

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第4445898号  
(P4445898)

(45) 発行日 平成22年4月7日(2010.4.7)

(24) 登録日 平成22年1月22日(2010.1.22)

(51) Int.Cl.

F I

**A 6 3 F 13/00 (2006.01)**

A 6 3 F 13/00

F

**A 6 3 F 13/10 (2006.01)**

A 6 3 F 13/00

C

**G O 6 T 13/00 (2006.01)**

A 6 3 F 13/00

P

A 6 3 F 13/10

G O 6 T 13/00

A

請求項の数 14 (全 27 頁)

(21) 出願番号 特願2005-150354 (P2005-150354)  
 (22) 出願日 平成17年5月24日(2005.5.24)  
 (65) 公開番号 特開2006-325695 (P2006-325695A)  
 (43) 公開日 平成18年12月7日(2006.12.7)  
 審査請求日 平成20年4月28日(2008.4.28)

早期審査対象出願

前置審査

(73) 特許権者 000233778  
 任天堂株式会社  
 京都府京都市南区上鳥羽鉾立町1番地1  
 (74) 代理人 100098291  
 弁理士 小笠原 史朗  
 (74) 代理人 100151541  
 弁理士 高田 猛二  
 (74) 代理人 100130269  
 弁理士 石原 盛規  
 (72) 発明者 岩本 大貴  
 京都府京都市南区上鳥羽鉾立町1番地1  
 任天堂株式会社内  
 (72) 発明者 新田 昌弘  
 京都府京都市南区上鳥羽鉾立町1番地1  
 任天堂株式会社内

最終頁に続く

(54) 【発明の名称】 ポインティングデバイスに対する入力を用いたゲームプログラムおよびゲーム装置

(57) 【特許請求の範囲】

【請求項1】

入力面上においてプレイヤーが指定した位置を検出する入力装置を備えるゲーム装置のコンピュータに実行させるゲームプログラムであって、

前記入力装置によって繰り返し検出された各入力位置を、前記ゲーム装置のメモリに記憶させる記憶ステップと、

前記入力面上においてプレイヤーによって描かれた軌跡の少なくとも一部により示される方向と、プレイヤーキャラクタの位置から前記軌跡によって決められる位置への方向との角度を判定する角度判定ステップと、

前記角度判定ステップで判定された角度に基づいたアクションを前記プレイヤーキャラクタが行う様子を表示装置に表示させる表示制御ステップとを前記コンピュータに実行させる、ゲームプログラム。

【請求項2】

前記表示制御ステップにおいては、前記プレイヤーキャラクタが表示される位置から前記軌跡の始点、中点、または終点への向きに前記プレイヤーキャラクタが前記角度判定ステップで判定された角度に基づいたアクションを行う様子が表示される、請求項1に記載のゲームプログラム。

【請求項3】

前記プレイヤーキャラクタは前記表示装置の画面の中央に位置する、請求項1に記載のゲームプログラム。

10

20

## 【請求項 4】

前記入力装置によって検出された入力位置に基づいて前記プレイヤーキャラクタの移動方向を決定する移動決定ステップをさらに前記コンピュータに実行させ、

前記表示制御ステップは、前記移動決定ステップにおいて決定された移動方向に前記プレイヤーキャラクタが移動する様子を前記表示装置に表示させるステップを含む、請求項 1 に記載のゲームプログラム。

## 【請求項 5】

前記移動決定ステップは、前記角度判定ステップで判定された角度に基づいたアクションを前記プレイヤーキャラクタが行っていないことを条件として、前記入力装置によって入力位置が検出される度に実行される、請求項 4 に記載のゲームプログラム。

10

## 【請求項 6】

前記表示制御ステップにおいては、前記軌跡の長さが所定値以上であるときのみ、前記角度判定ステップで判定された角度に基づいたアクションを前記プレイヤーキャラクタが行う様子が前記表示装置に表示される、請求項 1 に記載のゲームプログラム。

