指を弾いて遊べる仮想おはじきゲームの実装と評価

A Tabletop Virtual Marble Game and Its Evaluation

佐藤 俊樹　　福地 健太郎　　小池 英樹*

Summary. We have developed and evaluated a tabletop virtual Ohajiki game using finger flicking gesture recognition system "Ohajiki Interface". Ohajiki is a Japan's traditional marble game. A player flick a flat marble that is called "Ohajiki" on the table with his finger. This system can recongnize user's hand position and it's finger speed using a very rapid image recognition, and enables a user to flick a virtual Ohajiki projected on a table with his finger.

1　はじめに

指を弾く動作とは人差し指や中指等の指を用いて小さな物体を弾き飛ばす動作である（図 1）．この動作は机の上の小石やコインを弾いて遊んだり，埃を弾き飛ばしたりと日常的に行う動作でもあり，また古くからある遊戯としてガラスの玉を指で弾いて遊ぶ「おはじき」が親しまれてきた．

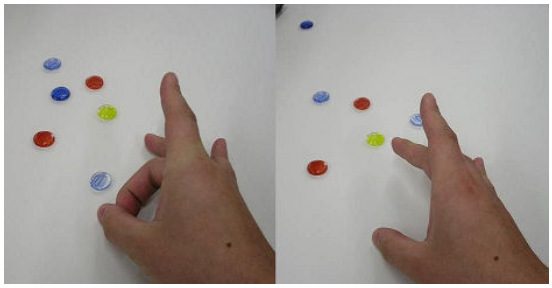

図 1. おはじきの動作はまず人差し指や中指を親指に引っ掛け，力をかける．次に親指にかけた指を外す事で指を弾く．

著者はこれまで，この指を弾く動作を用いたユーザインタフェースである「おはじきインタフェース」を開発してきた．このシステムは指を弾く速さと方向をハイスピードカメラを用いて認識するものでゴルフゲーム等への力の入力インタフェースとして用いる事ができるものである [1][2].

この論文では，新たに指を弾く動作を用いた AR システムとして，机上に投影されたおはじきをユーザが指で弾いて遊ぶ事ができる「テーブルトップおはじきシステム」について述べ，また新しい指の認識手法の提案を行い，それらについての評価実験の結果を示す．

* Toshiki Sato, Kentaro Fukuchi, Hideki Koike 電気通信大学大学院 情報システム学研究科 情報システム運用学専攻

2　指を弾く動作

「指を弾く動作」は力の強さと力の方向の 2 種類の要素から成ると考える事ができる．これらの要素は，それぞれ大まかには指を弾く時の強さと，狙いを定めた方向で定められる．我々はこの 2 種類の情報を得る事により，指で弾く動作を直感的なインタフェースとして利用する試みを行ってきた．

「指を弾く動作」は，実際に指に力を入れることで弾く強さを調節する動作である．また「物を弾き飛ばす動作」である事から，これを「ゴルフ」や「バッティング」等の「ボールを打ち出す」動作があるスポーツに用いると，実際の力を用いて打ち出す強さを制御できる点に加え，「弾く」という共通の感覚があるためより面白くより直感的に操作できると考える．

次に「狙いを付ける」という動作は方向を指示させるためには最も直感的な動作であると言え，ユーザは狙いをつける事で，力の方向を直感的に指示することができると考える．

以上のように，「指を弾く動作」は力の制御と方向の制御を同時に，かつ直感的に行う事ができる点で優れていると言える．

3　ハイスピードカメラを用いたビジョンベースでの認識

人間の指は細くて小さく，また大きさや長さにも個人差がある．従って指に対し，位置や加速を検出するようなセンサを取り付ける事は難しく，またこれらのセンサは動作の邪魔になる恐れもある．

そのため本研究では画像認識を用いたビジョンベースの認識手法で指の動きを捉える事で指の速さを測定する．また指を弾く時の手，腕の位置や，弾かれるオブジェクトとの位置関係をカメラ画像から得る事で方向を取得する．

