vArashi - Artificial Video Game Player

八重樫 剛史*

2002 年 8 月 30 日

1 はじめに

vArashi プロジェクト [1] は、 Linux Conference 2001 の Lightning Talk において発表した Video Game Playing Computer Project の構想をもとに 2002 年初頭にスタートしたプロジェクトで、ビデオゲームを人間のようにプレイする機械 (vArashi システム) を創造することを目的としている。

本論文では vArashi プロジェクトの概要と沿革について簡単に紹介するとともに、新しく開発した vArashi フレームワークとそれに基づく vArashi システムの構築手法について、ハードウェアとソフトウェアの両面から解説する。また vArashi システムの事例としてシスぷよ (syspuyo) をとりあげる。

本論文内で紹介したソフトウェアのソースコードは vArashi プロジェクトの Web サイト (`http://varashi.jp/`) より入手可能である。

2 vArashi プロジェクトの紹介

2.1 プロジェクトの概要

vArashi プロジェクトとは、現代の高度なテクノロジを無駄に駆使して、ビデオゲームを人間と同様のインタフェースでプレイする機械 (vArashi システム) を創造することを目的としたプロジェクトである。

vArashi システムは、ビデオ出力、オーディオ出力、コントローラ入力といったビデオゲームの入出力信号を扱うための専用装置を持っている。今日のありあまる計算機リソースや、安価に手に入る高性能なデバイスのおかげで、このようなシステムを気軽に構築することができるようになった。

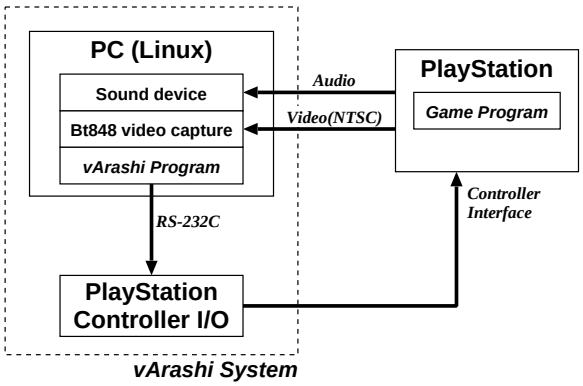
典型的な vArashi システムは、中核部分には一般的な PC を用い、ビデオキャプチャデバイス、オーディオデバイス、ゲーム機のコントローラに入力を行うためのデバイスなどで構成されている (図 1参照)。これにたとえば PlayStation のような家庭用ゲームコンソールを接続すれば、 PlayStation で動作するすべてのゲームを、この vArashi システムに遊ばせる対象として選ぶことができる。

vArashi システムを完成させるためにはゲームをプレイするためのプログラム (vArashi プログラム) が必要になる。 vArashi プログラムはゲーム機から入力された映像信号の処理・画像認識を行い、ゲームの戦略を考え、指示を出すといったことをリアルタイムで行うプログラムである。

このように、 vArashi システムはソフトウェア・ハードウェアの両面において数多くの興味深い技術的トピックを含んでいるといえる。それらを列挙すると以下のようになる。

- ビデオゲーム用画像認識の研究
  - ビデオキャプチャプログラミング

*vArashi プロジェクト email: `t@keshi.org`



(各コンポーネントの解説は第 3.1 節を参照)

図 1: 典型的 vArashi システムの構成

  - マルチメディア命令 (MMX, SSE など) の利用研究
- ゲーム機コントローラインタフェース
  - ハードウェア設計・製作
  - 組み込みソフトウェア開発
  - デバイスドライバ開発
- ビデオゲーム思考ルーチンの研究
- リアルタイムシステムプログラミング
- vArashi 汎用ライブラリ、プログラミングインタフェースの整備

vArashi プロジェクトではこれらのトピックすべてを網羅したわけではないが、本論文では最近構築した新しい vArashi システムを解説することによって、そのいくつかを取りあげる予定である。