## 【請求項 7】

前記表示制御ステップにおいては、前記軌跡が所定時間以内に入力されたときのみ、前記角度判定ステップで判定された角度に基づいたアクションを前記プレイヤーキャラクタが行う様子が前記表示装置に表示される、請求項 1 に記載のゲームプログラム。

## 【請求項 8】

前記角度判定ステップは、

20

前記入力面上においてプレイヤーによって描かれた軌跡を示す軌跡ベクトルを前記各入力位置から算出する軌跡ベクトル算出ステップと、

前記プレイヤーキャラクタの位置を始点とし、前記軌跡ベクトルによって決められる位置を終点とする基準ベクトルを算出する基準ベクトル算出ステップとを含み、

前記表示制御ステップにおいては、前記軌跡ベクトルと前記基準ベクトルとのなす角の大きさに基づいた前記プレイヤーキャラクタのアクションを前記プレイヤーキャラクタが行う様子が前記表示装置に表示される、請求項 1 に記載のゲームプログラム。

## 【請求項 9】

前記軌跡ベクトル算出ステップは、前記入力面上においてプレイヤーによって描かれた軌跡を 1 以上の直線に近似し、当該近似された各直線を示すベクトルを前記軌跡ベクトルとして当該直線毎にそれぞれ算出するステップを含み、

30

前記表示制御ステップにおいては、各前記軌跡ベクトルと前記基準ベクトルとのなす角の大きさに基づいたそれぞれのアクションを連続して前記プレイヤーキャラクタが行う様子が前記表示装置に表示される、請求項 8 に記載のゲームプログラム。

## 【請求項 10】

前記表示制御ステップにおいては、前記軌跡ベクトル算出ステップにおいて前記軌跡ベクトルが複数算出されるとき、前記軌跡ベクトルが 1 つ算出されるときとは異なるアクションを前記プレイヤーキャラクタが行う様子が前記表示装置に表示される、請求項 9 に記載のゲームプログラム。

## 【請求項 11】

40

前記軌跡ベクトル算出ステップにおいては、前記入力装置によって連続して検出された各入力位置のうちで、最初に検出された入力位置を始点とし、最後に検出された入力位置を終点とするベクトルが前記軌跡ベクトルとして算出される、請求項 8 に記載のゲームプログラム。

## 【請求項 12】

前記軌跡ベクトル算出ステップは、

前記入力装置によって連続して検出された各入力位置を検出された順に結ぶ折れ線について、互いに接続される 2 つの線分のなす角度が所定値以下となる頂点に対応する入力位置を選出するステップと、

前記選出された入力位置が 2 つ以上存在するとき、順に検出された 2 つの入力位置を

50

検出順に結ぶベクトルを前記軌跡ベクトルとして算出するステップとを含む、請求項 8 に記載のゲームプログラム。

【請求項 13】

前記基準ベクトル算出ステップにおいて、前記軌跡ベクトルの終点の位置を基準ベクトルの終点として当該基準ベクトルが算出される、請求項 8 に記載のゲームプログラム。

【請求項 14】

入力面上においてプレイヤーが指定した位置を検出する入力装置を備えるゲーム装置であって、

前記入力装置によって繰り返し検出された各入力位置を、前記ゲーム装置のメモリに記憶させる記憶手段と、

前記入力面上においてプレイヤーによって描かれた軌跡の少なくとも一部により示される方向と、プレイヤーキャラクタの位置から前記軌跡によって決められる位置への方向との角度を判定する角度判定手段と、

前記角度判定手段で判定された角度に基づいたアクションを前記プレイヤーキャラクタが行う様子を表示装置に表示させる表示制御手段とを備える、ゲーム装置。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、ゲームプログラムおよびゲーム装置に関し、より特定的には、ポインティングデバイスに対する入力をゲーム操作に用いたゲームプログラムおよびゲーム装置に関する。