しかしビジョンベースの入力手法は専用のセンサー等の特殊な入力装置を身につける必要がない

ためユーザの動きを制限しないといった利点があるがカメラのスキャンレートの低さや，シャッタースピードの遅延の影響で，特殊な入力装置を用いた手法と比べて認識精度や認識速度で劣るという問題があり，高速に動く指の動きの認識が難しいと言う欠点があった．そこで本研究では高速な動きを捕らえる事のできるハイスピードカメラと，最小限の処理を行う事で高速化した画像処理を用いて指の動きを認識する手法を開発した．

4 テーブルトップおはじきシステム

指を弾く動作はおはじき等の遊戯に直接用いられる動作である事から，この動作をコンピュータ上で仮想的に再現し，また拡張を行う事で様々なエンタテインメントアプリケーションに応用できると考える．

このシステムはおはじきインタフェースを用いて実際の「おはじき」という遊戯を仮想的に再現するシステムであり，ユーザは机上に投影されたおはじきを，実際のおはじきと同じように指で弾いて飛ばす事ができる．

4.1 ハードウェア構成

本システムのハードウェア構成は図 2 のようになる．画像処理プログラムが動作する画像処理用 PC と，アプリケーションが動作するアプリケーション用 PC からなり，それぞれに手の動きを撮影するハイスピードカメラ，机上におはじきを投影するプロジェクタが接続されている．

カメラは机上方の天井に取り付けられ，カメラの視界は机全面を撮影する．

プロジェクタも同様に机の上方に固定され，机全面に映像を投影する．机にはユーザの手の他に，プロジェクタによって投影されたおはじきの画像やスコア等のテキスト情報が描画される．

本システムで用いているハイスピードカメラは Point Grey Research 社「Dragonfly Express」である．Dragonfly Express は 640×480（8bit グレイスケール）で最大 200fps の撮影が可能で，さらに撮影範囲を絞る事で 320×240 で最大 350fps，160×120 で最大 420fps 等での撮影が可能である．また撮影した画像は IEEE1394b を通してリアルタイムに PC に転送する事ができる．

4.2 カメラキャリブレーション

カメラ座標中の手の位置をアプリケーションで用いたり，逆にアプリケーション上でのおはじきの位置をカメラ側で用いる場合にはカメラ座標とディスプレイ座標の座標変換を行う必要がある．本システムではパースペクティブ変換を用いて座標変換を行っている．

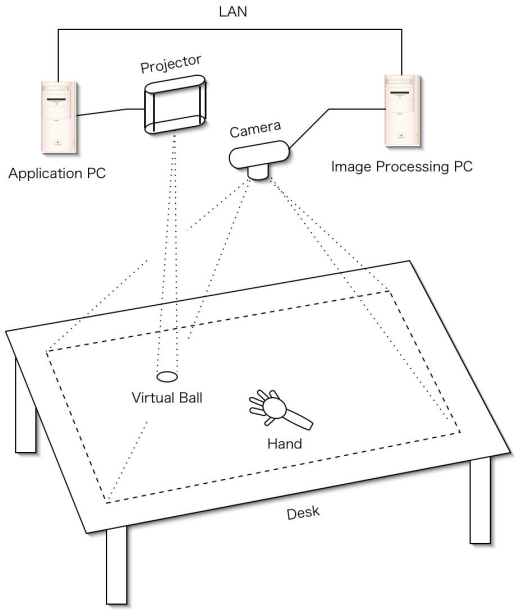

図 2. 本システムは画像処理用の PC（Xeon 3.2Ghz, 2GB RAM），とアプリケーション用 PC（Pentium4 3.2Ghz,1GB RAM），カメラ，プロジェクタから構成される．机の色は背景差分を行う際に照明によってできる手の影が影響しないように黒色にした．

