実世界指向プログラミング

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

概要

Wearable Computing, Ubiquitous Computing などに代表される「実世界指向インタフェース」が注目を集めているが、これを容易に構築するための「実世界指向プログラミング」を提案する。本稿では、実世界指向プログラミングの必要機能、プログラミングテクニック、実際の応用例などについて述べる。

1　実世界指向インタフェースと実世界指向プログラミング

計算機の外側のデータや事物と計算機内データとのギャップを最小にすることによって、計算機を意識することなく透明な存在として活用する実世界指向インタフェース [10][14] が近年注目を集めている。従来のユーザインタフェースの研究では、計算機内のデータを扱うための手法として、比較的限られた入出力装置が対象であったが、実世界指向インタフェースでは実世界のあらゆる事物が計算や操作の対象となり計算機内外のデータを統一的に融合して使えることが必要になっている。例えば、紙の上に式を書くと自動的に計算機内のデータとなり計算結果がその右に写し出されるようなシステムや、外科医が患者の体内を CG で見ながら手術を進めることができるシステムなどが提案されているし、計算機内のデータと印刷されたデータを容易に変換するための技術なども研究されている。

特に今後は携帯計算機や装着型計算機 (Wearable Computer) を誰もが持ち歩くようになるであろうし、実世界のあらゆる場所に計算機を配置して生活を豊かにしようという Ubiquitous Computing が普及するなどしてますます計算機が生活に密着すると考えられるため、日常生活で使われる事物と計算機を融合する実世界指向インタフェース技術は非常に重要になってくると考えられる。

実世界指向インタフェースは実世界の事物を主体として扱うものであるべきにもかかわらず、現状では主体があくまで計算機であり、その入出力装置を工夫したにとどまっているものも多い。たとえば「7 時になればベルが鳴る装置」(目覚し時計) について考えてみる。たとえばこれを UNIX で実現する場合、‘**at**’ コマンドを起動して、サウンドを鳴らすプログラム名とベル音のサウンドファイル名と起動時刻を指定しなければならないが、これらがすべて計算機上のシンボル (名前や数字) を介した間接的な操作や指定にもとづいているため理解しづらい。一方、アナログ目覚し時計を使う場合は、時計を示す文字盤の上で視覚的に連続的に起動時刻を指定できるし、現在時刻を将来に回してみれば未来の実際の動作も確かめることができるので UNIX での場合に比べはるかに直感的であり間違いも少ないと考えられる。このように、実世界の事物そのものを使ってプログラミングを行なうことは、それを示すシンボルなどを使ってプログラミングを行なうことに比べてはるかに容易であることが多い。

現状の実世界指向インタフェースシステムでは、入出力装置として実世界の事物を使っている場合でも、プログラミングは計算機上のテキストエディタで行なっているようなことが多い (e.g. [9])。実世界の事物の処理のプログラミングは実世界の事物のみを使って行なうことができるようにした方がはるかに都合が良いと考えられる。実世界の事物を最大限に活用して実世界指向インタフェースのプログラミングを行なう手法「実世界指向プログラミング」を提案する。

2　実世界指向プログラミングの特長

実世界指向プログラミングは以下のような特長を持っている。

プログラムと操作対象の整合性

従来のプログラミング言語は、計算機内のテキストとして表現されており、プログラムで扱う対象も計算機内のテキストや数であることがほとんどであった。このように、プログラムとその扱う対象の形式が近い場合は問題が少ないと考えられるが、形式が大きく異なる場合は整合性の問題が生じてしまう。例えばテキストベースのプログラミング言語で青い正方形を描く場合、

```
glColor3f(0.0,0.0,1.0);
glRectf(0,0,100,100);
```

などとプログラムしなければならないが、青い矩形と上記のテキストとのギャップは非常に大きい。一方グラフィカ

Real-World Programming.
Toshiyuki Masui.
Sony Computer Science Laboratories, Inc.