【背景技術】

【0002】

従来、マウスやタッチパネル等のポインティングデバイスを入力装置として用いたゲーム装置が考えられている。例えば、画面上の敵キャラクタをマウスで指定した状態でマウスの入力ボタンを押すことによって、プレイヤーキャラクタが敵キャラクタに対して攻撃を行うゲームがある（例えば、非特許文献 1 参照。）。このゲームでは、プレイヤーはマウスを用いてプレイヤーキャラクタを移動させる。そして、敵キャラクタに対する攻撃が可能な間合いにプレイヤーキャラクタを移動させ、マウスの左ボタンをクリックすることにより攻撃を行う。さらに、プレイヤーは、左ボタンを連打することによって連続攻撃を行うことも可能である。

【非特許文献 1】“黄昏の世界を旅する、動きの美しいアクションRPG 「Revenant」”、[online]、1999年8月18日、駒沢 丈治、[2005年4月13日検索]、インターネット<URL: <http://www.forest.impress.co.jp/article/1999/08/18/game48.html>>

【発明の開示】

【発明が解決しようとする課題】

【0003】

上記のゲームにおける攻撃操作は、マウスの左ボタンをクリックするだけであるので、非常に単純である。プレイヤーによる攻撃操作は 1 種類しかなく、この 1 種類の攻撃操作に対してプレイヤーキャラクタが行う攻撃パターンが一義的に決まってしまうので、攻撃操作は非常に単調になってしまう。そのため、プレイヤーが操作に慣れると攻撃操作の面白みはすぐに薄れてしまう。つまり、ポインティングデバイスの入力を用いた従来のゲーム装置では、バリエーションに富んだアクションをプレイヤーキャラクタに行わせることができず、ゲーム操作が単純になってしまうという課題があった。また、ゲーム操作が単純であることから、ゲームの戦略性や興趣性が低くなってしまうという課題があった。

【0004】

それ故、本発明の目的は、ポインティングデバイスの入力を用いたゲームにおいて、プレイヤーキャラクタに多彩なアクションを行わせることが可能なゲームプログラムおよびゲーム装置を提供することである。

## 【課題を解決するための手段】

## 【0005】

本発明は、上記の課題を解決するために、以下の構成を採用した。なお、本欄における括弧内の参照符号および補足説明等は、本発明の理解を助けるために後述する実施形態との対応関係を示したものであって、本発明を何ら限定するものではない。

## 【0006】

第1の発明は、入力面上においてプレイヤーが指定した位置を検出する入力装置（タッチパネル15）を備えるゲーム装置（10）のコンピュータ（CPUコア21等）に実行させるゲームプログラムである。ゲームプログラムは、記憶ステップ（S5またはS11）と、角度判定ステップ（S43またはS72、S62、S63、S64）と、表示制御ステップ（S65）とをコンピュータに実行させる。記憶ステップにおいては、コンピュータは、入力装置によって繰り返し検出された各入力位置を、ゲーム装置のメモリ（RAM24）に記憶させる。角度判定ステップにおいては、入力面上においてプレイヤーによって描かれた軌跡の少なくとも一部により示される方向と、プレイヤーキャラクタの位置から前記軌跡によって決められる位置への方向との角度（図4に示す1）を判定する。表示制御ステップにおいては、上記角度判定ステップで判定された角度に基づいたアクションをプレイヤーキャラクタが行う様子を表示装置（第2のLCD）に表示させる。

10

## 【0007】

また、第2の発明においては、表示制御ステップにおいては、プレイヤーキャラクタが表示される位置から軌跡の始点、中点、または終点への向きにプレイヤーキャラクタが上記角度判定ステップで判定された角度に基づいたアクションを行う様子が表示されてもよい。

20

## 【0008】

また、第3の発明においては、プレイヤーキャラクタは表示装置の画面の中央に位置してもよい。

## 【0009】

また、第4の発明においては、ゲームプログラムは、移動決定ステップ（S52およびS53）をさらにコンピュータに実行させてもよい。移動決定ステップにおいては、コンピュータは、入力装置によって検出された入力位置に基づいてプレイヤーキャラクタの移動方向を決定する。このとき、表示制御ステップは、移動決定ステップにおいて決定された移動方向にプレイヤーキャラクタが移動する様子を表示装置に表示させるステップ（S54）を含む。