2.2 プロジェクトの沿革

- **1998 年 10 月**

  Video4Linux プログラミングについて解説した web ページを作っているときに最初の着想を得る。

- **2001 年 6 月**

  オープンソースのつどい 2001 in 名大 にて妄想を形にする気になる。

- **2001 年 9 月**

  PlayStation コントローラ入力装置が完成。 Linux Conference 2001 の Lightening Talk のセッションにて、“Video Game Playing Computer Projectl” としてアイディアを発表。同時にデモンストレーションを行う。

- **2001 年 12 月**

  このころまでにプロジェクト名称を vArashi プロジェクトと定める。 varashi.jp ドメインを取得し web ページを開設する [1]。同時に紹介記事の執筆も行う [2]。

- **2002 年 1 月**

  デモンストレーション用に新しくキューブ型 PC を購入する。 PlayStation 用ソフト「ぷよぷよ通」をプレイする vArashi システム “syspuyo2” がついに完成するが話にならないほど弱い。 jus 勉強会にてデモンストレーションを行う。

- **2002 年 5 月**

  LinuxWorld 2002 Tokyo .Org パビリオンにてデモンストレーションを行う。

- **2002 年 7 月**

  vArashi システムをビデオゲームエミュレータに適用した vFubuki システムのアイディアを実装する [3]。同時に vArashi フレームワークを考案し新世代 vArashi システム構築に着手。

3 vArashi ハードウェアの解説

3.1 システム構成

図 1 に示した標準的 vArashi システムについて、各コンポーネントの解説を行う。

- **PC (IBM PC-AT 互換機)**

  vArashi システムの核となるコンピュータである。昨今の状況では、性能やコスト、入手性などあらゆる面で x86 互換プロセッサを搭載した PC を選択するのが最善といえる。

  vArashi プロジェクトでは Pentium III-S 1.26GHz、メモリ 512MB の PC と Debian GNU/Linux システムを用いている。筐体には近年流行のキューブ型ケースを用いるなどして小型化をはかっているが、それでも後述する PCI のキャプチャデバイスの使用にこだわっているため、 vArashi システム中で最もかさばるコンポーネントになってしまっている。 PC/104-Plus など組み込み用の製品を探したほうがよいのかもしれない。

- **Bt848 系ビデオキャプチャカード**

  これは Bt848 ビデオキャプチャチップファミリを搭載した PCI ビデオキャプチャカードのことであり、現在最も安価に入手できるビデオキャプチャデバイスのひとつである。このデバイスは安価ながらも非常に高機能で、vArashi プロジェクトの目的には最も適しているといえる。

  Linux においてこの Bt848 系デバイスは bttv ドライバにより利用することができる。 vArashi プロジェクトではデバイスの機能を最大限に生かすために bttv ドライバの拡張を行った (第 3.2 節参照)。

- **PlayStation**

  vArashi システムにプレイさせるゲームマシンである。映像出力を PC に入力する手段があり、入力インタフェース装置を作ることができれば特に PlayStation である必要はないが、入手性やゲームタイトルの多さなどを考慮して PlayStation を選択した。

  最近液晶ディスプレイと一体化した小型バージョンが発売され、外出先でのデモンストレーションがやりやすくなった。

- **PlayStation コントローラ入力装置**

  vArashi システム実現の鍵となるデバイスで任意の種類の PlayStation コントローラをエミュレートすることができる。 PC からは RS-232C でコントロールすることができる。第 3.3 節でこの装置の詳細を説明する。

3.2 bttv ドライバと Video4Linux2 の拡張

bttv ドライバ [4] は Brooktree Bt848 で総称される PCI 用ビデオキャプチャチップファミリのドライバである。単体で NTSC/PAL/SECAM の映像信号をデコードすることができ、安価な PCI ビデオキャプチャカードに多く採用されている。なお現在このチップファミリの主流は Conexant Systems の Fusion 878A という製品になっている。