4.3 認識の流れ

このシステムは画像処理用 PC 上で動作する「認識システム」と，アプリケーション用 PC 上で動作する「アプリケーションシステム」の 2 つのシステムで構成され，互いにネットワークで情報をやり取りしながら次の手順のようにユーザの手の認識とおはじきの投影を行う．

1. 構え検出とおはじきへの接近の判断
   認識システムはユーザの手が机上で図 3(左) のような「構え」の状態にある事を，指で囲まれた領域の有無を調べる事で検出する．

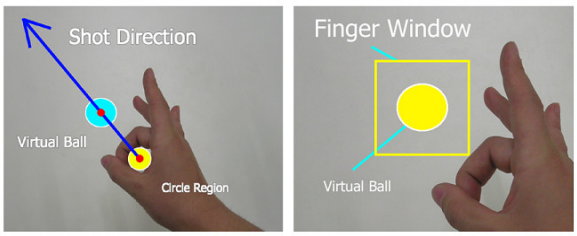

図 3. 構え状態の手とおはじきの 2 点を使った方向の設定 (左) と指認識ウインドウの位置 (右)．なお本システムでは人差し指や中指を親指にかけて弾く動作を認識対象とする．

この領域はユーザが手を構えた場合のみ生じ，

また領域の重心は手の平の重心にも近い事からユーザの手の位置としても用いる．この指で囲まれた領域はこれまでにも手のロバストな認識に用いられてきた [3]．構え状態の手が机上に投影されたおはじきに接近するとユーザがおはじきへ狙いを付ける動作を行ったとみなし，ユーザの狙う動作の検出処理へ移行する．

2. ユーザの狙う動作の検出
ユーザが机上の何処に手を置き，おはじきをどの角度から弾こうとしているのかを検出する．この角度はユーザの手の位置とおはじきの位置の 2 点を用いて図 3(左) のように求める．このとき，ユーザには方向が分かりやすいように机に矢印を描画する．

   ここでユーザが数秒間手の動きを止めると，ユーザがおはじきを弾く準備が出来たと判断し，アプリケーションは認識システムに投影されているおはじきの位置を送信し，おはじきの投影されている付近の机上での指先検出を開始させる．またこの時音を鳴らしユーザにも指を弾く事が可能になった事を通知する．

3. 指認識ウインドウの設定
アプリケーションからおはじきの位置を取得した認識システムは，ユーザがおはじきを指で弾いた時に指が指認識ウインドウ内に入り，ウインドウ内を移動して再び外に出るような図 3(右) のような位置にウインドウを設定する．このとき，構え方や手を置く位置の個人差によりウインドウ内に弾く指以外の部分が入ったり，逆に遠すぎて指がウインドウに届かない等の問題が生じる場合があるため速さ検出に移行する直前にウインドウを 1pixel ずつ動かして最適な位置にウインドウを設定する処理を行う．

4. 指の速さの推定
カメラの撮影範囲をウインドウ内だけに切り替え，ウインドウ内を移動する指に対してその速さを推定する処理を行う．指の速さの推定方法については後で詳しく述べる．ユーザが指を弾き終えると，推定された指の速さはアプリケーションに通知される．

   もしユーザが指を弾いた場合に適切に指の位置が取得出来なかった場合は「空振り」したとみなす．また一定時間指が弾かれなかった場合はユーザが狙う角度を変更したとみなし，おはじき検出モードへ戻る．

5. おはじきへの力の適用
検出された指の速度（指の速さと手とおはじきの位置から計算した角度）はおはじきへと即座に反映され，おはじきは弾き飛ばされたように移動する．おはじきが弾かれると，処理は初期状態へ戻る．

5 指の速さの推定方法

これまで指の速さの推定は指先位置を検出する方法を用いてきた．今回，指先検出手法を改良すると共に，新たに指の速さの推定方法として，角速度を使った指の速さの推定の手法を考えた．