ルエディタを使って正方形を描く操作をそのままプログラムとすることができれば理解がはるかに容易である。

ビジュアルな GUI のプログラミングシステムとして、いわゆる「インタフェースビルダ」が近年よく使われているが、この場合にはプログラムの表現と操作対象が近いために理解が容易になっている。ビジュアルな対象を扱うためにはビジュアルなプログラミングが適しているということになる。

同様に、実世界の対象を扱う実世界インタフェースのプログラミングを行なうには対象となる実世界の事物を使うことが最も適していると考えられる。

具体性

実世界の事物をプログラミング要素として扱う場合、具体的な指定を行なうことが容易である。たとえば「ビデオからテレビに画像を転送する」という処理を指定する場合、テキストベースの計算機を使う場合は、あらかじめ各 AV 機器に名前をつけたうえで

```
% videocopy video1 TV1
```

などと指定しなければならないであろうが、

- ビデオの名前が "**video1**" であること
- テレビの名前が "**TV1**" であること
- ビデオ信号を転送するためのコマンドが "**videocopy**" であること

など多くのことを記憶して正しく指定する必要がある。実際にビデオからテレビにケーブルをつなぐ場合のようにこれらの機器を実際に使ってプログラミングを行なう場合にはこのような困難は発生しない。

例示プログラミングとの整合性

実世界の事物を使って操作を指定する場合、具体的な処理は指定しやすいが、変数を含む処理や繰り返し / 条件分岐などを含む処理は指定しにくいという問題がある。しかし例示プログラミング (PBE, Programming by Example)[4] の手法を用いれば、ある程度の抽象度をもつプログラムを自動的に生成することも可能である。たとえば何度もビデオからテレビに信号を送る操作を繰り返していた場合は、信号元としてビデオを指定した時点で送出先をシステムが推論することができるだろうし、毎日同じ操作をしていたような場合はその操作を自動実行させてしまうことも可能であろう。

シンボルとして表現しにくい対象の指定

たとえば携帯端末において「駅のそばで電源を入れると時刻表を表示する」という機能をプログラムしたい場合、テキストベースのプログラムを使う場合は緯度 / 経度や電界強度などを数値化して比較するプログラムを書く必要があり指定が困難であるが、地図の上で領域を指定したり実際に駅に行って電界強度を計測しながらプログラムを書けば簡単に指定することができる。このように、他の表現手法が使いにくいものでも、実物またはその代理を使うことにより容易にプログラム中で用いることができる場合がある。

計算機で音楽を扱おうとすると、音のリアルタイム性や並列性のためにテキスト言語でプログラミングすることは非常に困難であったが、ビジュアルなプログラミングシステム "MAX" の登場後は計算機に詳しくない音学家でも簡単に計算機音楽のプログラミングを行なうことができるようになった。これと同様に、適当な実世界指向プログラミングシステムの使用により実世界インタフェースを誰もが簡単に構築できるようになると考えられる。

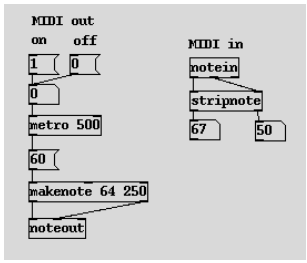

図 1: PD (MAX の後継システム) プログラム例

3 実世界指向プログラミングの例

実世界指向プログラミングにより作成したいプログラムの例を以下に示す。

- 7 時になったらベルが鳴る
- 電話をとると TV が消える
- アメリカに行くと時計のタイムゾーンが切りかわる
- 夜に駅に行くと終電時刻が表示される
- 印刷された文書を机に置くとファイルがあらわれる
- 4 時までに宅急便が来れば隣の家に持って行くよう伝える
- 長野の山が白くなったらスキー情報をダウンロードする[1]
- 郵便局の近くに来ると手紙を出すことを思い出させる
- 誕生日が近づくとプレゼントを思いだす
- 重要なメールが来ると「至急！」タグつきで印刷される
- 壁のボタンを押すと赤外線が出てテレビがつく
- 図書データを本から吸いあげる
- 雨がふると物干竿にメールが届いて洗濯物を取り入れる
- ルーチンワークを行なうとマクロが自動登録されてカードに印刷される