Bt848 は RISC 命令でプログラミングできる非常強力な DMA コントローラを備えており、PCI バスからアクセスできるメモリ (ビデオカードの VRAM も含む) に対して、キャプチャしたイメージの任意の部分を任意の位置・形式・サイズで直接転送することができる。認識の邪魔となる圧縮処理などを行うこともないので、Bt848 は vArashi システムのような画像処理アプリケーションに最も適したビデオキャプチャデバイスのひとつということができる。

bttv ドライバは初代 Bt848 の時代から広く用いられており、Video4Linux(V4L)[5]、Video4Linux2(V4L2)[6] の両 Linux 標準ビデオキャプチャ API に準拠している。しかしこれらのドライバおよび API は人間が視聴することを主眼に開発されており、画像処理用途で特に必要になる機能がサポートされていない。

そこで vArashi システムで Bt848 の機能を十分に活用できるようにするために、bttv ドライバと V4L2 API の拡張を試みた。この拡張で追加される機能は以下のとおりである。

- **フィールド毎キャプチャ**

  インタレース映像信号の 1 フレームを構成する 2 つのフィールドを連続キャプチャ処理できる。

  V4L2 にはインタレース映像信号の偶数奇数フィールドを区別して扱うための規定が存在し、bttv ドライバもこれに対応しているが、実際には 2 フィールド単位で処理することしかできなかった。そこで 1 フィールドごとの割り込みを処理し、毎フィールド遅延なく処理できるように拡張した。

- **フレームメモリ ioctl**

  フレームメモリとしてユーザプロセス空間の任意の場所を指定できる。

  V4L/V4L2 API では、mmap システムコールによりアプリケーションがビデオキャプチャデバイスのフレームメモリをユーザプロセス空間にマップする方法を規定しているが、ユーザプロセス空間に事前に確保された領域をフレームメモリとして設定する方法はない。

  今回の拡張で `VIDIOC_SETBUF` という新しい ioctl を追加してこれを可能にした。この機能はたとえば、複数のプロセスが共有メモリを用いてキャプチャイメージを共有するようなアプリケーションで、メモリコピーによる性能低下を回避するために有効である。

これらの拡張は V4L2 に準拠する開発バージョン bttv 0.8.45 に対して行った。パッチは vArashi プロジェクトの Web ページから取得可能である。

3.3 PlayStation コントローラ入力装置

vArashi プロジェクトで作成した PlayStation 用コントローラ入力装置は、秋月電子通商で販売されている H8/3664 ワンチップマイコンのキットと、XILINX の CPLD による専用送受信回路で構成されている (図 2, 3 参照)。

PlayStation のコントローラと本体の間の通信プロトコルは [7] に詳しく解説されている。これはいわゆる SPI に似たシリアル通信プロトコルであるが、クロックの間隔が 4μsec (250kHz) 程度しかないことから、マイコンのソフトウェアと汎用入出力ポートのみで対処するのは難しく、このため CPLD で専用の受信回路を作る必要があった。

CPLD の内部には PlayStation からのクロックで駆動されるシフトレジスタがあり、1 バイトの送受信が完了するごとに割り込みを生成するようになっている。

ホスト PC とのインタフェースは H8/3664 に内蔵されたシリアルポートを用いている。この通信プロトコルはとても単純で、2 つのマジックナンバー `0x55`、`0xaa` に続いて PlayStation に読みとらせたいパッドの状態 8 バイトを書きこむだけで利用できる。

図 2: PlayStation コントローラ入力装置の外観

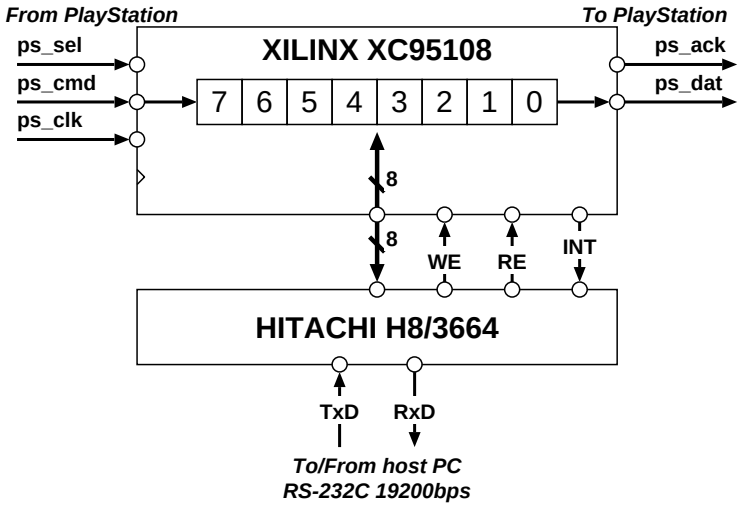


図 3: PlayStation コントローラ入力装置のブロック図

この PlayStation コントローラ入力装置の基板回路図、 CPLD 回路の VHDL ソースコード、 H8/3664 プログラムのソースコードは、 vArashi プロジェクトの web ページより入手可能である。

