパラメタが表示される
後退キー	9	
後退キー	10	最初に入力したパラメタの表示
20	20	新しいパラメタ (20℃) を入力
＝	68	20℃に対して華氏の計算を再実行
30	30	さらに別のパラメタ (30℃) を入力
＝	86	30℃に対する再計算結果
…		

再計算の状態遷移



例示インタフェース

- システムに対して明示的に例データを与え、ユーザの意図や好みを推論させる

例示インタフェースの実例

- Editing by Example
- Text Editing from Example
- SmallStar
- Edward
- Pursuit
- Triggers
- Meptamouse
- Chimera
- Mondrian
- Peridot
- Matquise
- KidSIM

- Pavlov
- Layout by Example
- IMAGE
- GP Layout
- Gold

Editing by Example [Nix]

- 編集前のテキストと編集後のテキストの組の例をシステムに提示
  - ユーザの実際の操作は考慮しない
- 正規表現による文字列置換のサブセットを使用
- システムが変換規則を推論し、残りのテキストに適用
- (例) `@i[O.K.]` → `{\sl O.K.}` から `@i[(1)]` → `{\sl (1)}` を推論

TELS [Mo]

- テキストエディタ上での GUI 操作の繰り返しを検出 / プログラム生成
- 挿入 / 削除 / 移動 / 選択の 4 種類の編集操作のみ
- 各操作の前後でのコンテキストの汎化によりループを検出
  - “空白文字の後の数字 222-3456 の選択”、“空白文字の後の数字 234-5555 の選択”

↓

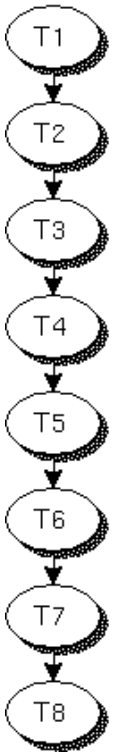
  - “空白文字の後の数字 2**-**5* の選択の繰り返し” というプログラム生成
- 正規表現による文字列置換のサブセットを使用

編集前 / 後のテキスト

```
John Bix, 2416 22 St., N.W., Calgary, T2M 3Y7. 284-4983
Tom Bryce, Suite 1, 2741 Banff Blvd., N.W., Calgary, T2L 1J4. 229-4567
Brent Little, 2429 Cherokee Dr., N.W., Calgary, T2L 2J6. 289-5678
Mike Hermann, 3604 Centre Street, N.W., Calgary, T2M 3X7. 234-0001
```

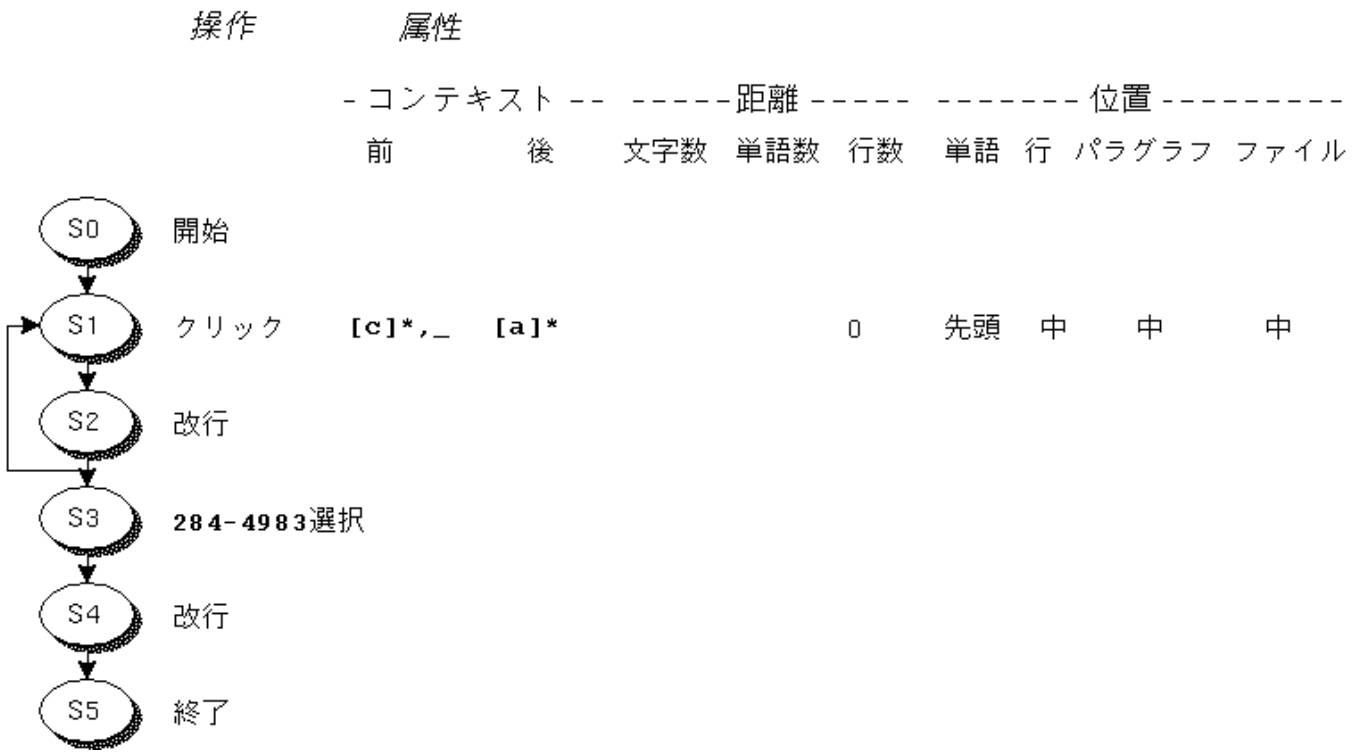
```
  John Bix,
  2416 22 St., N.W.,
  Calgary,
  T2M 3Y7.

  Tom Bryce,
  Suite 1,
  2741 Banff Blvd., N.W.,
  Calgary,
  T2L 1J4.

  Brent Little,
  2429 Cherokee Dr., N.W.,
  Calgary,
  T2L 2J6.

  Mike Hermann,
  3604 Centre Street, N.W.,
  Calgary,
  T2M 3X7.
```

プログラム生成



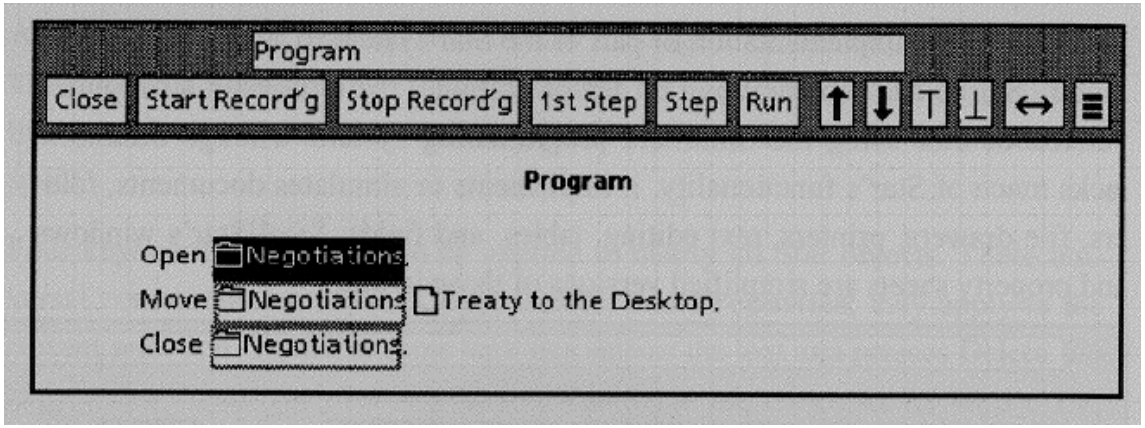
	操作	*属性*								
		--コンテキスト--		-----距離-----			--------位置----------			
		前	後	文字数	単語数	行数	単語	行	パラグラフ	ファイル
T1	クリック	**Bix, _**	**2416**	+10	+2	0	先頭	中	中	中
T2	改行									
T3	クリック	**N.W.,**	**Calgary**	+19	+4	0	先頭	中	中	中
T4	改行									
T5	クリック	**Calgary,**	**T2M**	+9	+1	0	先頭	中	中	中
T6	改行									
T7	**284-4983**選択									
T8	改行									

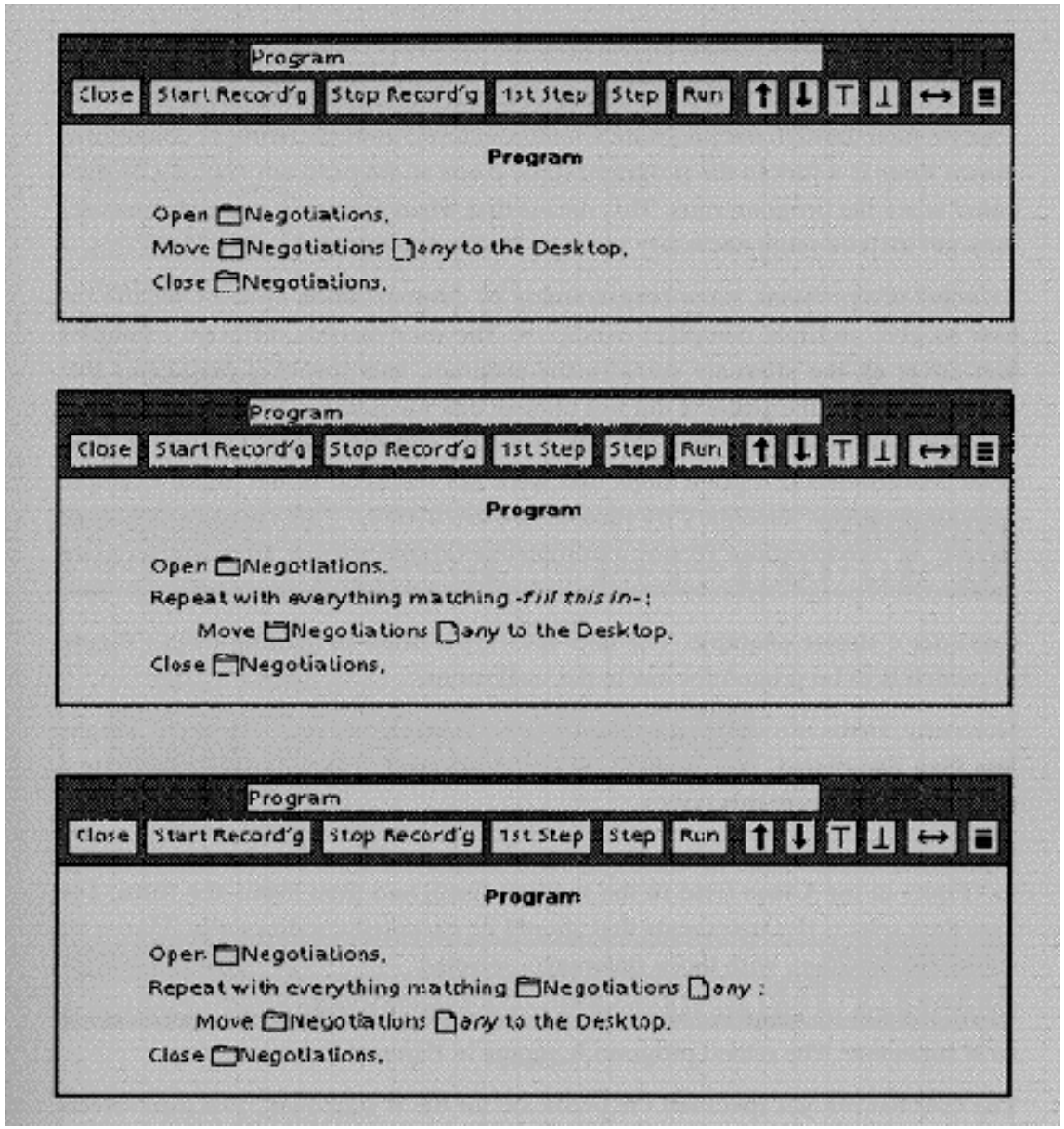
プログラム生成 (Cont'd)