[1] WebCam 画像をプログラムで扱うことができればよい。

- 「コピー棒」を近づけると同じ文書が印刷される
- 机の上に未知の物を置くと内容が教えてもらえる[2]

4　実世界指向プログラミングの手法

4.1　適切なイディオム

各種のユーザインタフェースが普及した背景には、時間をかけて「イディオム」[3]が共通知識化されてきたという経緯があると思われる。テキストベースの端末においては「リターンキーを押すとそれまでに入力した文字列がコマンド及び引数として解釈される」というイディオムが定着しているし、グラフィカルインタフェースでは「盛りあがったように見える矩形上にマウスカーソルを移動してからマウスをクリックするとコマンドが実行される」とか「へこんで見える矩形上にマウスカーソルを移動してからマウスをクリックするとその項目が選択される」といったイディオムが浸透していると思われる。実世界インタフェースではまだこのようなイディオムがあまり存在しないが、操作しやすく理解しやすいイディオムを見出だす必要があるだろう。

たとえば、適当な物体を媒介して実世界の物体間でデータを移動するという実世界指向インタフェースが提案されているが [8][7]、データが媒介物上に存在する場合とそうでない場合を見ただけでは区別することができないため、データが実際にその物体に添付されているという実感をユーザに持たせることがむずかしいという問題があるようであるし、媒介物をなくしてしまった場合などにどうなるかという実感がわきにくいと思われる。直感的で誤った操作をしにくいイディオムが必要であろう。

また、バーコードを認識するとコマンドが起動するという実世界インタフェースもよくみられるが、これはボタンのみで構成された GUI を操作しているようなものであり、あまり高度な処理はできないと考えられるし、「データ削除」といったコマンドがバーコードに割りあてられている危険もあるとすれば気軽に使うわけにはいかない。バーコードを解釈するときの適切なイディオムが必要であると思われる。

4.2　使用可能なデータ

実世界インタフェースを実現する場合、操作対象や自分の位置などの認識が必要になることが多い。現在、ほとんどの工業製品や本 / 雑誌には JAN コード[4]や ISBN/ISSN を示すバーコードが印刷されているので容易に種類を認識することが可能である。また GPS、 PHS の電界強度、住所、電柱番号、電話番号などから位置を判断することもかなり容易になってきている。

多くのものにはこのような情報が付加されていないため識別することができないが、バーコードやタグを添付することにより位置や種類などの意味を関連づけることが可能である。

4.3　プログラム要素

任意のプログラムを作成するためには、通常のプログラミング言語と同様に変数、繰り返しなどの抽象的なプログラミング要素が必要であるが、 3節の例のような単純で具体的な応用がほとんどの場合には以下のような要素の方が重要になるであろう。

実物の代理

プログラム作成時に実物そのものを使えない場合は実物の代理を使う必要がある。たとえば「駅のそば」を指定するためには実際に駅のまわりを歩き回ることもできるだろうが、地図上の領域として指定する方が楽である。同様に、人や時間を指定するかわりに名刺 / 写真や時計を使うことができるだろう。また、数字や「繰り返し操作」といった抽象的な事物もカードなどとして具体化しておけば実世界指向プログラミング中で扱うことができるようになる。

実世界パタンマッチ

主にテキストを扱うプログラミング言語では文字列比較やパタンマッチングが重要であるのと同様に、実世界指向プログラミングでは「夜になると」「駅の近くで」といった実世界的パタンマッチングがしばしば必要となる。このような状況を指定するためには、実際にその時間にプログラミングしたり駅に行ってプログラミングしたりすることもできるし、実物の代理を使ってプログラミングすることもできる。