5.1 指先位置の検出を用いた手法

弾かれた指先の位置を検出し指先の移動距離を求める事で指先の速さを推定する手法である．この処理は次のような手順で行う．

1. テンプレートマッチングによりウインドウ内の指先の位置を求める（図 4）．

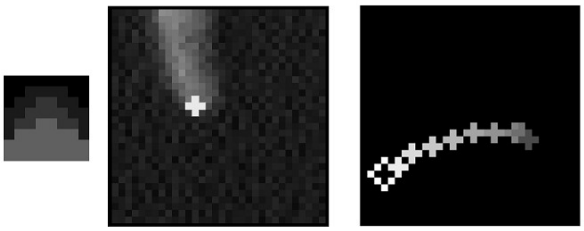

図 4. 8×8 のテンプレート画像 (左) と指先の検出結果 (中) および指先の移動の軌跡 (右)．どの方向からでも指先が検出できるように，テンプレート画像は手の位置に応じて回転させる．

2. この処理を指がウインドウ内に入ってから，再び外に出るまで毎フレーム行い，指先位置やタイムスタンプを記録していく．この時のカメラのスキャンレートおよび画像処理のフレームレートは約 630fps である．

3. 指が領域外に出たら次の式を用いて指の速さを推定する．

$$V = \frac{\sum_{i=1}^{n-1} L_i}{T} \tag{1}$$

   ウインドウ内で最初に指が検出されてから最後に検出されるまでの時間を T[msec]，検出された n 番目と n+1 番目の指の間の距離を L_n [pixel] とすると指の速さ V [pixel/msec] は次の式（1）のように求める．ここでの時間間隔を計測するタイマは，CPU の RDTSC 命令を用いたクロックカウンタを計測する手法を用いておりナノ秒オーダの測定が行える．

以前の実装では入力画像から背景差分により指の領域だけを抽出し，そこから円形のテンプレート画

像を用いて指先の検出を行っていたが，探索ウインドウの大きさが 32×32 と非常に小さいため，二値化により指先の画像が崩れ，指先が正しく検出出来ない場合がある事が分かった．そこで今回は二値化は行わず，その代わり図 4 のような画像をテンプレートとして用いる事でマッチングを行うようにした．

さらに，探索ウインドウに指が入りかけた状態と，出かけた状態では指先の検出が失敗する事があるため，指先がウインドウ境界周辺にある場合はこれを無視し，一定以上ウインドウの中に入った指のデータだけを用いるようにした．

以上の認識処理の改良により指先検出の誤検出を軽減させる事が出来た．

5.2 角速度を用いた手法

指の動きを単純な弧を描く動きであると仮定し，ウインドウ内の指の角度を検出する事で角速度を求め，指の速さとする手法である．この処理は次のような手順で行う．

1. 背景差分と領域分割を用いて，ウインドウ内の指の 2 値画像を得る．
2. 得られた指画像に対してモーメントから重心および慣性主軸を得る事で指の角度を求める．この処理を指がウインドウ内に入ってから再び外にでるまで繰り返し，指の角度を記録していく．この時のカメラのスキャンレートおよび画像処理のフレームレートも約 630fps である．
3. 指が領域外に出たら，次の式（2）のように，指が領域に入った時の指の角度 A_s[rad] と，最後に領域外に出た時の指の角度 A_e[rad] の差を求め，移動に要した時間 T[msec] で割ることで角速度 V_{angle}[rad/msec] を求める．

$$V_{angle} = \frac{A_e - A_s}{T} \qquad (2)$$

この手法においても，指先検出のサイト同様にウインドウの境界周辺の指は無視する事で誤検出を回避している．

6 システムの評価

今回，指先の移動量と角速度を用いた手法において，指の速さの検出が行えているかどうか，またユーザによる指にかける力の強さの制御で机上のおはじきの飛距離を制御できているかを検証する実験を行った．

さらに実際のおはじきを指で弾いた場合の結果も調べ，人間の「指を弾く動作」がどの程度の力の制御を可能としているかを調べ，本システムとの比較を行った．

6.1 実験手順

今回の実験では机上に投影されたおはじきを，あらかじめ机上に定められた目標地点に向かって飛ばしそしてどれだけ目標の近くにおはじきを停止させる事が出来たかを測定する．