SmallStar [Halbert]

- Star の GUI 操作マクロを例示により生成
- **操作を行なった後で、操作列をエディタで編集**



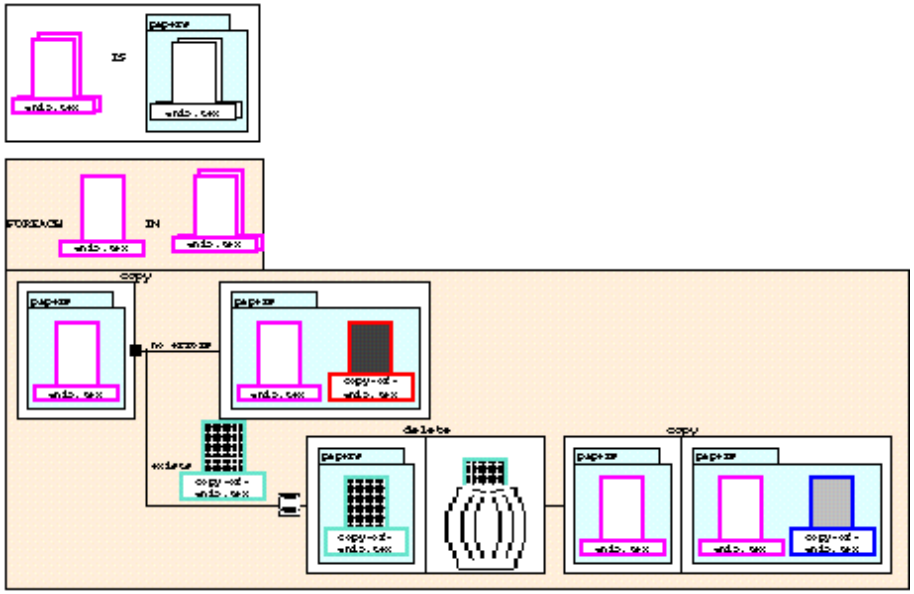

Program
Close
Start Record'g
Stop Record'g
1st Step
Step
Run
Program
Open Negotiations.
Move Negotiations any to the Desktop.
Close Negotiations.
Program
Close
Start Record'g
Stop Record'g
1st Step
Step
Run
Program
Open Negotiations.
Repeat with everything matching -fill this in-:
Move Negotiations any to the Desktop.
Close Negotiations.
Program
Close
Start Record'g
Stop Record'g
1st Step
Step
Run
Program
Open Negotiations.
Repeat with everything matching Negotiations any :
Move Negotiations any to the Desktop.
Close Negotiations.

Edward [Bos]

- GUI の操作手順から次の操作を推論
- 自然言語で操作を指示
- システムが推論結果を自然言語で説明
- ユーザへの質問

Pursuit [Modugno]

- 例示操作から視覚言語プログラムを生成
- それをユーザが直接編集することによりシステムの間違った推論を修正
- 視覚言語の表現を画面上での実際の操作で使われるアイコンに似せている



DemoOffice [杉浦]

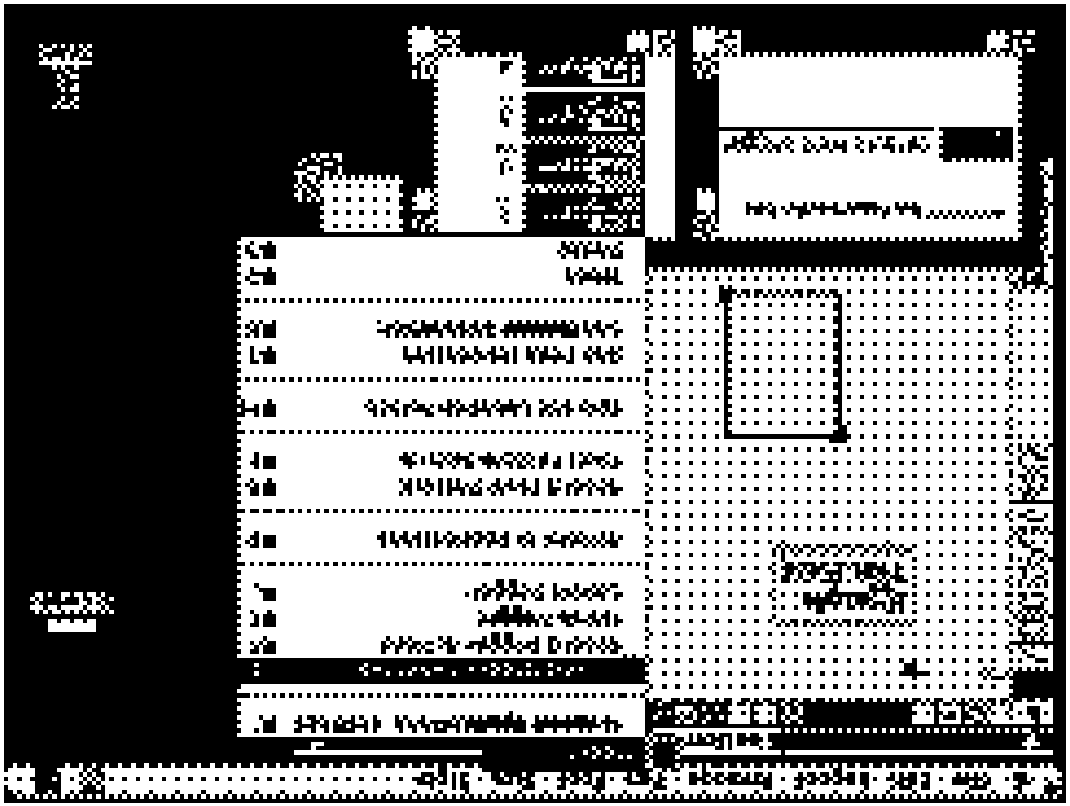
- GUI 操作から再利用可能な部分を抽出し / マクロとして登録
- データ集合 (e.g. メールボックス) の要素 (e.g. メール) に対して操作が行なわれた場合、マクロ自動生成開始
- 操作された要素に関連した操作のみを履歴から抽出 / 汎化してマクロを生成
- 新しい要素を適用可能かどうか試行可能

Internet Scrapbook [杉浦]

- **既存の** Web **ページからの** Copy/Paste **により自分のページを作成**
- **切り貼りのパタンを自動生成**

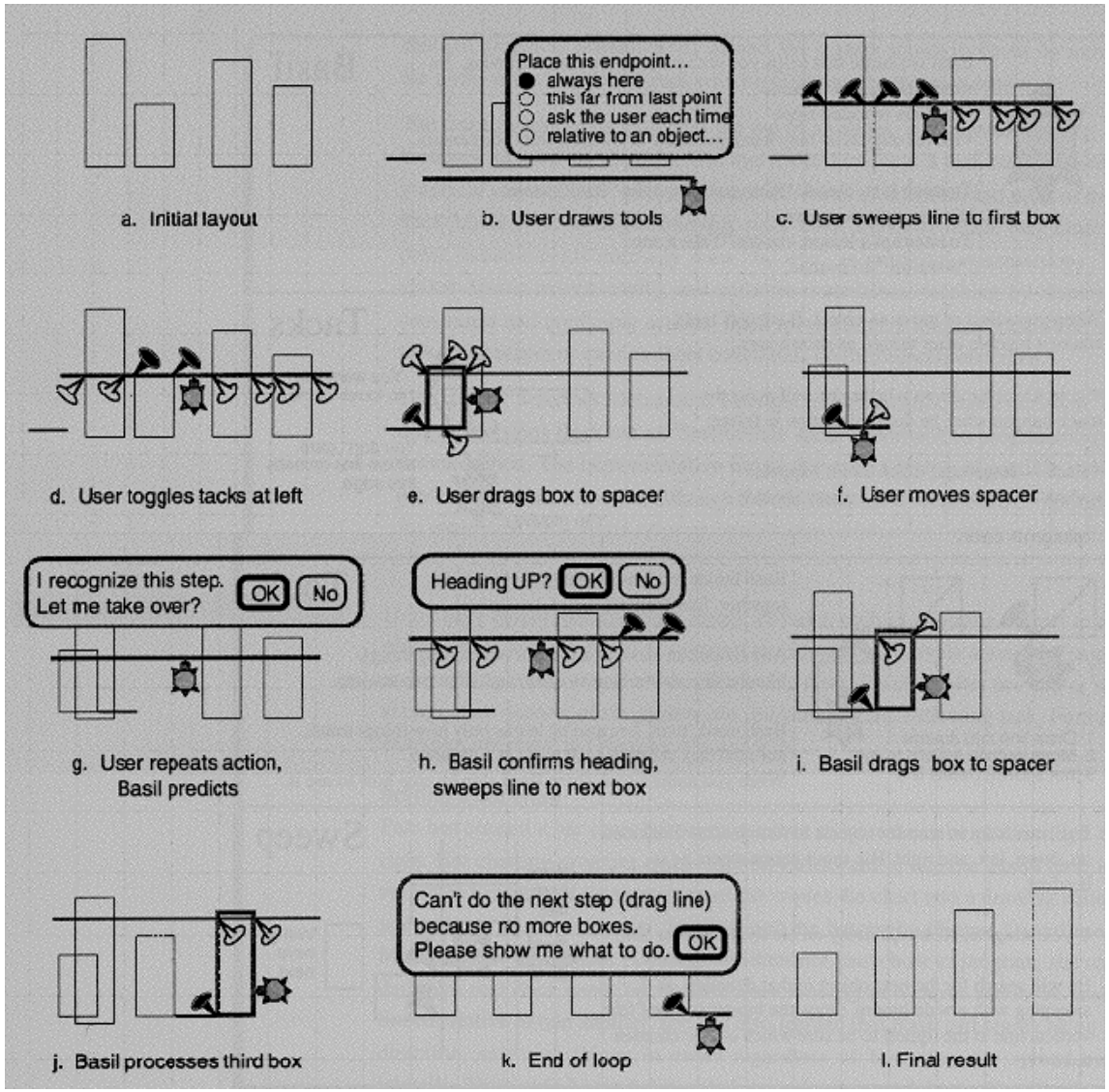
Triggers [Potter]