5　新しい装置の活用

従来は実世界インタフェースの実現のためには高価で大きな装置が必要になることも多かったが、近年はプリンタ / スキャナ / カメラなどの周辺機器が急速に低価格化していることなどにより、各種の安価な装置を組みあわせて実世界インタフェースを構築することがやりやすくなっている。ここではそのような手法をいくつか紹介する。

5.1　実物の代理の動的生成

石井は実世界インタフェースを容易にするために “Phicon” (Physical Icon)[5] の使用を提案している。 Phicon はグラフィカルインタフェースにおけるアイコンを実世界中で使用するものであり、実物の代理などとして使うことができる。

---

[2]国立民族学博物館「ものの広場」でこのようなシステムが使われている。用途がよくわからないアフリカの木製品などに、外から見えないように巧妙に RFID タグが埋めこんであり、特別な机の上に持って行くとタグが認識されて音声やビデオで解説を聞くことができる。

[3]イディオムとは、初めて見聞きしたときには意味がよくわからないこともあるが一旦覚えてしまえばなかなか忘れにくい操作のことをさす。良いインタフェースには優れたイディオムが必要である [3]。

[4]Japanese Article Number。米国で使用されている UPC (Universal Product Code) をもとに JIS で制定され、日本の商品のほとんどに印刷されている。

木やプラスチックなどで Phicon を作る場合は作成が大変であるし、大量の Phicon を扱うことは場所の問題から現実的ではないが、カラープリンタで印刷した紙を折って Phicon を作ることにすれば、動的に Phicon を生成したり Phicon を折り畳んで格納しておいたりすることが簡単にできるようになると考えられる。このような「折紙 **Phicon**」を使うことにより Phicon を使った実世界インタフェースをより手軽に利用することができるだろう。

3 次元物体を使う必要がない場合は ID をもつカードやステッカー [12] を必要に応じて印刷して実物の代理として使うことができる。前述の例のような、実世界の事物間でデータをコピーするような場合は、データをカードやステッカーに変換してから持ち運ぶようにすれば間違いが少なくなるであろう。また、重要なメールや案件を印刷して実物化してカードとして持ち歩いたり、よく使う計算機内データの代理として使ったりすることもできる。最近各種発売されている名刺サイズの携帯プリンタを使えば便利であろう。

図 2: バーコードつきカードの例

5.2 バーコードの活用

バーコードリーダでバーコードを読み取ったときは即座にコマンドとして解釈されるかデータとして解釈されるのが普通である。単純なアプリケーションではそれで充分であるが、汎用の実世界指向プログラミングを行なう場合はバーコードを付加した事物をいろいろな形で活用できた方が都合が良い。

たとえばビデオデッキに付加したバーコードをリーダで認識したときは、そのビデオの各種の操作コマンドを指定したい場合もあるだろうし、信号の転送元 / 転送先として指定したい場合もあるだろう。リモコン情報を吸い上げたいといった場合もあるかもしれない。このような場合は実物をオブジェクトのように扱い、バーコードの認識後なんらかの方法でメソッドを選択できるようにする方が望ましい。たとえば、バーコードをグラフィカルインタフェースにおけるボタンではなくメニューのように扱うことができればよいだろう。

また、バーコードリーダとマウスのようなポインティングデバイスを同時に使えばさらに応用が広がる [11]。リーダで認識した後そのままマウスを移動させればバーコードからの相対位置を追跡することができるから、たとえば地図上の領域をこの装置で指定したりメニューから項目を選択したりできるようになる。

5.3 新しいハードウェア

実世界インタフェースを支援するための新しいハードウェアが有効な場合も多いと思われる。以下にそのような例を示す。

赤外線ボタン

通常の AV 機器のリモコンや電気のスイッチなどの形や位置を変更することはできないが、押すと赤外線信号を発するような小さなボタンを使えば面白い応用が考えられる。

たとえば離れた場所にあるステレオの音量を机の横に貼ったボタンで制御したり、ベッドから電気やラジオを点滅させたりといったことを自由にプログラミングして使えるであろう。