30

## 【0010】

また、第5の発明においては、移動決定ステップは、上記角度判定ステップで判定された角度に基づいたアクションをプレイヤーキャラクタが行っていないことを条件として、入力装置によって入力位置が検出される度に実行されてもよい。

## 【0011】

また、第6の発明においては、表示制御ステップにおいては、軌跡の長さが所定値以上であるときのみ、上記角度判定ステップで判定された角度に基づいたアクションをプレイヤーキャラクタが行う様子が表示装置に表示されてもよい（図19に示すステップS73参照）。

40

## 【0012】

また、第7の発明においては、表示制御ステップにおいては、軌跡が所定時間以内に入力されたときのみ、上記角度判定ステップで判定された角度に基づいたアクションをプレイヤーキャラクタが行う様子が表示装置に表示されてもよい（図16に示すステップS32、図19に示すステップS71参照）。

## 【0013】

また、第8の発明においては、角度判定ステップは、入力面上においてプレイヤーによって描かれた軌跡を示す軌跡ベクトルを各入力位置から算出する軌跡ベクトル算出ステップ（S43またはS72）と、プレイヤーキャラクタの位置を始点とし、軌跡ベクトルによって決められる位置を終点とする基準ベクトルを算出する基準ベクトル算出ステップ（S6

50

２）とを含み、表示制御ステップにおいては、軌跡ベクトルと基準ベクトルとのなす角の大きさに基づいたプレイヤーキャラクタのアクションをプレイヤーキャラクタが行う様子が表示装置に表示されてもよい。

【００１４】

また、第９の発明においては、軌跡ベクトル算出ステップは、入力面上においてプレイヤーによって描かれた軌跡を１以上の直線に近似し、当該近似された各直線を示すベクトルを軌跡ベクトルとして当該直線毎にそれぞれ算出するステップ（Ｓ３１，Ｓ３２およびＳ３４）を含んでいてもよい。このとき、表示制御ステップにおいては、各軌跡ベクトルと基準ベクトルとのなす角の大きさに基づいたそれぞれのアクションを連続してプレイヤーキャラクタが行う様子が表示装置に表示される。

10

【００１５】

また、第１０の発明においては、表示制御ステップにおいては、軌跡ベクトル算出ステップにおいて軌跡ベクトルが複数算出されるとき、軌跡ベクトルが１つ算出されるときとは異なるアクションをプレイヤーキャラクタが行う様子が表示装置に表示されてもよい。

【００１６】

また、第１１の発明においては、軌跡ベクトル算出ステップにおいては、入力装置によって連続して検出された各入力位置のうちで、最初に検出された入力位置（図５に示す点Ｐ２）を始点とし、最後に検出された入力位置（図５に示す点Ｐ３）を終点とするベクトルが軌跡ベクトルとして算出されてもよい。

【００１７】

20

また、第１２の発明においては、軌跡ベクトル算出ステップは、次の２つのステップを含んでいてもよい。すなわち、１つ目のステップ（Ｓ３１およびＳ３２）においては、コンピュータは、入力装置によって連続して検出された各入力位置を検出された順に結ぶ折れ線について、互いに接続される２つの線分のなす角度が所定値以下となる頂点に対応する入力位置を選出する。２つ目のステップ（Ｓ３３）においては、選出された入力位置が２つ以上存在するとき、順に検出された２つの入力位置を検出順に結ぶベクトルを軌跡ベクトルとして算出する。