- ビットマップ画面上のパタンマッチングにより操作の自動実行
  - e.g. 「OK」が表示されたらクリックする
- 定義したビットマップパタンが表示されたとき定義された操作を自動実行させる
- (例) 表計算ソフトの全ての負の数値データに印をつける
  - マイナス記号 “-” を示すビットマップパタンを指定してそのパタンの近辺に印をつける操作を例示により定義



Metamouse [Maulsby]

- 図形の編集操作から編集プログラムを自動生成
- プログラムをプロダクションルールの集合として表現
  - ◇ e.g. 「上に移動させた直線が矩形に接したとき直線の長さだけ矩形を移動させる」
  - ◇ ループも表現可能
- 「接する」「交わる」といった幾何的条件を重視
- システムの推論誤りを修正しながら正しいプログラムを作成
- 単一の例からでも変数や制約を抽出したプログラムを生成
- 似たようだがちょっと違う例があるときは、より詳細な条件をもつ rule が生成され、条件分岐のように働く
- ユーザが以前の操作と完全に同じ操作を始めたときは、すぐに次の行動を予測して提示



a. Initial layout

b. User draws tools

c. User sweeps line to first box

d. User toggles tacks at left

e. User drags box to spacer

f. User moves spacer

g. User repeats action, Basil predicts

h. Basil confirms heading, sweeps line to next box

i. Basil drags box to spacer

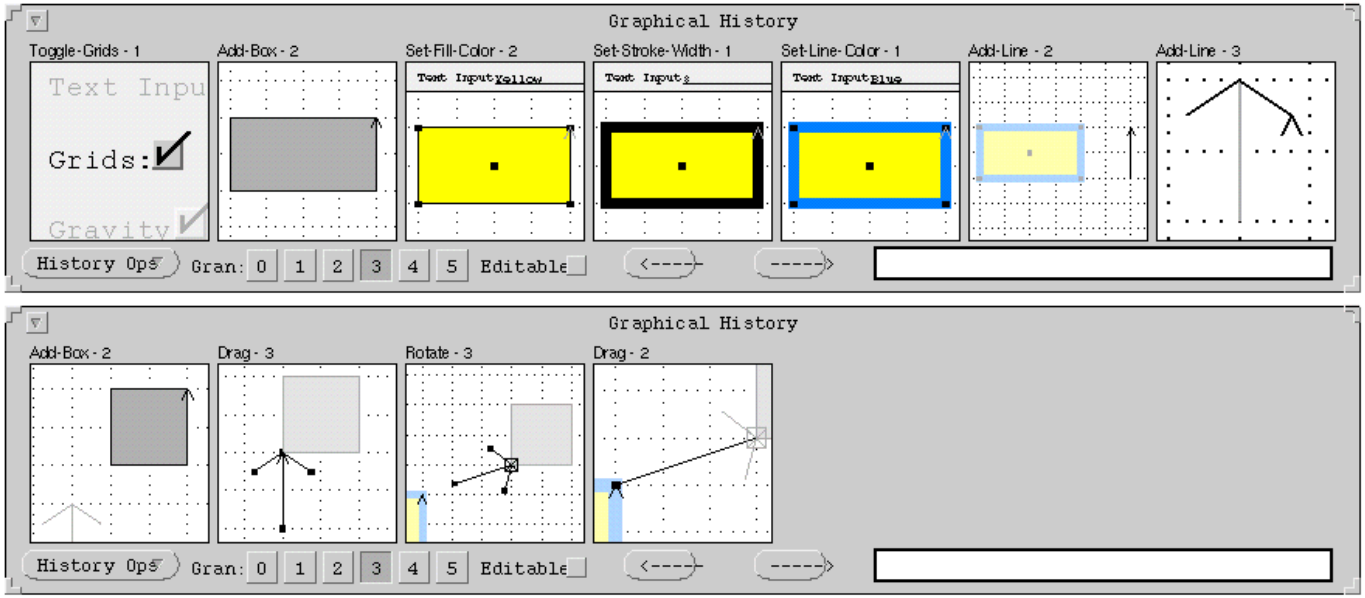
j. Basil processes third box

k. End of loop

l. Final result

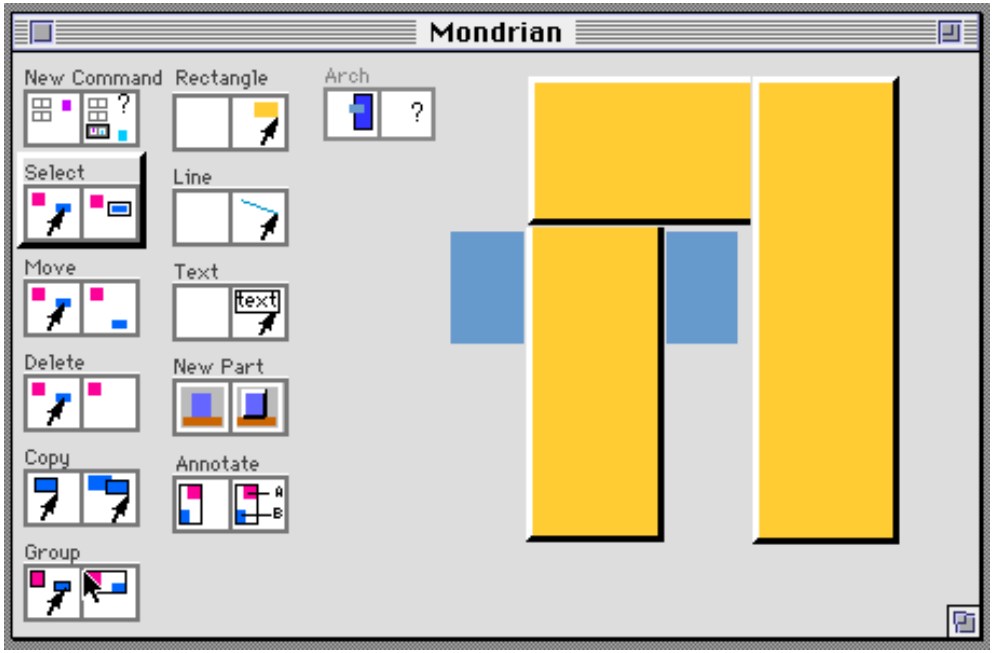
Chimera [Kurlander]

- 図形編集操作の履歴をプログラムとして後で編集 / 定義
- GUI 操作列を紙芝居 (Storyboard) 風に表現
  - ◇ 重要な部分だけアイコンとして表示
  - ◇ 細かい一連の操作はまとめてひとつのアイコンとする
- グラフィカルな形で表現したまま編集可能



Mondrian [Lieberman]

- 図形の編集操作履歴をマクロとして登録
- 操作前と操作後の状態を示すアイコンの組 (Domino) で表現
- 音声 / 自然言語利用



Peridot [Myers]

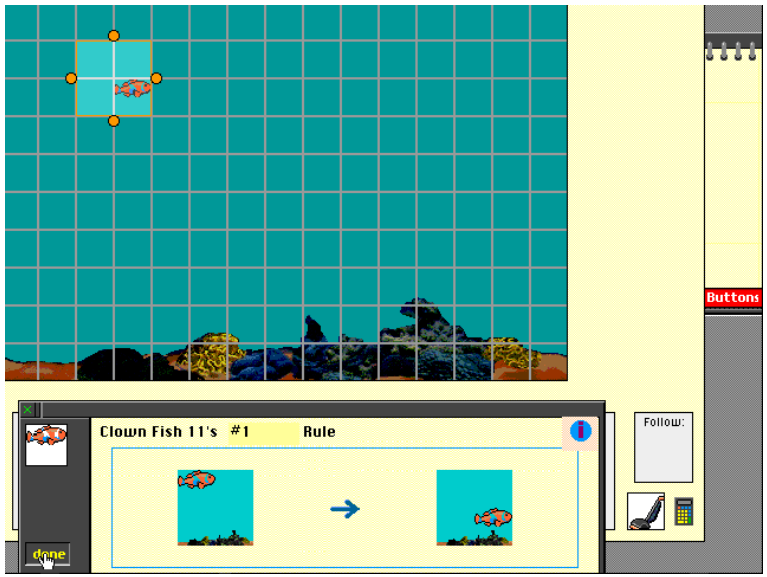
- GUI プログラムを例示により作成
- GUI 操作例からメニューやスクロールバーなどの動きをカスタマイズ

Marquise [Myers]

- GUI プログラミングのほとんどすべての要素を例示のみで実現
- 静的な画面設計 / テストモード (e.g. IB) に加え、動的な反応も例示で定義
- 例示されたアクションから本当のアクションを推定
  - ◇ e.g. Train モードにおいてマウス押下とマウス移動を指定し、 Show モードでその間に点線を引くと、システムは「マウスドラッギングにより点線が移動する」ことを推論
  - ◇ 推論の間違いをパネルで訂正可能
- 編集操作のパラメタ化にすぎないが実用性は高い

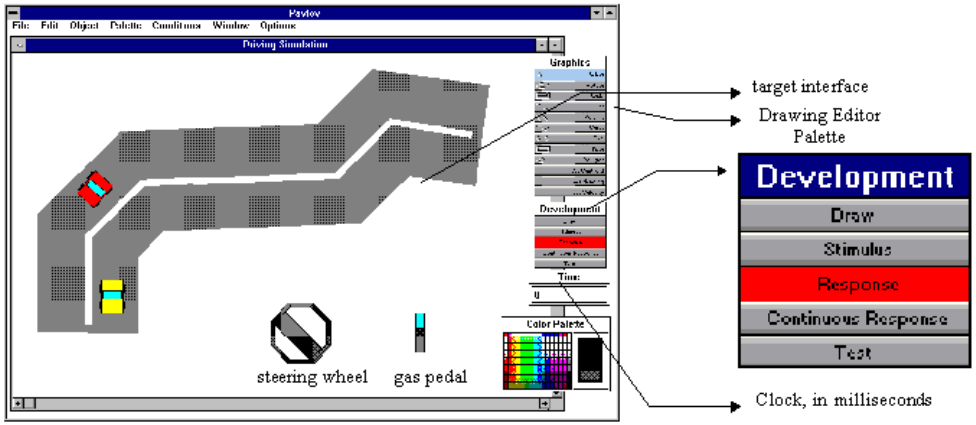
KIDSIM [Cypher]

- (子供でもできる) 例示によるアニメーション生成システム
- 格子の上の物体を動かすことにより物体の移動規則を例示で示す
  - ◇ 移動前と移動後の状態の絵の組が変換規則を示す
- 碁盤の目状の盤面とドミノ状の書き換え規則の集合
- パタンにマッチする規則があればそれに従って盤面が変化
- Cocoa という名前で商品化