変調蛍光灯

実世界インタフェースでは赤外線 LED などを使って位置を検出することがよく行なわれるが、既存の照明装置からの光に変調をかければ部屋の ID を機器に通知したり、その他の情報をブロードキャストすることができるようになるだろう。

6 実世界指向プログラミングの例

以下のようにして実世界の事物を活用しつつ必要な操作をプログラミングすることが可能である。

6.1 携帯端末のプログラミング

例えば以下のようにして、夜に駅のそばにいれば時刻表を表示するように携帯端末をプログラミングすることができる。

明示的に時刻と場所を指定する場合

実世界パタンに対して携帯端末の操作をマクロ登録できる機能が携帯端末に備わっているものとする。

- マクロ登録機構を起動する。
- 実世界パタンとして「夜」を指定する。「夜」の指定は時計版上の領域を指定することにより行なう。
- 場所パタンとして「駅の近く」を指定する。地図上で駅周辺領域を指定することにより行なう。
- 携帯端末上で駅の時刻表を開く。
- マクロ登録機構を終了する。

このような操作はテキストエディタにおけるキーボードマクロの登録に相当する。

例示により操作を登録する場合

夜に駅のそばで時刻表を開く操作を何度か繰り返したことがある場合は、夜に駅のそばで端末に操作を予測させることにより時刻表を開くことができる。またそれを正式にマクロとして登録することができる。このような操作はテキストエディタにおけるキーボードマクロの自動登録 [6] に相当する。

6.2 ビデオ予約

ビデオを再生することは誰にでもできるがビデオの番組予約 (プログラミング) は難しいと一般に考えられている。予約が難しい理由は沢山あるだろうが、通常の操作と異なる方法で将来の操作を指定しなければならず、確認も簡単ではないことが一因であると思われる。**at** コマンドで目覚し時計を設定したりテキストベースのプログラミグ言語でビデオを予約しようとするのが難しいのと同様である。

目覚し時計と同様に簡単にビデオ予約を行なうためには、予約時にも例えば以下のように通常と同じ具体的な操作を行なう手法が有効と考えられる。

- 時計を進めて将来の録画実行時間にあわせる (将来の実行モード = 予約モードに入る)
- 通常のチャンネル選択ボタンでチャンネルをあわせる
- 録画ボタンを押して録画開始を指示する
- 時計を進めて将来の録画終了時間にあわせる
- 停止ボタンを押して録画停止を指示する
- 時計を現在時刻に戻す (通常モードに復帰する)

このようにすれば、時計を除けば通常と同じ手法で例示風にビデオ操作を指示することができるので操作誤りが軽減されると思われる。また、時計を進める操作を行なうために日付を間違って登録することは少なくなるし、予約動作の確認も容易である。

6.3 AV 機器間データ転送

将来の AV 機器は IEEE1394 などのネットワークにより 1 本の線や無線で簡単に接続できるようになると予想される。また複数の機器を計算機などから集中的に制御することもできるようになると予想される。このとき、機器間でデータ転送を行なおうとするとその指定が非常にわかりにくくなる可能性がある。たとえば 2 台のビデオデッキ間でのデータコピーを計算機から指定するような場合、逆に指定してしまったりすると大変であるが、

```
% videocopy video1 video2
```

などと指定しなければならないとすると、どちらの機械が**video1** だったか / どちらの引数が転送元だったか[5]/ などを覚えておかなければならないので非常に使いづらい。

この場合は、ケーブルでビデオを接続する場合と同様に、ビデオデッキの実物を使用して転送元 / 転送先を指定することにすれば誤ってコピーするような間違いはなくなると予想される。具体的には以下のような操作を行なえばよいだろう。