4 vArashi ソフトウェアの解説

4.1 vArashi フレームワーク

当初 vArashi システムのソフトウェアは、ビデオキャプチャデバイスや PlayStation コントローラなどとの入出力を行うコードと、画像処理やゲームの判断を行う vArashi プログラムのコードが、単一のプログラムとしてリンクされていた。

しかし xmame.vFubuki の開発 (第 4.4 節参照) を契機として、 vArashi システムの構成全体が見直された結果、現在の vArashi システムではこれらのコードをその役割ごとに異なるプロセスに分離し、 vArashi フレームワークと呼ばれるプロセス間通信 (IPC) を用いたサーバ・クライアント構成をとることができるようになっている。つまりサーバプロセスが入出力を行い、クライアントプロセスが vArashi プログラムとして画像認識とゲームプレイを行うのである。

これによって様々なデバイスや vArashi プログラムの組み合わせが簡単に試せるようになり、またネットワークを介して異なるプラットフォームによる混成も可能となった。図 4 に vArashi フレームワークにおけるプロセスと IPC の関係を示す。

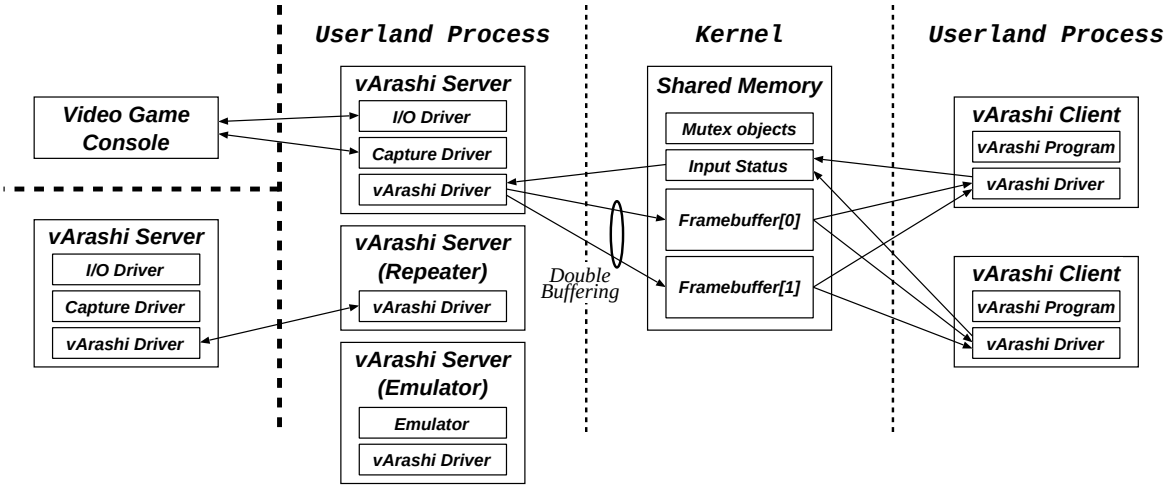


図 4: vArashi フレームワークのプロセス間通信

4.2 ローカル IPC クライアント・サーバ

vArashi フレームワークでは、共有メモリとセマフォのローカル IPC で結ばれた vArashi サーバと vArashi クライアントの 2 種類のプロセスが vArashi システムの基本構成とする。この共有メモリの内容は図 5 のようになっている。