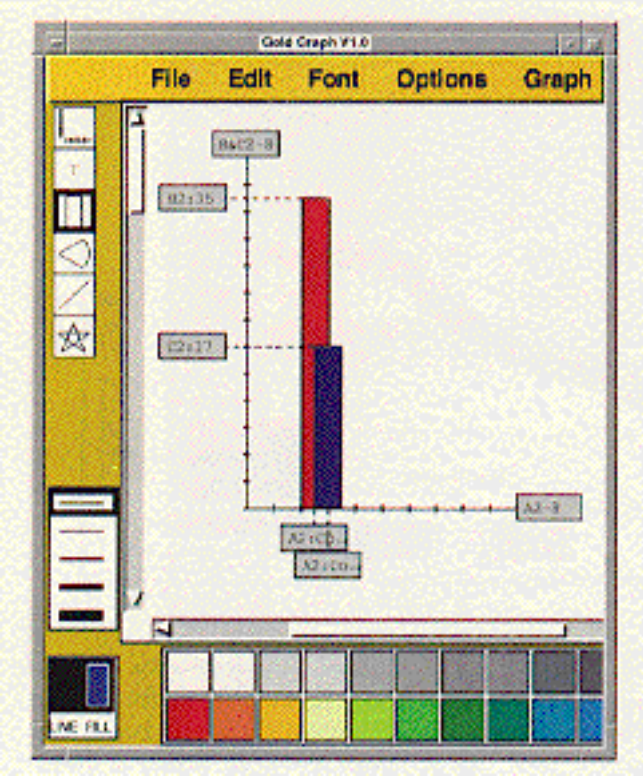
Demo，DemoII, Pavlov [Wolber]

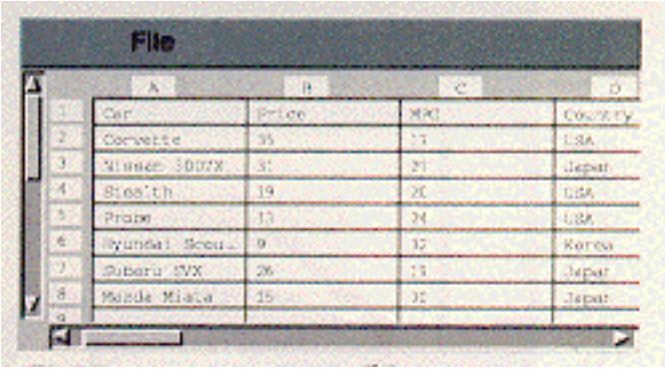
- stimulus(**ユーザの操作**)-response(**システムの反応**) **モデル**
- **例示によりアニメーション / シミュレーションも作成** (Pavlov)
  - ◇ **時間経過やユーザ操作を刺激とする**
- **描画モード / 刺激定義モード / 反応定義モード / テストモードを使用**
- **刺激信号に呼応するオブジェクトの生成 / 変型 / 削除 / 前進 / などの反応を定義**

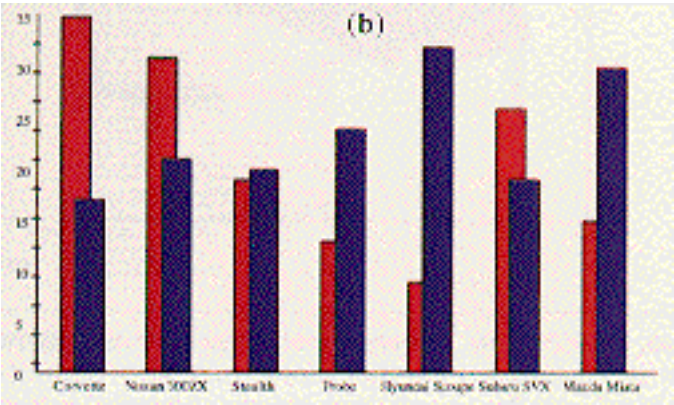


Gold [Myers]

- 例示による表データの視覚化
- データ値と図形の属性との対応を例示により指定

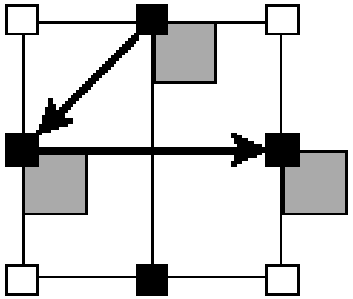




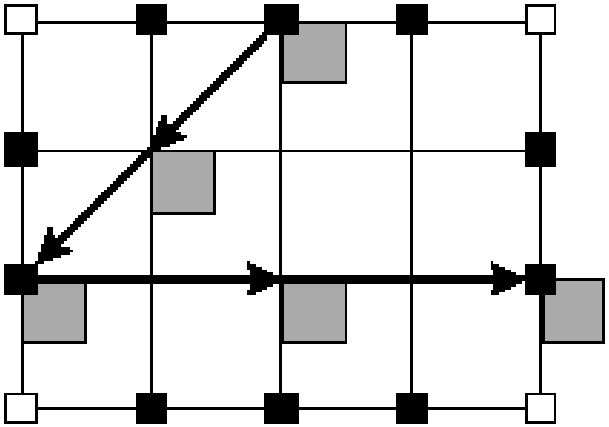


Layout By Example [Hudson]

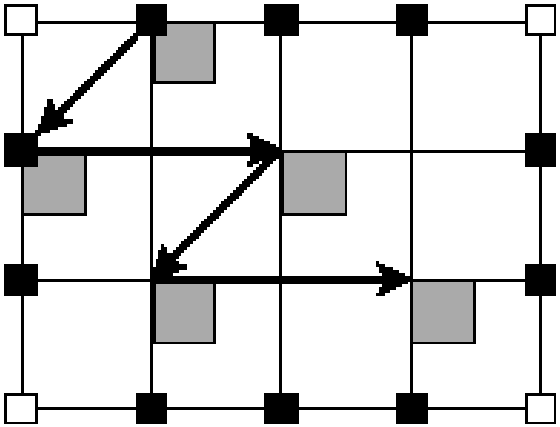
- 図形配置規則を例示により推論
- 図形が 2 個の場合の配置例，3 個の場合の配置例… をユーザが与える
- システムは配置アルゴリズムを推論



(a) 与えられた例



(b) 内挿による拡張



(c) 繰り返しによる拡張

IMAGE [宮下]

- 図形間の制約関係の宣言的記述から図形を自動配置
- 実際の配置の変更により制約の記述に反映
- 複数の配置例から宣言的配置制約を推論
- 推論された配置制約を別の例に適用した結果をユーザに提示

遺伝的プログラミングによるグラフ配置 [増井]

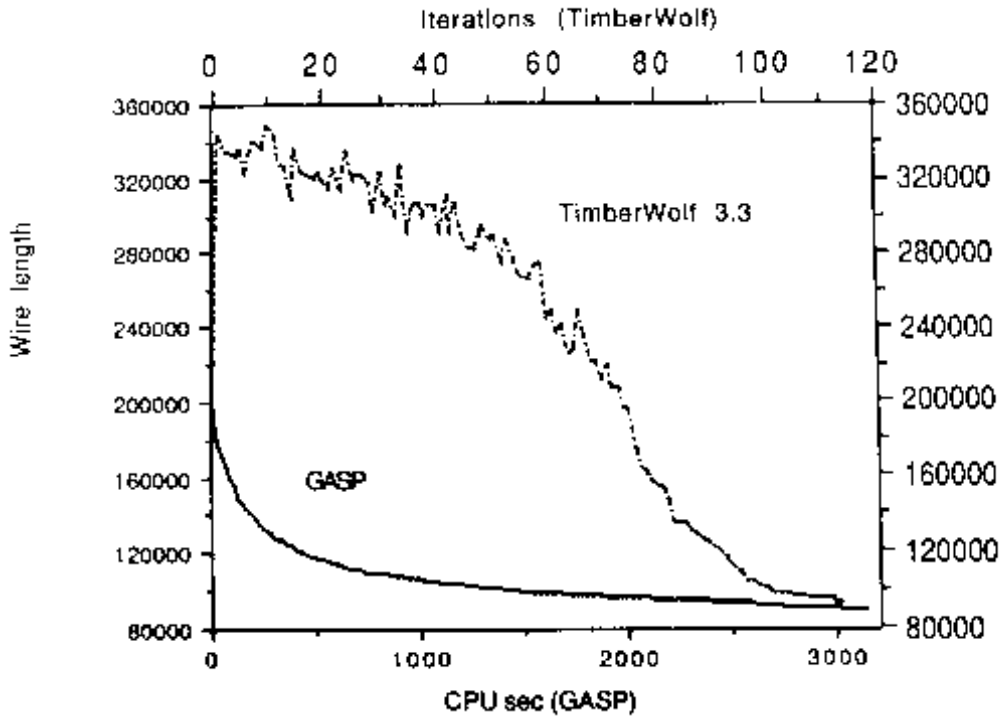
- 確率的手法 (GA, SA など) を用いた図形配置システムでは評価関数の選択が重要
- 配置の好みをユーザが指定することにより、ユーザと同じ判断を行なう評価関数を遺伝的プログラミングで自動生成

確率的手法

- 配置のよしあしを反映する評価関数を最適化することにより徐々に良い解を求める
- 前の解をもとに次の解を計算する
- 手続きを用いずに良い解を得ることが可能
- 代表的なアルゴリズム
  - 遺伝的アルゴリズム (GA)
  - 焼きなまし法 (SA)