【００１８】

また、第１３の発明においては、基準ベクトル算出ステップにおいて、軌跡ベクトルの終点の位置を基準ベクトルの終点として当該基準ベクトルが算出されてもよい。

30

【００１９】

また、本発明は、上記ゲームプログラムを実行することによって実現される機能と同等の機能を有するゲーム装置の形態で提供されてもよい。

【発明の効果】

【００２０】

本願発明によれば、プレイヤーは、画面上に描く軌跡を変化させることによって複数種類のアクションを使い分けることができる。すなわち、ポインティングデバイスをを用いたゲームにおいても、プレイヤーキャラクタに多彩なアクションを容易に行わせることができる。例えば、攻撃のアクションについて第１の発明を適用すれば、所定位置から入力軌跡への方向と、入力軌跡が描かれた方向との関係によって、プレイヤーキャラクタの攻撃パターンを変化させることができる。これによって、従来では単調となっていた敵キャラクタへの攻撃にバリエーションを持たせることができ、ゲームの戦略性や興趣性を向上することができる。

40

【００２１】

本願発明によれば、画面上に表示されるプレイヤーキャラクタと入力軌跡との関係に応じて、プレイヤーキャラクタのアクションを変化させることができる。画面上におけるプレイヤーキャラクタの位置は変化するので、画面上の同じ位置に同じ入力軌跡を描いたとしても、プレイヤーキャラクタの位置に応じてアクションが変化する。したがって、プレイヤーは、プレイヤーキャラクタの位置を考慮しながら所望のアクションを行わせるための入力軌跡を描くという、従来にはないゲームの操作感を味わうことができる。

50

## 【 0 0 2 2 】

本願発明によれば、プレイヤーキャラクタは、軌跡が描かれた方向にアクションを行うので、プレイヤーは、軌跡を描く操作によってアクションを行う方向を指定することができる。したがって、アクションを行う方向を指定する操作と軌跡を描く操作とを別に行う必要がないので、プレイヤーの操作が簡易になり、ゲームの操作性を向上することができる。

## 【 0 0 2 3 】

本願発明によれば、プレイヤーは、画面中央に表示されるプレイヤーキャラクタを中心として上下左右のどの方向にも軌跡を入力することができる。したがって、プレイヤーは、プレイヤーキャラクタにアクションを行わせる操作をどの方向であっても容易に行うことができる。

10

## 【 0 0 2 4 】

本願発明によれば、プレイヤーキャラクタを移動させる操作と、プレイヤーキャラクタにアクションを行わせる操作とを単一の入力装置で行うことができるので、ゲームの操作性を向上することができる。

## 【 0 0 2 5 】

本願発明によれば、所定のアクションを行っている間はプレイヤーキャラクタが移動することはないので、アクション中のプレイヤーキャラクタが移動するという不自然な動作を防止することができる。

## 【 0 0 2 6 】

本願発明によれば、アクションを行う操作と移動操作とを区別することができる。すなわち、軌跡ベクトルの大きさが所定値よりも小さい場合には、プレイヤーキャラクタの移動のみが行われる。プレイヤーは、移動操作を行いたい場合には、軌跡ベクトルの大きさが所定値よりも小さくなるように軌跡を描けばよく、これにより、プレイヤーが意図しないアクションが行われることがない。これによって、上記2つの操作を行う場合に操作性を向上することができる。また、プレイヤーは、入力面に対する入力操作という1種類の操作方法で、2つの操作を使い分けることができる。

20

## 【 0 0 2 7 】

本願発明によれば、アクションを行う操作と移動操作とを区別することができる。すなわち、プレイヤーは、移動操作を行いたい場合には、軌跡を描く操作を所定時間よりも長く行えばよく、これにより、プレイヤーが意図しないアクションが行われることがない。これによって、上記2つの操作を行う場合に操作性を向上することができる。また、プレイヤーは、入力面に対する入力操作という1種類の操作方法で、2つの操作を使い分けることができる。

30

## 【 0 0 2 8 】

本願発明によれば、プレイヤーによって描かれた軌跡を1以上の直線に近似することによって、プレイヤーが複雑な形状の軌跡を描いた場合でも、軌跡ベクトルを算出することができる。また、軌跡を描く1回の操作によって、複数回のアクションを連続してプレイヤーキャラクタに行わせることができる。