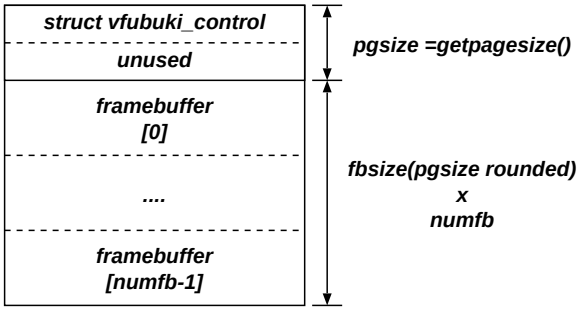


図 5: 共有メモリのメモリマップ

この共有メモリの先頭に配置されている struct vfubuki_control は以下で表される構造体である。

```
#define KEY_MAX         128          /* SDL の keyboard.h の定義 */
#define VFUBUKI_MAXFB   32
struct vfubuki_control {
    int numfb;                       /* フレームバッファの枚数 */
    int fbsize;                      /* フレームバッファのサイズ */
    int width, height, depth;        /* フレームバッファのサイズと色数 */
    int currentfb;                   /* 最後に書き込まれたフレームバッファ */
    int currentseq;                  /* 最後に書き込まれたシーケンス番号 */
    unsigned long fb[VFUBUKI_MAXFB]; /* フレームバッファの位置 */
```

```
    char keystate[KEY_MAX];          /* キー入力状態 (SDL の定義による) */
    char title[32];                  /* ゲームタイトルを表す文字列 */
};
```

vArashi サーバは毎フレーム共有メモリに画像を書きこみ、 vfubuki_control 構造体の内容を更新する。クライアントはそれを読み出して画像処理と判断を行う。 vArashi クライアントが下したゲームの判断結果はキー入力として共有メモリの vfubuki_control 構造体に書きこまれ、それをサーバが読みとってゲームプログラムに伝える。

共有メモリには複数枚のフレームバッファメモリが確保されるため、ダブルバッファリングの技法によりクライアント・サーバとも効率的な同時処理を行うことができる。このとき必要になる同期機構にはセマフォが用いられる。

1 つの vArashi システム内でサーバは 1 つだけであるが、クライアントは 1 つのサーバに対して複数存在することができる。これによって、多人数対戦のゲームにおいてそれぞれのプレイヤごとにプログラムを分離することが可能になる。

現在の実装では共有メモリおよびセマフォには System V IPC を用いている。 System V IPC は現時点では UNIX 間での移植性が最も高いと考えられる。当初 GNU/Linux システムにおいて Posix IPC による実装を試みたが、最近の Linux カーネル (2.4.19)、 glibc(2.2.5) においても必要な API が未だ実装されていなかった [3]。

4.3 TCP/IP クライアント・サーバ

vArashi サーバは前節で述べたようなローカル IPC による通信の他に、 TCP/IP による通信を扱うこともできる。サーバプロセスは TCP 接続要求を受け付けると新しい子プロセスを生成してソケット入出力を行う (図 6参照)。この

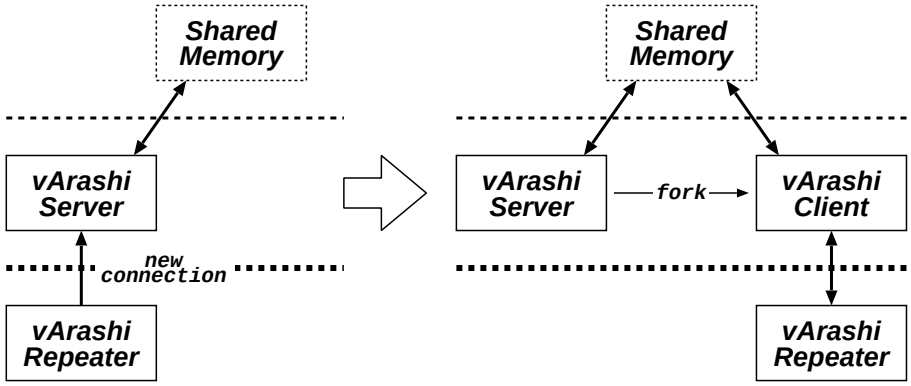


図 6: vArashi サーバに対する TCP 接続

子プロセスは前節で説明したローカル IPC の vArashi クライアントとして振舞い、親プロセスからフレーム情報を受け取りつつソケットに送信することを繰り返す。またクライアントからのキー入力データも随時送信されてくるので、その都度共有メモリに反映する。