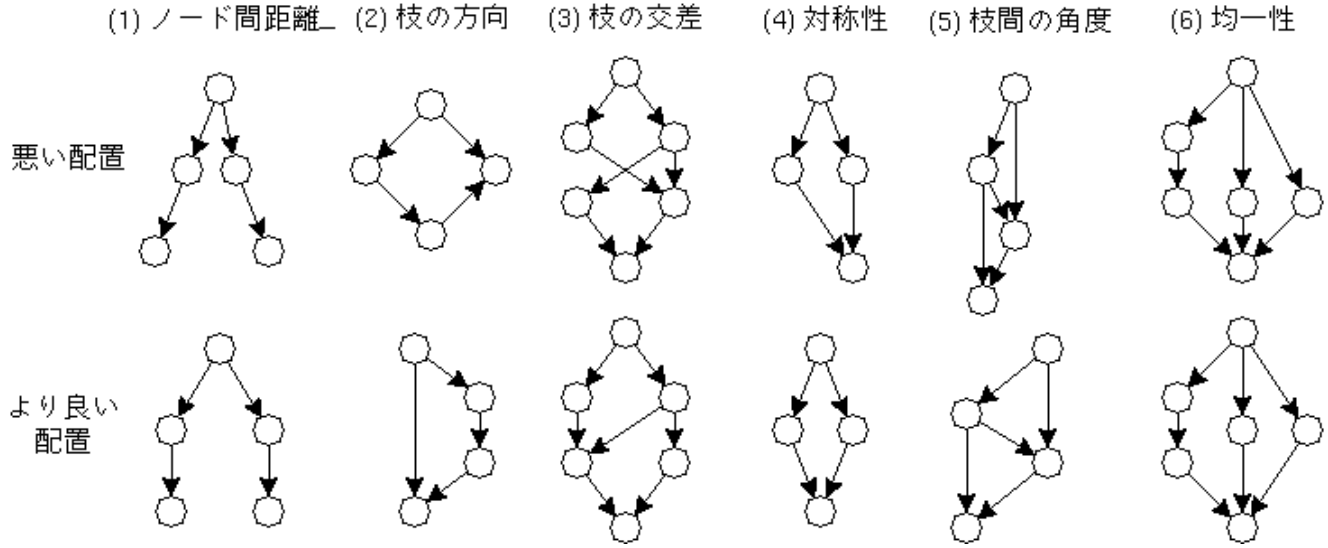
確率的手法の例

- GA(GASP) と SA(TimberWolf) による VLSI 配置



確率的手法のグラフ描画への適用

- **美しさの基準を示す評価関数を最適化する**

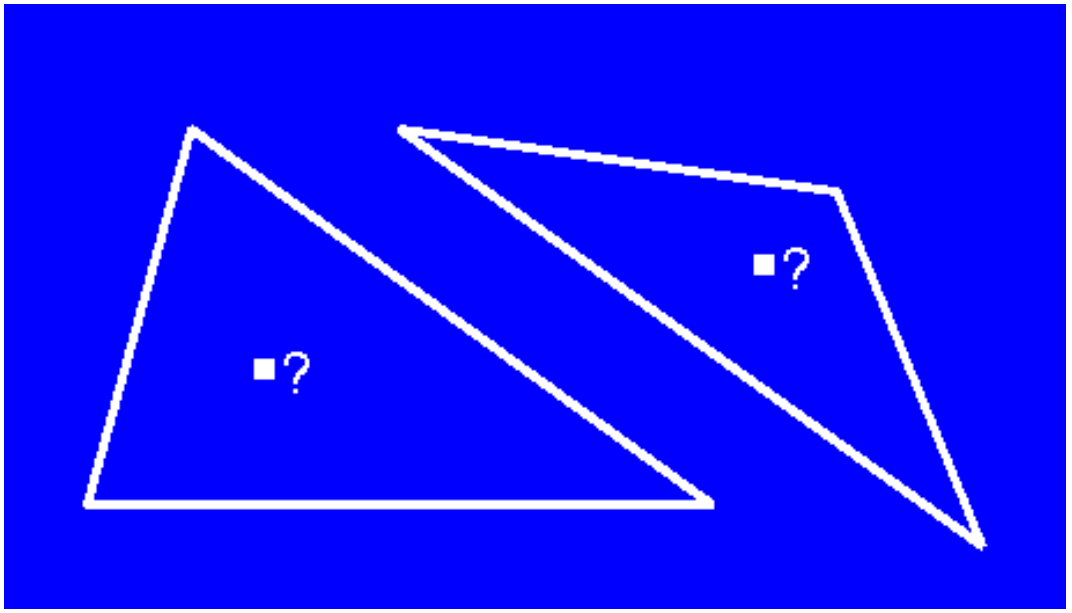


確率的手法の問題点

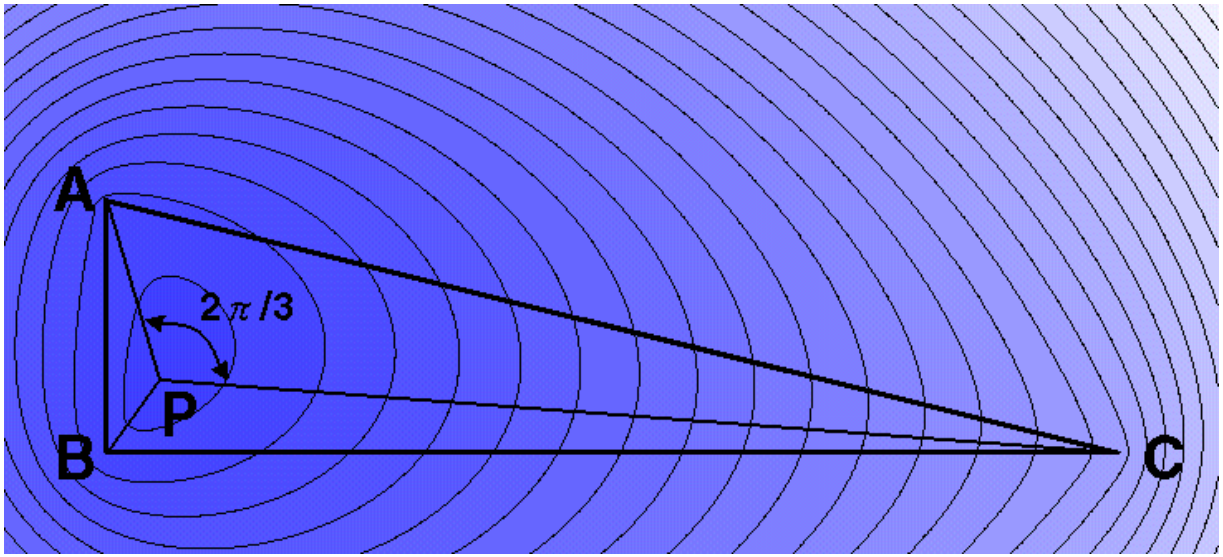
- 遅い
- 最適に近い解が得られるとは限らない
- 評価関数がはっきりしているとは限らない
- 配置制約と評価関数の関係が自明でない

例: 三角形の中の適当な位置に別の点を配置

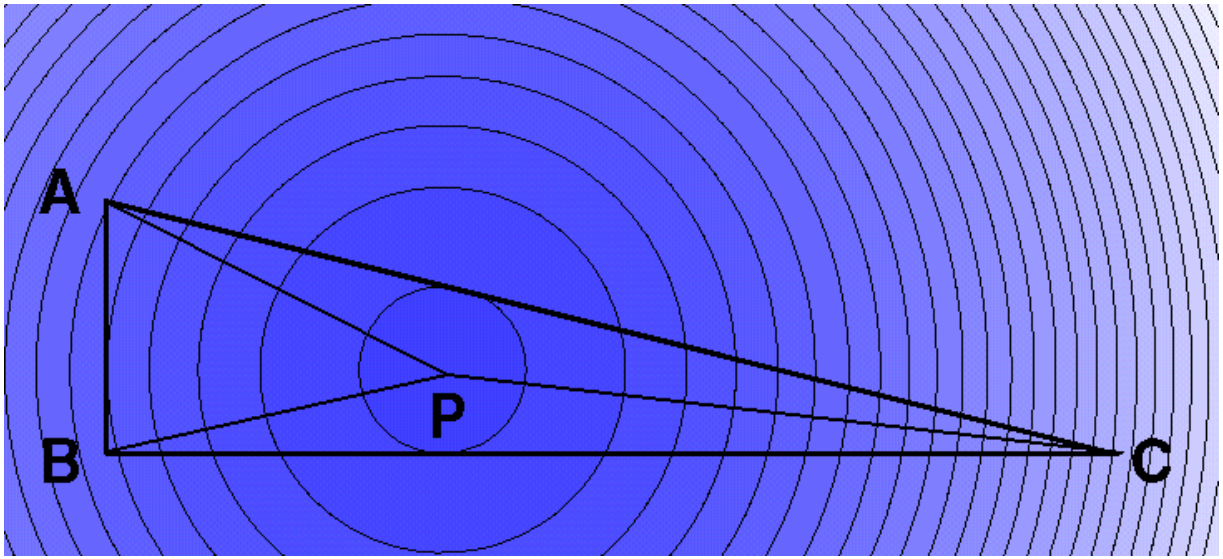
- なんらかの評価関数の最適化によりその点の位置が定まるようにしたい



AP+BP+CP を最小化



$AP^2+BP^2+CP^2$ を最小化



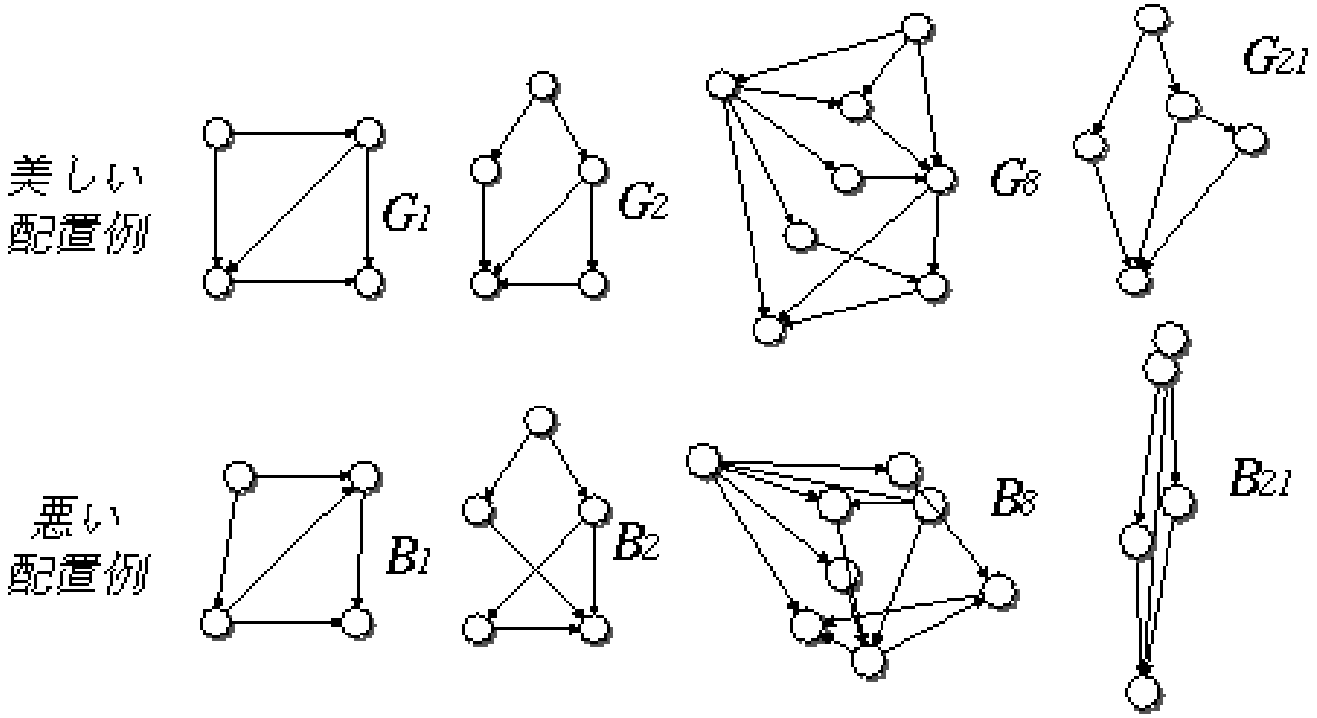
例示によるアプローチ

- 例示により評価関数を作成する
- ユーザはシステムに好みを伝えるだけ
- 遺伝的プログラミングによりユーザの与えた例から評価関数を自動生成

手法

- 良い配置例と悪い配置例の組をシステムに与える
- 遺伝的プログラミングにより配置の良否を決定する評価関数を抽出
- 与えた例における評価基準にもとづいて良い配置を生成することが可能