- 転送元デッキのバーコードをコントローラ端末でスキャンする
- メニューから「転送元」を選択する
- 転送先デッキのバーコードをコントローラ端末でスキャンする
- メニューから「転送先」を選択する
- コントローラから「コピー」を指示する

7 関連研究

実世界指向インタフェースという言葉の定義は人により異なるようであり、 Augmented Reality, Ubiquitous Computing, Wearable Computing など様々な名前で呼ばれているものを含むと考えられるが、 [10] では大きく以下のように分類を行なっている。

- Ubiquitous Computing

  現実世界の沢山の事物に計算能力をもたせる方式

- 識別子認識

  現実世界の事物にバーコードなどの ID をもたせる方式

- 位置認識

  GPS やジャイロなどによる位置認識を使う方式

- オーバレイ

  HMD などを使い実世界画像の上に CG を重ねたり実世界にプロジェクタで計算機画像を投影したりすることにより実世界情報と計算機内情報を融合させる。狭い意味で前者を “Augmented Reality” を呼ぶ研究者も多いようである。 Augmented Reality 全般については Azuma のサーベイ論文 [1] が詳しい。

石井は Augmented Reality を広い意味でとらえ、 “Tangible Bits” という名前で実世界指向インタフェースを捉えている [5]。本稿では、計算機内情報と紙の間でデータを簡単にやりとりするための手法や、計算機がほとんど要らない場合も含めて実世界指向インタフェースとして扱っている。

8 課題

実世界指向プログミングを普及させるために必要な課題は多い。以下にいくつかについて述べる。

[5]UNIX の**cp** コマンドは転送元 / 転送先の順であるが C のライブラリ関数**strcpy()** では転送先 / 転送元の順でなければならない。

8.1 イディオム

4.1節で述べたように、実世界指向インタフェースのイディオムはまだまだ確立していないため優れたイディオムを発明する必要がある。グラフィカルインタフェースシステムでは、ズーミングインタフェース (e.g. [2]) のような優れたイディオムが提案されているにもかかわらず旧来の非連続 / 非可逆のインタフェースが広く使われていることからわかるように、一度広まってしまったイディオムを後で消しさることは非常にむずかしい。実世界インタフェースのための優れたイディオムを早急に開発する必要があるだろう。

8.2 簡単なオーサリング手法

実世界の静的な事物に計算機で解釈を付加すればそれらを動的なメディアのように解釈することが可能である。たとえば絵本の中のバーコードをスキャンすることにより絵本の主人公に言葉を喋らせたり、論文内の写真を認識することにより動画を計算機画面に表示したりさせるすることが可能である。このようにすることによりあらゆる静的な事物を動的にすることができる。たとえばただの紙に様々なマルチメディア情報をリンクして "**Active Paper**" とすることができる。

実世界指向プログラミングの手法を用いて各 ID に動作を割りあてていけばこのような作品を作ることは可能であるが、多数の ID に対していちいち細かい動作を指定しなければならないのでオーサリングに大きな手間がかかる。単純にバーコードをスキャンしてそのログをとるだけでも興味深い結果が得られることもあるので[6]、実世界事物にログをとることにより手軽に日記や旅日記をつけることができると面白いであろう。

8.3 プログラムのブラウジング

実世界指向プログラミングで作成されたプログラムはテキストの形をしていないため通常の方法で印刷したりブラウズしたりすることがむずかしい。インタフェースビルダのようなビジュアルプログラミングシステムで作成されたプログラムが読みにくいのも同様の理由による。実世界指向プログラムをブラウズするためには簡単な拡張現実感システムを用いる必要があるかもしれない。

8.4 時空間の指定方法

実世界パタンとして最も使用頻度が高いと予想されるのは時刻と位置の指定である。前述の例のように時計と地図を使って指定するよりも効果的に時空間を指定する手法が望まれる。地図と時間を 3 次元視覚化するなどの方法が使えるかもしれない。

[6] http://www.icepick.com/ では、ゴミ箱にとりつけたバーコードリーダにより捨てられた全製品の ID を認識することにより生活のすべてのログをとっている。