## 【 0 0 2 9 】

本願発明によれば、プレイヤーキャラクタが行うアクションのバリエーションをさらに増やすことができ、ゲームの戦略性や興趣性をさらに向上することができる。

40

## 【 0 0 3 0 】

本願発明によれば、プレイヤーによって描かれた軌跡から軌跡ベクトルを容易に算出することができる。

## 【 0 0 3 1 】

本願発明によれば、軌跡に複数の直線部分が含まれる場合であっても近似される各直線を容易に算出することができる。また、1つの軌跡から複数の軌跡ベクトルを容易に算出することができる。

## 【 0 0 3 2 】

本願発明によれば、基準ベクトルの終点の位置を軌跡ベクトルに基づいて容易に決定す

50

ることができる。

【発明を実施するための最良の形態】

【0033】

以下、本発明の一実施形態に係るゲームプログラムおよびゲーム装置について説明する。まず、ゲームプログラムを実行するゲーム装置の一例である、携帯型のゲーム装置の構成について説明する。図1は、ゲーム装置の外観図である。図1において、ゲーム装置10は、第1のLCD(Liquid Crystal Display:液晶表示装置)11および第2のLCD12を含む。ハウジング13は上側ハウジング13aと下側ハウジング13bとによって構成されており、第1のLCD11は上側ハウジング13aに収納され、第2のLCD12は下側ハウジング13bに収納される。第1のLCD11および第2のLCD12の解像度はいずれも256dot×192dotである。なお、第1の実施形態では表示装置としてLCDを用いているが、例えばEL(Electro Luminescence:電界発光)を利用した表示装置など、他の任意の表示装置を利用することができる。また任意の解像度のものを利用することができる。

10

【0034】

上側ハウジング13aには、後述する1対のスピーカ(図2の30)からの音を外部に放出するための音抜き孔18aおよび18bが形成されている。

【0035】

下側ハウジング13bには、入力装置として、十字スイッチ14a、スタートスイッチ14b、セレクトスイッチ14c、Aボタン14d、Bボタン14e、Xボタン14f、Yボタン14g、Lボタン14LおよびRボタン14Rが設けられている。また、さらなる入力装置として、第2のLCD12の画面上にタッチパネル15(図1に示す2点鎖線)が装着されている。また、下側ハウジング13bには、電源スイッチ19や、メモリカード17やスティック16を収納するための挿入口(図1に示す一点鎖線)も設けられている。

20

【0036】

タッチパネル15としては、例えば抵抗膜方式や光学式(赤外線方式)や静電容量結合式など、任意の方式のものを利用することができる。タッチパネル15は、その表面をスティック16で触れると、その接触位置に対応する座標データを出力する機能を有している。タッチパネル15は、プレイヤによる入力位置を所定のサンプリング時間間隔で検出し、上記座標データを出力する。なお、以下ではプレイヤがタッチパネル15をスティック16で操作するものとして説明を行うが、スティック16の代わりにペン(スタイラスペン)や指でタッチパネル15を操作することももちろん可能である。第1の実施形態では、タッチパネル15として、第2のLCD12の解像度と同じく256dot×192dotの解像度(検出精度)のものを利用する。ただし、必ずしもタッチパネル15の解像度と第2のLCD12の解像度が一致している必要はない。

30

【0037】

メモリカード17はゲームプログラムを記録した記録媒体であり、下部ハウジング13bに設けられた挿入口に着脱自在に装着される。

【0038】

次に、図2を参照してゲーム装置10の内部構成を説明する。図2において、ハウジング13に収納される電子回路基板20には、CPUコア21が実装される。CPUコア21には、バス22を介して、コネクタ23が接続されるとともに、入出力インターフェース回路(図面ではI/F回路と記す)25、第1GPU(Graphics Processing Unit)26、第2GPU27、RAM24およびLCDコントローラ31が接続される。コネクタ23には、メモリカード17が着脱自在に接続される。メモリカード17は、ゲームプログラムを記憶するROM171と、バックアップデータを書き換え可能に記憶するRAM172を搭載する。メモリカード17のROM171に記憶されたゲームプログラムはRAM24にロードされ、RAM24にロードされたゲームプログラムがCPUコア21によって実行される。RAM24には、ゲームプログラムの他に