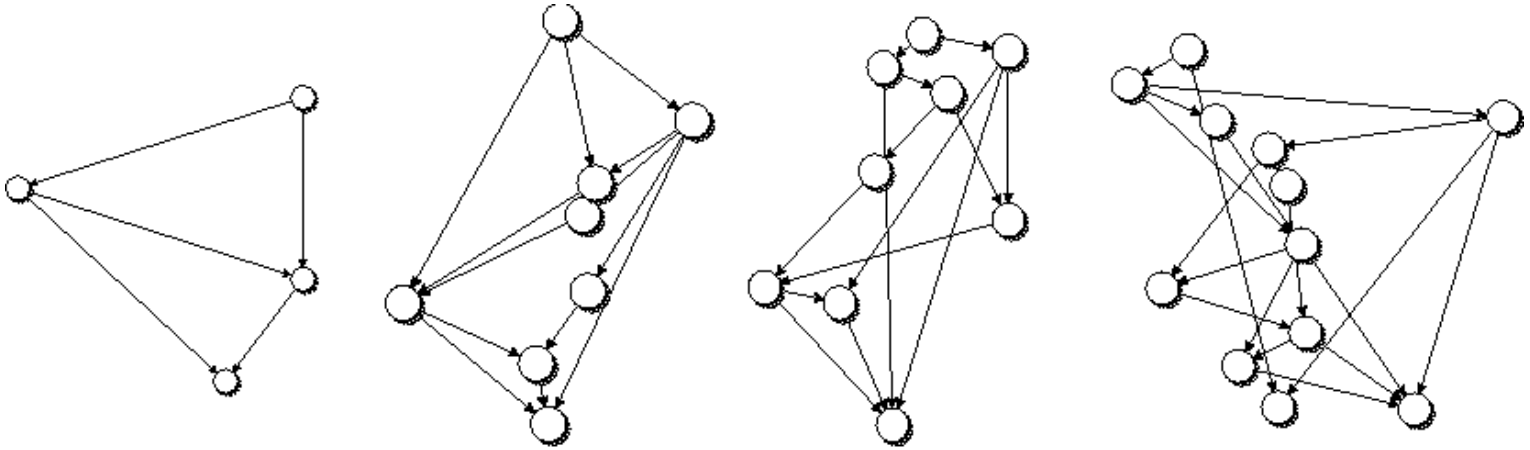
システムに与える配置例



得られた評価関数

(ADD (SUB (ADD (MUL (MUL (MUL (ADD (ADD (ADD SUM(diry) SUM(minangle)) (ADD 44.00 69.00)) (MUL 43.00 MIN(diry))) 5.00) (ADD (ABS MAX(minangle) MIN(dist)) (ADD (ABS 74.00 MIN(dirx)) (ABS 15.00 SUM(locx))))) SUM(minangle)) (MUL 12.00 (CMP (DIV 57.00 MIN(locx)) (CMP 94.00 MIN(intersec))))) (DIV (ABS (MUL (SUB SUM(locy) 27.00) (CMP 28.00 65.00)) 62.00) SUM(dirx))) (CMP (ABS (DIV 67.00 SUM(locy)) (CMP (ABS (ABS 73.00 (CMP 67.00 SUM(dist))) MIN(intersec)) (ABS MIN(dist) MIN(diry)))) MIN(diry)))

得られた評価関数を使って得られた配置



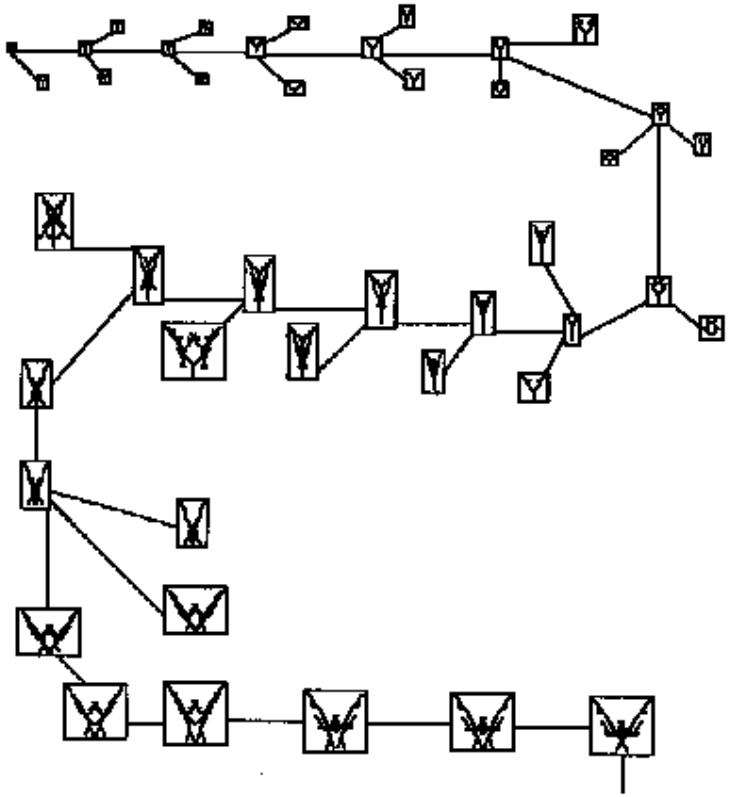
進化的アート作品生成

- **システムに作品をいくつか自動生成させ、良いと思われるものを選択することにより進化させる**
  - Biomorph (Richard Dawkins)
  - Galapagos (Karl Sims)
  - sbart (**畝見**)

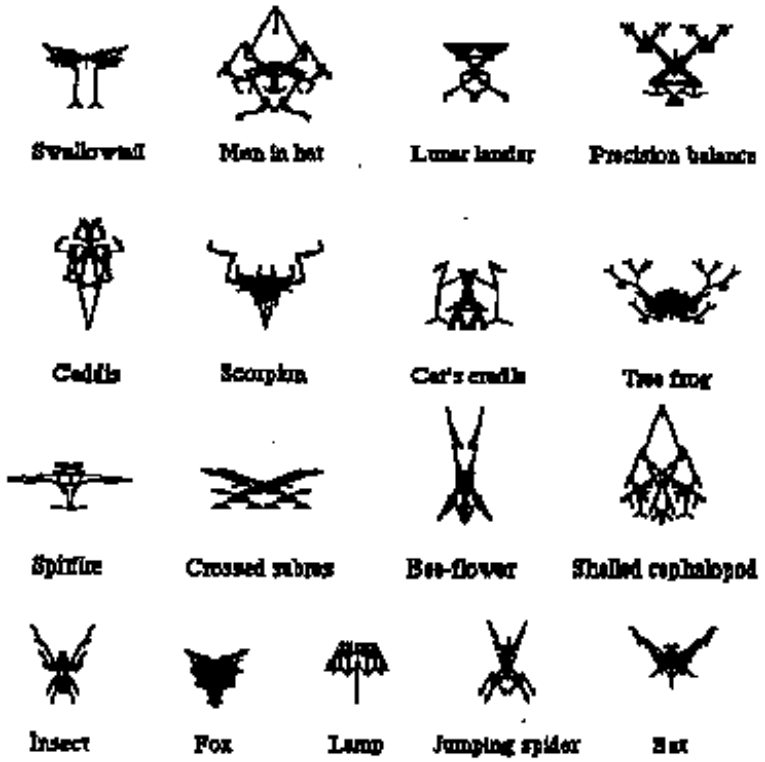
Biomorph

- 「利己的な遺伝子」「Blind Watchmaker」の Richard Dawkins の作品
- 「遺伝子」をもつ「虫」を進化させる
- ランダムに遺伝子を交配させて作った子孫の中から最も気に入ったものを選ぶという行為を繰り返す

Biomorph の進化過程

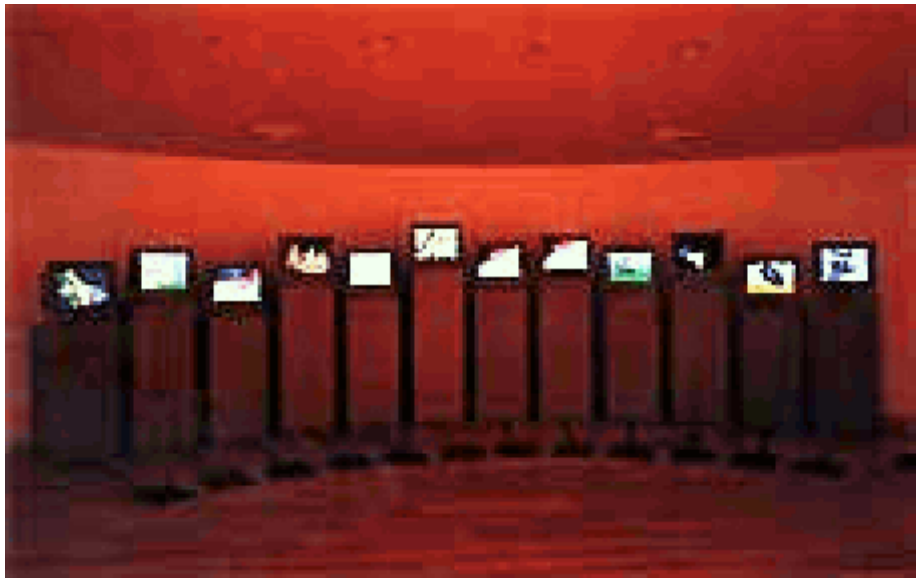


進化結果

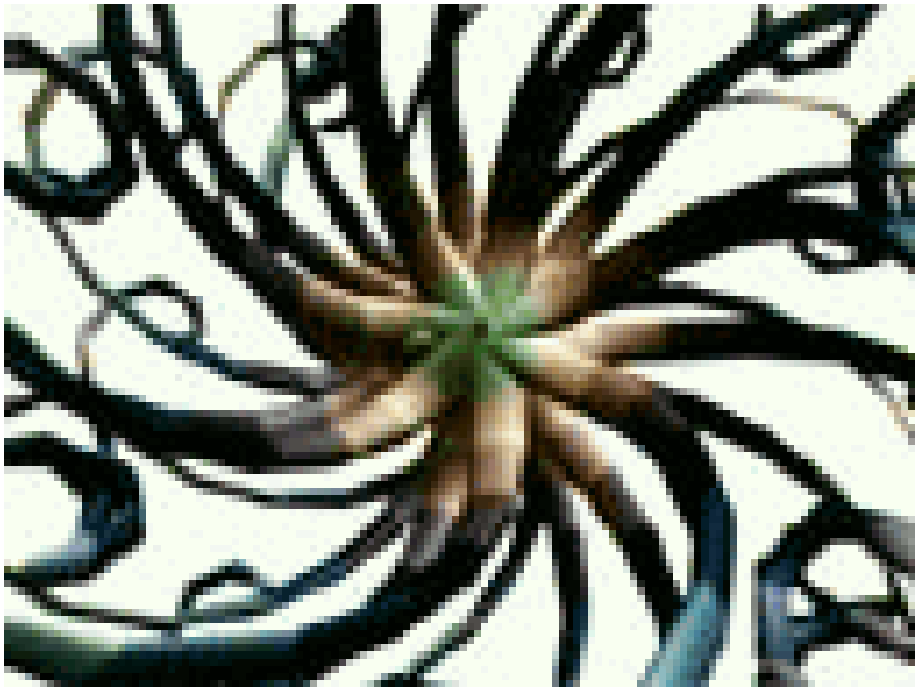

Swallowtail
Man in hat
Lunar lander
Precision balance
Caddis
Scorpion
Cat's cradle
Tree frog
Spitfire
Crossed sabres
Bee-flower
Shelled cephalopod
Insect
Fox
Lamp
Jumping spider
Bat