8.5 単機能化

実際の使用を考えると、「実世界指向プログラミングによりサウンド再生が可能な計算機」よりも、シンプルな目覚し時計の方が使い勝手は良いはずである。実世界指向プログラミングの結果を凍結して、ひとつの機能だけをもつ機械を作成できると便利であろう。たとえば、バーコードをなぞると中身が印刷される「印刷棒」や、なぞると関連ページが表示される「Web 棒」などといった単機能装置が便利かもしれない。

9 おわりに

実世界指向インタフェースは今後ますます広まっていくと予想されるが、実世界指向プログラミングの重要性はあまり注目されていないようである。現在の計算機においてもエンドユーザプログラミングやカスタマイズの容易さが重要であるが、実世界指向インタフェースをもつ装置ではプログラミングやカスタマイズの機能がさらに重要であると考えられるので、実世界指向プログラミングの支援技術を早急に開発することが重要であろう。

参考文献

[1] Ronald T. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, Vol. 6, No. 4, pp. 355–385, August 1997. http://www.cs.unc.edu/~azuma/ARpresence.pdf.

[2] Benjamin B. Bederson and James D. Hollan. Pad++: A zooming graphical interface for exploring alternate interface physics. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'94)*, pp. 17–26. ACM Press, November 1994. ftp://ftp.cs.unm.edu/pub/pad++/pad-uist94.ps.gz.

[3] Alan Cooper. *About Face – The Essentials of User Interface Design*. IDG Books, August 1995. http://www.cooper.com/biblio/bib_about_face.html.

[4] Allen Cypher, editor. *Watch What I Do – Programming by Demonstration*. The MIT Press, Cambridge, MA 02142, 1993. http://www.atg.apple.com/Allen_Cypher/WatchWhatIDo/WatchWhatIDo.html.

[5] Hiroshi Ishii and Brygg Ullmer. Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'97)*, pp. 234–241. Addison-Wesley, April 1997. http://www.acm.org/sigchi/chi97/proceedings/paper/hi.htm.

[6] 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. http://www.csl.sony.co.jp/person/masui/papers/CHI94/CHI94.pdf

[7] Jun Rekimoto. Pick-and-Drop: A direct manipulation technique for multiple computer environments. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'97)*, pp. 31–39. ACM Press, November 1997. http://www.csl.sony.co.jp/person/rekimoto/pickdrop/

[8] Brygg Ullmer, Hiroshi Ishii, and Dylan Glas. mediablocks: Physical containers, transports, and controls for online media. In *SIGGRAPH'98 Proceedings*, pp. 379–386, July 1998. http://www.media.mit.edu/people/ullmer/index.html#papers.

[9] 綾塚祐二, 暦本純一, 松岡聡. Ubiquitouslinks: 実世界環境に埋め込まれたハイパーメディアリンク. 情報処理学会ヒューマンインタフェース研究会研究報告 96-HI-67, Vol. 96, No. 62, pp. 23–30, 1996. http://www-masuda.is.s.u-tokyo.ac.jp/~aya/works/UbiLink-j.html

[10] 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 [13], February 1996.

[11] 椎尾一郎. Scroll Browser: 「壁の中」, 1998. http://fig.ele.eng.tamagawa.ac.jp/~siio/projects/scrollbrowser/indexj.html.

[12] 椎尾一郎, 美馬義亮. Iconsticker: 実世界に取り出した紙アイコン. 安村通晃 (編), インタラクティブシステムとソフトウェア VI: 日本ソフトウェア科学会 WISS'98, p. to appear. 近代科学社, December 1998. http://siio.ele.eng.tamagawa.ac.jp/projects/iconsticker/.

[13] 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.

[14] 暦本純一. 実世界指向インタフェースの研究動向. コンピュータソフトウェア, Vol. 13, No. 3, pp. 4–18, May 1996.