この通信で用いるために以下のようなヘッダ構造体が定義されている。

```
/* サーバ→クライアント ヘッダ定義 */
struct vfubuki_tcp_frame {
    unsigned char magic[2];    /* マジックナンバ "@@" */
    unsigned char format[1];   /* フレームのフォーマット (V4L2 の定数) */
    unsigned char depth[1];    /* フレームの色数 (8bit) */
    unsigned char width[4];    /* フレームの幅 (MSB 32bit) */
    unsigned char heigth[4];   /* フレームの高さ (MSB 32bit) */
    unsigned char frameseq[4]; /* フレーム番号 (MSB 32bit) */
    unsigned char fbsize[4];   /* この構造体に続くフレームデータのバイト数 (MSB 32bit) */
};
```

```
/* クライアント→サーバ ヘッダ定義 */
struct vfubuki_tcp_keystate {
    unsigned char magic[2];    /* マジックナンバ "@@" */
    unsigned char code[1];     /* キースキャンコード */
    unsigned char flag[1];     /* フラグ 0=離した 1=押した */
};
```

vArashi クライアントは、この vArashi TCP サーバに対して直接接続を行ってもかまわないが、図 4 にあるような vArashi リピータプロセスを経由したほうがプログラミングが簡単になる。vArashi リピータプロセスは、vArashi TCP クライアントとして振舞うと同時に、vArashi ローカル IPC サーバとしても機能することになる。

この方式をとることにより、vArashi クライアントはローカル IPC 接続用のコードのみでローカル・リモート両方の vArashi サーバに接続することができるようになる。

4.4 xmame.vFubuki

xmame.vFubuki は、著名なアーケードビデオゲームエミュレータ XMAME[8] を改造して vArashi サーバとしたプログラムの名称である。これを用いれば、エミュレータ上で動作するゲームを対象とした vArashi プログラムを書くことができるようになる。

ビデオゲームエミュレータを対象とする vArashi システムはソフトウェアだけで全システムが実現できるため、一般的 vArashi システムにある特殊なインタフェースデバイスの入手性や結果の再現性などの問題がない。また、画像入力などにおけるノイズとも無縁となるため vArashi プログラムの開発も非常に簡単になるという特徴もある。

xmame.vFubuki については、[3] に XMAME の改造手法も含めて詳しく解説されている。

5 vArashi システム事例: シスぷよ (syspuyo)

5.1 シスぷよの概要

本節では、vArashi システムの開発事例としてシスぷよ (syspuyo) を紹介する。シスぷよはコンパイルのビデオゲームぷよぷよシリーズ向け vArashi システムで、vArashi プロジェクトによる最初の開発事例である。このゲームを選択した理由は次のとおりである。

- 知名度が高くルールも簡単
- キャラクタの動きが少なく、決まった位置にのみ出現するので画像認識がやさしい
- 連鎖の組み立てなど、思考ルーチンの奥が深い
- 相手プレイヤーの出方を伺いつつ戦略を変更する必要があり、リアルタイムな駆け引きが面白い

そしてシスぷよ通 (syspuyo2) は、以下のハードウェアとソフトウェアに対するシスぷよの実装である。

- PlayStation 用 ぷよぷよ通 決定盤 (SLPS-00530)
- PlayStation 用 ぷよぷよ通 決定盤 the Best (SLPS-91194)

5.2 シスぷよ通の画像認識

ぷよぷよ通のゲーム画面のレイアウトを図 7 に示す。各プレイヤーのフィールドは 6 列 12 段のセルで構成されており[1]、ゲーム画面内でぷよが表示される場所は基本的にこのセルの中に限られている。また、ぷよの種類には赤青黄緑紫の 5 色のぷよにおじゃまぷよを加えた計 6 種類である。