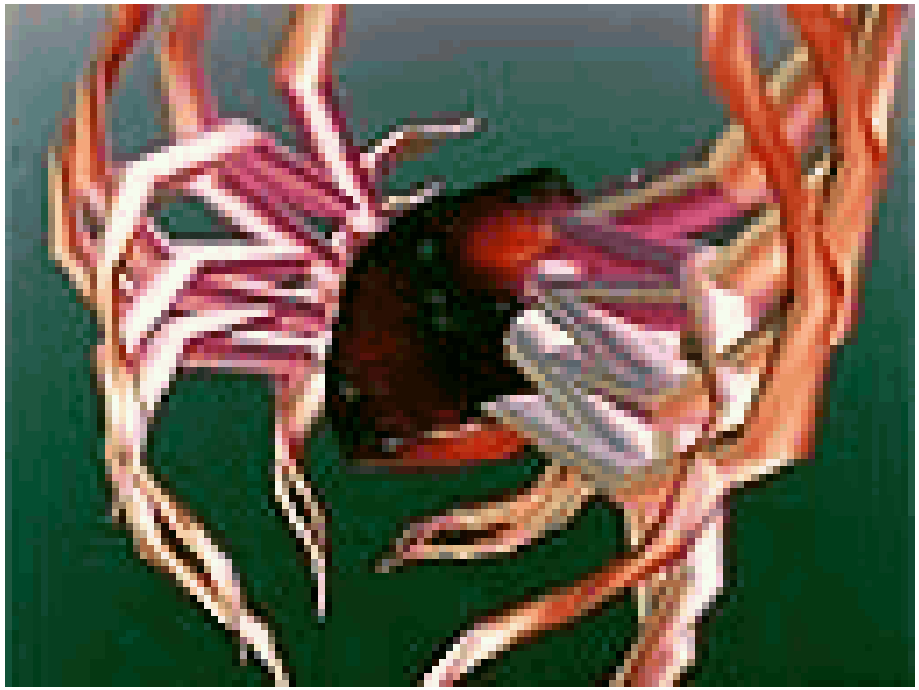
Galapagos [Sims]

- NTT Intercommunication Center(**初台オペラシティ**) **に展示中**
- 9 **個の進化例の中から好みのものを選ぶことを繰り返す**

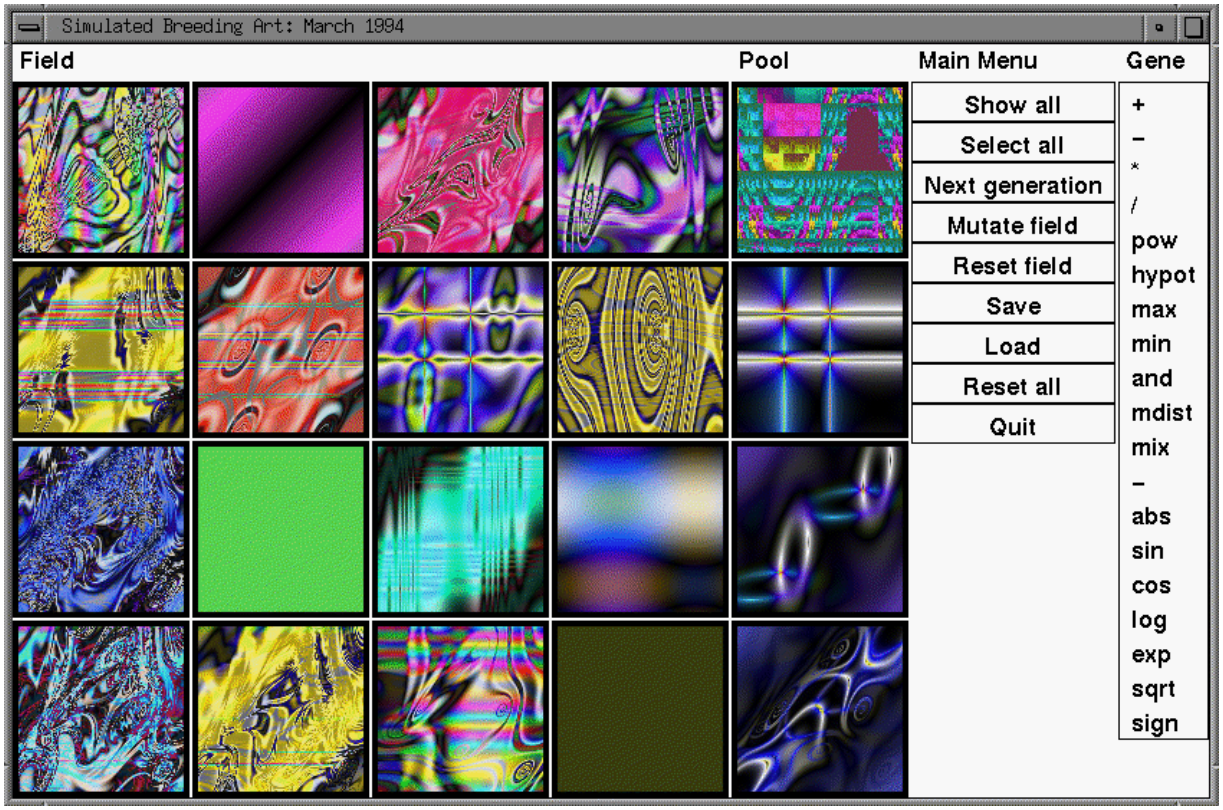


進化した「生物」例





sbart [畝見]


Simulated Breeding Art: March 1994
Field
Pool
Main Menu
Gene
Show all
Select all
Next generation
Mutate field
Reset field
Save
Load
Reset all
Quit
+
−
*
/
pow
hypot
max
min
and
mdist
mix
−
abs
sin
cos
log
exp
sqrt
sign

予測 / 例示インタフェースの現状

- 本当に役にたつシステムは少ない
- 普通にプログラミング / マクロ定義した方が楽なことが多い
- 少ない例からの推論 / 汎化が難しい
  - ◇ ヒューリスティクスの多用
  - ◇ ユーザからのフィードバックによる修正
- システムの推論方式をよく知らないと使えない

予測 / 例示インタフェースの要件

- 予測を使わない場合に比べて全く不都合がないこと
- 予測による効果が余分な労力に見合うだけ大きいこと
- 予測がユーザの期待に一致する確率が高いこと
- 不安感がつきまとわないこと
- 予測機能を使わないユーザの邪魔にならないこと
- 誤った予測を実行したときの被害が小さいこと
- 突然の予測実行 / 挙動変化に驚かされないこと

Brooks による批判

「本書で紹介されている予測 / 例示インタフェースシステムにより自動化できる仕事はプログラムを数行書けば実現できるようなものばかりであり，それにより飛躍的に使い勝手が良くなるようなものではない」

Nardi による批判

「予測 / 例示インタフェースシステムには、

- 終了条件や条件分岐を示すことができない
- 操作が間違いを多く含んでいる場合に困る
- 推論エラーを修正できない
- 推論に高いコストがかる

といった欠点がある」

予測 / 例示インタフェースの展望

- 実世界指向インタフェースにおける応用
- 検索インタフェースとの融合
- テキスト入力 / 描画エディタへの応用

実世界指向インタフェースにおける応用

- 実世界の状況を例データとして活用
- (例) 夜の品川駅で電源を入れたときは時刻表を表示する PDA
  - ◇ プログラミング
    - – センサから得られた位置情報 / 電界強度 / 時刻などの数値から条件分岐
  - ◇ 例にもとづく判断
    - – 一度夜の品川駅で時刻表を使ったことがあるという以前の事例を検索

検索インタフェースとの融合

- **既存データを例として使用 + ユーザによる例データ追加**
- e.g. **仮名漢字変換**
  - ◇ **システム辞書とユーザ辞書が同じ**

POBox

- ペンを用いた文章入力システム
- 読みの部分指定及びコンテクストから次単語を予測

参考文献

- Watch What I Do
- Henry Lieberman のページ
- 筆者のページ
- 2000 年の Communications of the ACM 誌 3 月号で PBE 特集予定

参考文献

1. Apple Computer, Inc. *Welcome to Cocoa – Internet Authoring For Kids*, February 1997.
2. T. C. Bell, J. G. Cleary, and I. H. Witten. *Text Compression*. Prentice Hall, Englewood Cliffs, NJ, 1990.
3. Edwin Bos. Some virtues and limitations of action inferring interfaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'92)*, pp. 79–88. ACM Press, November 1992.
4. Ruven Brooks. Watch what i do - reviewed by ruven brooks. *International Journal of Man-Machine Studies*, December 1993.
5. W. Buxton, M. R. Lamb, D. Sherman, and K. C. Smith. Towards a comprehensive user interface management system. *Proceedings of SIGGRAPH*, Vol. 17, No. 3, pp. 35–42, July 1983.
6. Allen Cypher. Eager: Programming repetitive tasks by example. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'91)*, pp. 33–39. Addison-Wesley, April 1991. also in [7].
7. Allen Cypher, editor. *Watch What I Do – Programming by Demonstration*. The MIT Press, Cambridge, MA 02142, 1993.
8. Allen Cypher and David Canfield Smith. KIDSIM: End user programming of simulations. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'95)*, pp. 27–34. Addison-Wesley, May 1995.
9. John J. Darragh, Ian H. Witten, and Mark L. James. The Reactive Keyboard: A predictive typing aid. *IEEE Computer*, Vol. 23, No. 11, pp. 41–49, November 1990.
10. Gene L. Fisher, Dale E. Busse, and David A. Wolber. Adding rule based reasoning to a demonstrational interface. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'92)*, pp. 89–97. ACM Press, November 1992.
11. George Furnas. New graphical reasoning models for understanding graphical interfaces. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'91)*, pp. 71–78. Addison-Wesley, April 1991.
12. Saul Greenberg and Ian H. Witten. Adaptive personalized interfaces - a question of viability. *Behaviour and Information Technology*, Vol. 4, No. 1, pp. 31–35, 1984.
13. Daniel. C. Halbert. Programming by example. Technical Report OSD-T8402, Xerox Office Systems Division, December 1984.
14. Scott E. Hudson and Chen-Ning Hsi. A synergistic approach to specifying simple number independent layouts by example. In *Proceedings of ACM INTERCHI'93 Conference on Human Factors in Computing Systems (CHI'93)*, pp. 285–292. Addison-Wesley, April 1993.
15. John R. Koza. *Genetic Programming*. The MIT Press, Cambridge, MA, 1992.
16. David Kurlander. *Chimera: Example-Based Graphical Editing*, chapter 12, pp. 270–290. In Cypher [7], May 1993.