おはじきの初期位置と目標との位置関係は，図のようになっている．被験者にはそれぞれの目標にむかって遠い目標から順番に 5 回ずつ投影されたおはじきを弾いてもらう．またこの時のおはじきには，現実のおはじきを弾く場合と同様に摩擦力を発生させている．この時の摩擦係数等の物理パラメータについては，厳密に机の表面とおはじきとの摩擦係数を求めたわけでなく適当な値を用いている．これは本実験においては被験者が 1 回目に弾いた時の飛距離から次に弾く時の力加減の見当がつくものと考えたためである．

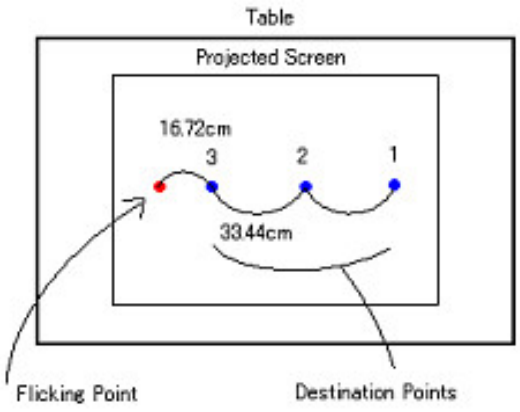

図 5. 机上に 3 箇所の目標地点を置いた．投影面の幅は約 110m である．目標地点の位置は，おはじきを弾く位置から近い場所 (距離 100pixel = 33.44cm)，中くらいの場所 (距離 200pixel)，遠い場所 (距離 300pixel) の 3 箇所である．

実験を単純化させるために，方向については予め目標に向かって固定しておき，弾く強さのパラメータのみを用いた．

実際のおはじきを用いた場合は重さ約 $12g$，直径約 2.5cm のおはじきを用い，また本システムで用いている指の弾き方を使い，システムと同じ机上での飛距離のみを同様に測定した．

被験者は著者らのいる研究室の大学院生 5 人である．被験者全員が過去に「おはじき」やその他の指を弾く遊戯で遊んだ経験を持つ者であるが，その経験は一般的なもので，ごく浅いと考えられる．被験者には 3 分程度の操作の説明と，操作に慣れるための 10 分程度の練習時間を与え，指先検出，角速度，本物の順に 5 回ずつ試行を行った．また画像処理の閾値等のパラメータは各ユーザに合わせて設定を行った．

6.2 実験結果

得られた実験結果を次の図 6.2 に示す (グラフは上から指先検出，角速度，本物のおはじきを用いた手法の順).