40

50

も、CPUコア21がゲームプログラムを実行して得られる一時的なデータや、ゲーム画像を生成するためのデータが記憶される。I/F回路25には、タッチパネル15と、スピーカ30と、図1の十字スイッチ14aやAボタン14d等からなる操作スイッチ部14とが接続される。スピーカ30は、音抜き孔18aおよび18bの内側に配置される。  
【0039】

第1GPU26には、第1VRAM(Video RAM)28が接続され、第2GPU27には、第2VRAM29が接続される。第1GPU26は、CPUコア21からの指示に応じて、RAM24に記憶されているゲーム画像を生成するためのデータに基づいて第1のゲーム画像を生成し、第1VRAM28に描画する。第2GPU27は、同様にCPUコア21からの指示に応じて第2のゲーム画像を生成し、第2VRAM29に描画する。第1VRAM28および第2VRAM29はLCDコントローラ31に接続されている。

10

【0040】

LCDコントローラ31はレジスタ32を含む。レジスタ32はCPUコア21からの指示に応じて0または1の値を記憶する。LCDコントローラ31は、レジスタ32の値が0の場合は、第1VRAM28に描画された第1のゲーム画像を第1のLCD11に出力し、第2VRAM29に描画された第2のゲーム画像を第2のLCD12に出力する。また、レジスタ32の値が1の場合は、第1VRAM28に描画された第1のゲーム画像を第2のLCD12に出力し、第2VRAM29に描画された第2のゲーム画像を第1のLCD11に出力する。

20

【0041】

なお、上記のようなゲーム装置10の構成は単なる一例に過ぎず、本発明は、少なくとも1つの表示装置を有する任意のコンピュータシステムに適用することができる。また、本発明のゲームプログラムは、メモリカード17などの外部記憶媒体を通じてコンピュータシステムに供給されるだけでなく、有線または無線の通信回線を通じてコンピュータシステムに供給されてもよいし、さらにはコンピュータシステム内部の不揮発性記憶装置に予め記録されていてもよい。

【0042】

以下、ゲーム装置10においてゲームプログラムを実行することによって行われるゲームの概要を説明する。本ゲームにおいては、プレイヤーによって操作されるプレイヤーキャラクタおよび敵キャラクタが3次元の仮想のゲーム空間に登場する。プレイヤーは、プレイヤーキャラクタを操作して敵キャラクタに対して攻撃を行い、敵キャラクタを倒すことを目的としてゲームを行う。以下、本ゲームの詳細について図3～図12を用いて説明する。

30

【0043】

図3は、ゲーム装置10の第2のLCD12に表示されるゲーム画面の例を示す図である。図3に示すように、本実施形態においては、プレイヤーキャラクタ41を含むゲーム空間を所定の仮想カメラから見た画像が第2のLCD12に表示される。本実施形態においては、プレイヤーキャラクタ41は画面の中央に表示されるように仮想カメラの位置や視線方向等のパラメータが設定されるが、他の実施形態においては、プレイヤーキャラクタ41は画面の中央に常に表示されなくてもよい。なお、第1のLCD11は使用されなくてもよいし、プレイヤーキャラクタの体力や能力を示す各種パラメータや、プレイヤーキャラクタが所持するアイテム等の情報が第1のLCD11に表示されてもよい。