17. James A. Landay and Brad A. Myers. Interactive sketching for the early stages of user interface design. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'95)*, pp. 43–50. Addison-Wesley, May 1995.

18. Henry Lieberman. An example-based environment for beginning programmers. *Instructional Science*, Vol. 14, pp. 277–292, 1986.

19. Henry Lieberman. *Mondrian: A Teachable Graphical Editor*, chapter 16, pp. 340–358. In Cypher [7], May 1993.

20. Toshiyuki Masui. Evolutionary learning of graph layout constraints from examples. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'94)*, pp. 103–108. ACM Press, November 1994.

21. Toshiyuki Masui and Ken Nakayama. Repeat and predict – two keys to efficient text editing. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'94)*, pp. 118–123. Addison-Wesley, April 1994.

22. David L. Maulsby and Ian H. Witten. *Metamouse: An Instructible Agent for Programming by Demonstration*, chapter 7, pp. 154–181. In Cypher [7], May 1993.

23. David L. Maulsby, Ian H. Witten, and Kenneth A. Kittlitz. Metamouse: Specifying graphical procedures by example. In *Proceedings of SIGGRAPH'89*, Vol. 23, pp. 127–136, Boston, MA, July 1989.

24. Richard G. McDaniel. Improving communication in programming-by-demonstration. In *CHI'96 Conference Companion*, pp. 55–56. ACM Press, April 1996.

25. Micro Logic Corp., POB 70, Hackensack, NJ 07602. *KeyWatch*, 1990.

26. Ken Miyashita, Satoshi Matsuoka, Shin Takahashi, and Akinori Yonezawa. Interactive generation of graphical user interfaces by multiple visual examples. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'94)*, pp. 85–94. ACM Press, November 1994.

27. Ken Miyashita, Satoshi Matsuoka, Shin Takahashi, Akinori Yonezawa, and Tomihisa Kamada. Declarative programming of graphical interfaces by visual examples. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'92)*, pp. 107–116. ACM Press, November 1992.

28. Dan H. Mo and Ian H. Witten. Learning text editing tasks from examples: a procedural approach. *Behaviour & Information Technology*, Vol. 11, No. 1, pp. 32–45, 1992. also in [7].

29. Francesmary Modugno. *Extending End-User Programming in a Visual Shell With Programming by Demonstration and Graphical Language Techniques*. PhD thesis, Carnegie Mellon University, 1995.

30. Francesmary Modugno and Brad A. Myers. A state-based visual language for a demonstrational visual shell. In *Proceedings of 1994 IEEE Symposium on Visual Languages (VL'94)*, 1994.

31. Brad A. Myers. Creating interaction techniques by demonstration. *IEEE Computer Graphics and Applications*, pp. 51–60, September 1987.

32. Brad A. Myers. *Creating User Interface by Demonstration*. Academic Press, San Diego, 1988.

33. Brad A. Myers. Text formatting by demonstration. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'91)*, pp. 251–256. Addison-Wesley, April 1991.

34. Brad A. Myers. Demonstrational interfaces: A step beyond direct manipulation. *IEEE Computer*, Vol. 25, No. 8, pp. 61–73, August 1992.

35. Brad A. Myers, Jade Goldstein, and Matthew A. Goldberg. Creating charts by demonstration. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'94)*, pp. 106–111. Addison-Wesley, April 1994.

36. Brad A. Myers, Richard G. McDaniel, and David S. Kosbie. Marquise: Creating complete user interfaces by demonstration. In *Proceedings of ACM INTERCHI'93 Conference on Human Factors in Computing Systems (CHI'93)*, pp. 293–300. Addison-Wesley, April 1993.

37. Bonnie A. Nardi. *A Small Matter of Programming*. The MIT Press, Cambridge, MA, 1993.

38. Robert P. Nix. Editing by example. *ACM Transactions on Programming Languages and Systems*, Vol. 7, No. 4, pp. 600–621, October 1985.

39. Richard Potter. *Just-in-Time Programming*, chapter 27, pp. 513–526. In Cypher [7], May 1993.

40. Richard Potter. *TRIGGERS: Guiding Automation with Pixels to Achieve Data Access*, chapter 17, pp. 361–380. In Cypher [7], May 1993.

41. Steven F. Roth, John Kolojejchick, Joe Mattis, and Jade Goldstein. Interactive graphic design using automatic presentation knowledge. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'94)*, pp. 112–117. Addison-Wesley, April 1994.

42. M. Schneider-Hufschmidt, T. Kuhme, and U. Malinowski, editors. *Adaptive User Interface – Principles and Practice*. North-Holland, Amsterdam, 1993.

43. Andrew Sears and Ben Shneiderman. Split menus: Effectively using selection frequency to organize menus. *ACM Transactions on Computer-Human Interaction*, Vol. 1, No. 1, pp. 27–51, March 1994.

44. Gurminder Singh, Chun Hong Kok, and Teng Ye Ngan. Druid: A system for demonstrational rapid user interface development. In *Proceedings of the ACM SIGGRAPH Symposium on User Interface Software and Technology (UIST90)*, pp. 167–177, October 1990.

45. James R. Slagle and Zbigniew Wieckowski. Ideas for intelligent user interface design. In *Tcl'94 Workshop Proceedings*, 1994.

46. David Canfield Smith, Allen Cypher, and Jim Spohrer. KIDSIM: Programming agents without a programming language. *Communications of the ACM*, Vol. 37, No. 7, pp. 55–67, July 1994.

47. Shin Takahashi, Satoshi Matsuoka, Akinori Yonezawa, and Tomihisa Kamada. A general framework for bi-directional translation between abstract and pictorial data. In *Proceedings of the ACM SIGGRAPH and SIGCHI Symposium on User Interface Software and Technology (UIST'91)*, pp. 165–174. ACM Press, November 1991.

48. Ian H. Witten. *A Predictive Calculator*, chapter 3, pp. 66–76. In Cypher [7], 1993.

49. Ian H. Witten and Dan H. Mo. *TELS: Learning Text Editing Tasks from Examples*, chapter 8, pp. 182–203. In Cypher [7], May 1993.

50. David Wolber. Pavlov: Programming by stimulus-response demonstration. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'96)*, pp. 252–259. Addison-Wesley, April 1996.

51. David Wolber and Gene Fisher. A demonstrational technique for developing interfaces with dynamically created objects. In *Proceedings of the ACM SIGGRAPH and SIGCHI Symposium on User Interface Software and Technology (UIST'91)*, pp. 221–230. ACM Press, November 1991.

52. 小島啓二. ビジュアルインタフェースの研究動向と応用, 第 2.9 章, pp. 168–175. 平川, 安村 [55], February 1996.

53. 佐藤雅彦. かな漢字変換システム SKK. *bit*, Vol. 23, No. 5, pp. 793–802, April 1991.

54. 中山健, 宮本健司, 川合慧. コマンド履歴からの動的スクリプト生成. 竹内彰一（編）, インタラクティブシステムとソフトウェア II: 日本ソフトウェア科学会 WISS’94, pp. 155–164. 近代科学社, 1994.

55. 平川正人, 安村通晃（編）. ビジュアルインタフェース – ポスト GUI を目指して. bit 別冊. 共立出版, February 1996.

56. 暦本純一, 長尾確. ポスト GUI: 今後の展望, 第 3 章, pp. 178–198. 平川, 安村 [55], February 1996.

57. 増井俊之. **`keisen.el`** プログラム, July 1991. Mule 配付パッケージに含まれる.

58. 増井俊之. 適応 / 予測型テキスト編集システム. 竹内彰一（編）, インタラクティブシステムとソフトウェア II: 日本ソフトウェア科学会 WISS’94, pp. 145–154. 近代科学社, 1994.

59. 増井俊之. GUI ベースのプログラミング, 第 2.2 章, pp. 45–64. 平川, 安村 [55], February 1996.

60. 松岡聡, 宮下健. 例示による GUI プログラミング, 第 2.4 章, pp. 79–97. 平川, 安村 [55], February 1996.

61. 山本格也. ビットマップに基づくプログラミング言語 visulan. 田中二郎（編）, インタラクティブシステムとソフトウェア III: 日本ソフトウェア科学会 WISS’95, pp. 151–160. 近代科学社, 1995.

62. 宮下健, 松岡聡, 高橋伸, 米澤明憲. 複数の視覚的例による直接インターフェイスの対話的実現. 竹内彰一（編）, インタラクティブシステムとソフトウェア I: 日本ソフトウェア科学会 WISS’93, pp. 241–248. 近代科学社, 1994.

63. 宮本健司. 汎化履歴にもとづく予測インタフェースの拡大. 田中二郎（編）, インタラクティブシステムとソフトウェア III: 日本ソフトウェア科学会 WISS’95, pp. 181–190. 近代科学社, 1995.

64. 原田康徳, 宮本健司. Visible dispatch: Visibility に基づくアプリケーション構築法. 田中二郎（編）, インタラクティブシステムとソフトウェア IV: 日本ソフトウェア科学会 WISS’96, pp. 61–70. 近代科学社, December 1996.

65. 増井俊之. 進化的学習機構を用いたグラフ配置制約の自動抽出. 竹内彰一（編）, インタラクティブシステムとソフトウェア II: 日本ソフトウェア科学会 WISS’94, pp. 195–204. 近代科学社, 1994.

66. 増井俊之. ペンを用いた高速文章入力手法. 田中二郎（編）, インタラクティブシステムとソフトウェア IV: 日本ソフトウェア科学会 WISS’96, pp. 51–60. 近代科学社, December 1996.

67. 増井俊之, 中山健. 操作の繰返しを用いた予測インタフェースの統合. コンピュータソフトウェア, Vol. 11, No. 6, pp. 484–492, November 1994.

68. 杉浦淳, 古関義幸. 例示プログラミングにおけるマクロ定義の簡略化. 田中二郎（編）, インタラクティブシステムとソフトウェア IV: 日本ソフトウェア科学会 WISS’96, pp. 101–110. 近代科学社, December 1996.

69. 萩谷昌己. ビジュアルプログラミングと自動プログラミング. コンピュータソフトウェア, Vol. 8, No. 2, pp. 27–39, March 1991.

70. 井田昌之, 亀井信義. Emacs 解剖学 – 入力の補完. *bit*, Vol. 29, No. 2, pp. 85–95, February 1997.