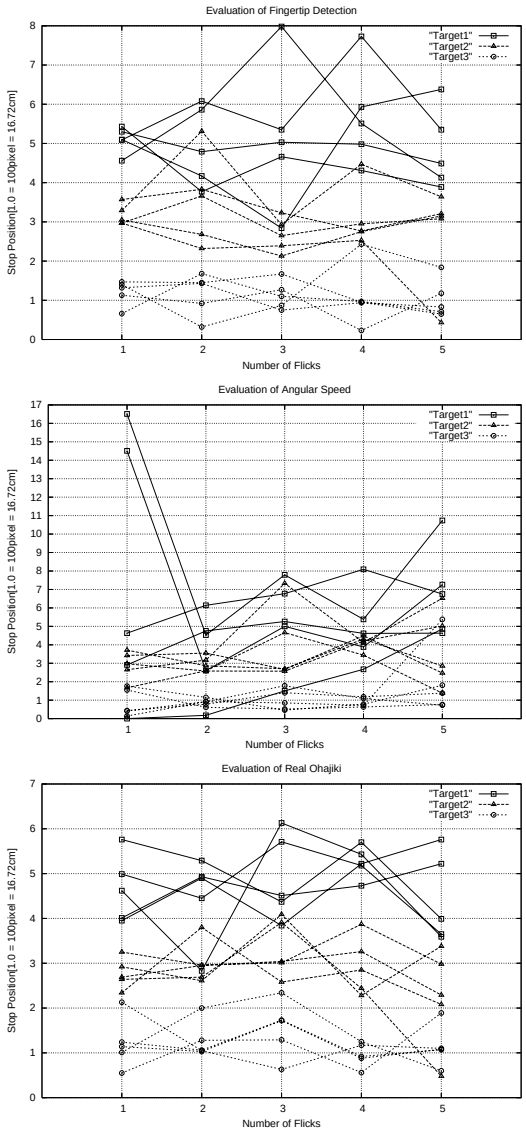

図 6. X 軸が 1 回目から 5 回目までの試行，y 軸はおはじきの到達距離を表しておりおはじきの初期位置は $y = 0$ である．3 つの目標地点 (Target1〜Target3) はそれぞれ $y = 1, 3, 5$ の位置にある．

またユーザごとの目標位置からの距離の平均値は次の表 2 のようになった.

7 考察

7.1 人間の指を弾く動作についての考察

今回の実験結果を見ると，我々の予想に反して，本物のおはじきを弾いた場合の精度が良くなかった．これにはいくつかの原因が考えられる．

表 1. 指先移動量を用いた手法においての狙った位置からのずれの平均値

User	Target 1	Target 2	Target 3
User 1	0.414	0.234	0.264
User 2	0.108	0.872	0.286
User 3	0.920	0.414	0.288
User 4	1.132	0.956	0.696
User 5	0.760	0.328	0.398
Average	0.667	0.561	0.271

表 2. 角度を用いた手法においての狙った位置からのずれの平均値

User	Target 1	Target 2	Target 3
User 1	0.670	0.522	0.350
User 2	3.158	0.820	0.406
User 3	1.156	1.900	1.150
User 4	1.424	0.664	0.496
User 5	4.178	1.092	0.360
Average	2.177	1.000	0.552

まず指を弾く動作というものが，実際のスポーツと同様に指への微妙な力加減の制御が必要な動作であることから被験者のその動作への慣れが影響してくる事が考えられる．今回の被験者においても，日常的に指を弾く練習を行っているわけではないため思ったより力の加減が出来ずにもどかしいという意見もあった．以上の事から判断すると，指を弾く時の力のコントロールにはある程度慣れが必要であり，うまく加減ができるようになるには練習を行って動作に慣れる必要があるようである．

また，指の力加減以外の要素が結果に影響した事も考えられる．おはじきを弾く場合，おはじきの飛距離には指の力加減以外の要素，例えばおはじきの指への接触位置や，角度，接触といったものの影響も考えられる．同じ強さで指を弾いた場合でも，当り具合によっては飛距離にばらつきが出る場合もあるだろう．これもある程度慣れてくるとどこに当てればどのような飛び方をするか身についてくるだろうがそこまで慣れるにはスポーツと同様にある程度の経験が必要だと考えられる．

7.2 指の速さの推定方法について

次に，今回のシステムで測定したデータについて考察する．本物のおはじきを弾いた場合の結果を考慮すると，システムでの測定結果にある程度のばらつきがある事は予想出来た.

まず指先検出においては，目標との距離やユーザによってユーザの狙った位置からのずれは表 2 のようである事が分かった．このずれは近い目標であると数 cm，遠い目標であると十数 cm 程度である．本システムがゲーム用のインタフェースである事を考慮

すれば，この手法による指先の速さの推定は数 cm 程度のずれである事から許容できる範囲であろう．目標が遠くなるにつれずれが大きくなっている点については，ユーザが狙いにくいのに加え，より速く指を弾く必要があるため力の制御がしにくい事や，指先の検出の回数が減る事が考えられる．