[1]実際には 12 段目の上に見えない 13 段目が存在する

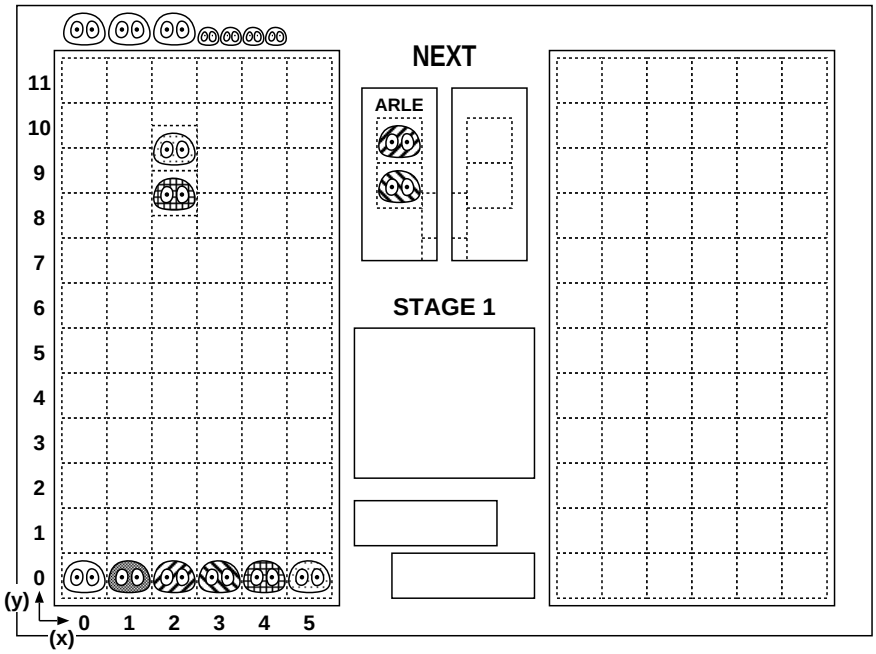


図 7: ぷよぷよ通のゲーム画面レイアウト

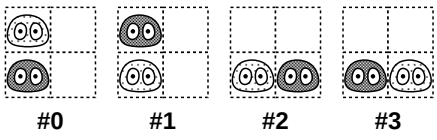
現状のシスぷよ通の画像認識ルーチンは上記の事実を利用した発見的手法に基いており、あらかじめ座標値を調べた各セルに含まれるピクセルの輝度と色相の分布を計算することによって、ぷよがそのセルに存在するか否か、何色のぷよが存在するのかを判定している。

また単一フレームのみを用いた認識を行っているため、ゲーム中随時表われる外乱 (点数や連鎖数や「全消し」などの表示、汗や涙といったような演出) に弱い認識アルゴリズムになってしまっているが、それでも各パラメータをチューニングすることによって、ゲームにほぼ支障のない認識性能を達成している。

5.3 シスぷよ通の思考ルーチン

プレイヤーが操作する組ぷよ (2 つ 1 組のぷよ) は、フィールド座標 (2, 11) のセルから出現する。プレイヤーはこの 2 つのぷよを落下させる x 座標とその姿勢を選択することによってゲームを進行させることになる。便宜上、この一回ごとのプレイヤーの選択を “手” と呼ぶことにする。

組ぷよの取りうる姿勢は図 8 に示す 4 通りが考えられる。フィールドは 6 列あり、右端では #3 と #4 のぷよ配置をとることはできないことを考えると、1 組のぷよに対しプレイヤーが選択することができる手は $5 \times 4 + 2 = 22$ 通りとなる。



※ 左下のセルを基準位置とする

図 8: ぷよの落下姿勢

画面中央の NEXT の領域には次回とその次の回に出現する組ぷよが予告されているため、相手プレイヤーからの妨害が入らない限り、プレイヤーは 3 手先までの結果を正確に予測することができる。

そこでプレイヤーの思考ルーチンとしては、まずフィールドの局面に対してスコアをつける評価関数を作成し、各手ごとに 2 手ないし 3 手先までの全シミュレーション結果に対してその評価関数を適用して、最善手を見つけるという方法が考えられる。