次に角速度を用いた手法であるが，指先検出と比較すると精度が良くない事がわかった．原因としては，今回の実装では指が入った時と出た時の角度の差分のみしか用いておらず，その間の指の動きを考慮していない点が精度に影響したと考えられる．より精度を高めるためには，ウインドウ内を移動する指の角速度を各フレーム毎に求めその平均値を用いるといった，間の指の動きを考慮する認識を用いる必要がありそうだ．

またどちらの方法にも言える事だが，今回は指先検出のスキャンレートを高めるために 32×32 という非常に小さなウインドウを用いての速度推定を行っている．もし，より高い解像度で高速に撮影が可能なカメラを用いる事が出来れば精度の向上が期待できるし，また例えば指がおはじきに接触する直前の指の速さを求めたり，おはじきのどの部分に当たったか等を検出する事も可能になると考えられる事からより厳密な速さの推定が可能になるであろう．

7.3 誤検出の問題について

また誤検出の問題もある事がわかった．今回の実験においても，この誤検出により極端に強い値や弱い値が測定された．今回見られた誤検出の原因は，大きく 2 つある．まず 1 つは，ウインドウ内に弾かれた指以外の部分が入ってしまう場合である．2 つ目は，ウインドウ内にうまく入らず，ウインドウの境界付近を移動した場合である．

前者の場合，指先検出においては誤った指先位置を記録してしまったり，テンプレートマッチングのスコアが低下する事で，ウインドウ内に指先とみなせる領域が見つからなくなり指先がウインドウ内にある時に指先検出が途中で終了していしまう場合がある．角度の測定においても，ウインドウ内に指の領域を正しく見つける事が出来なくなり，正確な角度が測定されない場合がある．今回の測定においても，グラフ 2 の 1 回目の試行における極端に大きな飛距離を示した 2 つのデータがこれであった．

次に後者であるが，ウインドウの境界付近を指が横切ると，カメラには指先の一部分しか撮影されない事になる．このような場合に指先検出を行うと，殆どのデータが有効なデータとして扱われずに指先検出の回数が極端に少なくなってしまう．また角度検出においては，指先の領域だけから角度を推定する事になるため，正しい角度が測定できなくなる場合がある．

これらの誤検出の原因は，指を弾く際の手を置く場所が近すぎたり，遠すぎた場合に起こる．安定して指の速さを推定するためには，領域に十分指が入っている事が望ましい．この問題を解決するためには，ユーザの手の置かれた位置とおはじきの位置関係を指先検出に移行する直前に判断し，ウインドウを適切な位置に移動させる処理を導入する事が有効だと思われる．

8 アプリケーション

今回アプリケーションとして机上でおはじきをぶつけ合って遊ぶアプリケーションを開発した．

図 7. 投影されたおはじきを弾くプレイヤーの様子．

このゲームは机上のおはじきを複数人で順番に弾き，相手のおはじきにぶつけ机の外に弾き飛ばす．そして相手のおはじきを机の外に弾き飛ばしたら勝ちとなるゲームである．

9 まとめと課題

本研究では，ハイスピードカメラを用いて指を弾くジェスチャを認識する事で，机上に投影されたバーチャルなおはじきを指を弾いて遊ぶ事ができる「テーブルトップおはじきシステム」を提案した．

今後は弾き方の個人差の識別や誤検出回避等の使い易さを高める改良を行い，様々な指を弾いて遊ぶ遊戯を構築し，運用および評価を行いたい．

参考文献

[1] T. Sato, K. Fukuchi, and H. Koike. OHAJIKI Interface: Flicking Gesture Recognition with a High-Speed Camera. *Proceedings of IECE 2006, pp.205-210.*

[2] 佐藤俊樹, 福地健太郎, 小池英樹. おはじきインタフェース: ハイスピードカメラを用いた指を弾くジェスチャの認識. 情報処理学会研究報告 2006-HI-119, pp. 103–110. 情報処理学会, 2006.

[3] A. Wilson. Robust Vision-Based Detection of Pinching for One and Two-Handed Gesture Input. *Proceedings of UIST 2006, pp.255-258.*