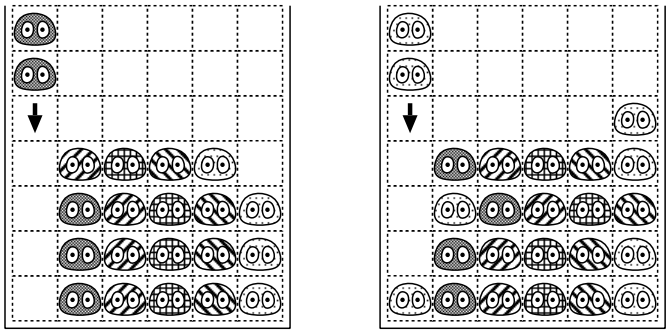
プレイヤーの強さはこの評価関数によって決まるが、現状ではまだ非常に単純な評価関数しか用意できていない。しかしながら、フィールドに積まれたぷよの高さ (y 座標の最大値) を低く保つことを目的とした極端に単純な評価関数を採用して 2 手先全数探索を行った場合でもそれなりの強さを発揮することはできるようである。具体的には、「れんしゅう」モードの「なれた」レベル全 5 ステージをクリアできる程度の強さは実現できているようである。

この評価関数は自フィールドの大部分がぷよで埋まってしまったときに妥当の戦略 (とにかくぷよを消して掘り起こす) にほぼ一致している。 vArashi システムの得意とする正確かつ高速なぷよ操作のおかげで、すぐにピンチに陥いるがなかなか負けず粘り強いという傾向が見られた。

5.4 今後の改良方針

まず画像認識ルーチンを改良する必要がある。テンプレートマッチングなど、より高度なアルゴリズムを適用して外乱に強い認識を実現するとともに、 MMX のようなマルチメディア命令を使用することを検討している。

思考ルーチンについては評価関数の改良が主な課題となる。連鎖が大きく評価されるように重みつけを改めるとともに、「階段積み」などのような連鎖の定石 (図 9 参照) を認識して高い評価を与えるアルゴリズムを考案する必要がある。



(a) 階段積み　　**(b)** 挟み込み

図 9: ぷよぷよで見られる連鎖の定石の例

さらに相手プレイヤーの動向を見つつリアルタイムに戦略を変更する機能を盛りこめば、ぷよ本来の駆け引きが vArashi プログラミングにおいても楽しめるようになるものと考えている。

6 まとめ

本論文では vArashi プロジェクトにおける以下のトピックをとりあげ解説した。

- PlayStation コントローラ入力装置
- Video4Linux2 API と bttv ドライバの改良
- vArashi フレームワークの考案と実装
- シスぷよ通 (syspuyo2) 事例紹介

新しい vArashi フレームワークの開発により vArashi システムを構成するソフトウェアコンポーネントの個別開発がやりやすくなり、また vFubuki システムの考案により容易に vArashi システム実行環境が実現できるようになったことは、特筆すべき成果といえる。

これを足がかりに本来の vArashi プログラムの開発に集中し、さらに vArashi システムの事例を増やし発表していくことができればと考えている。

参考文献

[1] vArashi Project. World Wide Web. http://varashi.jp/.

[2] 八重樫剛史. vArashi Project. De ぶ an BNU/Linux 不徹底入門, 2001. 2001 年冬に発表された同人誌 (売り切れ) http://www.debuan.org/ より PDF 版がダウンロード可能.

[3] 八重樫剛史. vFubuki システム制作日記. De ぶ an BNU/Linux 不徹底入門, 2002. 2002 年夏に発表された同人誌 (売り切れ) http://www.debuan.org/ より PDF 版がダウンロード可能.

[4] Gerd Knorr. bttv. World Wide Web. http://bytesex.org/bttv/.

[5] Alan Cox. Video4Linux. World Wide Web. http://roadrunner.swansea.uk.linux.org/v4l.shtml.

[6] Bill Dirks. Video for Linux Two. World Wide Web. http://www.thedirks.org/v4l2/.

[7] Nifty-ID:HFB03536 藤田. プレイステーション・ PAD/ メモリ・インターフェースの解析.

[8] XMAME. World Wide Web. http://x.mame.net/.