40

【0044】

プレイヤーは、タッチパネル15に対する入力(タッチ入力)によって、プレイヤーキャラクタ41を移動させる操作(移動操作)を行うことができる。ゲーム装置10は、プレイヤーによる入力が行われた位置(入力位置)を検出すると、検出した入力位置に基づいてプレイヤーキャラクタ41の移動方向を決定する。そして、当該移動方向へプレイヤーキャラクタ41を移動させる。プレイヤーキャラクタ41は、入力位置の方向へ予め定められた所定速度で移動する。例えば、図3に示すように、プレイヤーがスティック16で点P1の位置を指定した場合、プレイヤーキャラクタ41は、点P1の方向へ所定速度で移動する(図3

50



に示す矢印参照)。

【0045】

また、プレイヤは、タッチパネル15に対する入力によって、プレイヤキャラクタ41に攻撃を行わせる操作(攻撃操作)を行うことができる。つまり、本実施形態では、プレイヤは、プレイヤキャラクタ41に対する各操作(移動操作および攻撃操作)をタッチパネル15を用いて行うことが可能である。さらに、プレイヤは、第2のLCD12の画面上(タッチパネル15の入力面上)に描く軌跡に応じて異なる複数種類の攻撃操作を行うことができる。具体的には、プレイヤは、画面上に直線状の軌跡を描く操作によって4種類の通常攻撃(突き、右斬り、左斬り、および縦斬り)をプレイヤキャラクタに行わせることができ、さらに、直線状の軌跡を連続して描く操作(スクラッチ操作)によって連続攻撃をプレイヤキャラクタに行わせることができる。以下、攻撃操作を詳細に説明する。

10

【0046】

まず、プレイヤキャラクタが通常攻撃を行う際の処理について図4~図10を用いて説明する。図4は、プレイヤが通常攻撃の攻撃操作を行うときのゲーム画面を示す図である。図4においては、プレイヤキャラクタ41および敵キャラクタ42が登場するゲーム空間の画像が第2のLCDに表示されている。また、図4においては、プレイヤキャラクタ41の右側の領域において、軌跡T1がプレイヤによって描かれているとする。なお、プレイヤによって描かれた軌跡を示す画像は表示されてもよいし、表示されなくてもよい。

【0047】

図5は、図4に示す軌跡T1を示す図である。図5に示すように、プレイヤによって描かれる軌跡は、タッチパネル15によって連続して検出される入力位置(図5に示す黒丸)の集合として表される。以下では、連続して検出される入力位置のうちで最初に検出される入力位置(図5に示す点P2)を初期入力位置と呼ぶ。また、タッチパネル15によって入力位置が連続して検出される間において、最新の入力位置(最後に検出された入力位置。図5に示す点P3)を現在入力位置と呼び、現在入力位置の1つ前に検出された入力位置(図5に示す点P4)を前入力位置と呼ぶ。

20

【0048】

なお、本実施形態では、タッチパネル15によって検出される位置は、画面の左下を原点とし、画面の上方向をy軸正方向とし、画面の右方向をx軸正方向とする座標系によって表現される。つまり、タッチパネル15は、当該座標系における座標のデータを出力する。ここで、連続して検出された2つの入力位置を検出順に結ぶベクトルを差分ベクトルと呼ぶ。特に、前入力位置を始点とし現在入力位置を終点とするベクトルを、現在差分ベクトルと呼ぶ。図6は、図5に示す現在入力位置および前入力位置を示す図である。図6に示す例では、現在差分ベクトルは( $x_1$ ,  $y_1$ )として表される。また、現在差分ベクトルの1つ前のベクトルを前差分ベクトルと呼ぶ。前差分ベクトルは、前入力位置の1つ前に検出された入力位置を始点とし、前入力位置を終点とするベクトルである。

30

【0049】

図4の説明に戻り、プレイヤによって軌跡が描かれたとき、ゲーム装置10は、軌跡ベクトルおよび基準ベクトルを算出する。図7は、軌跡ベクトルおよび基準ベクトルを示す図である。なお、図7では、軌跡ベクトルv1および基準ベクトルv2を見やすくする目的でプレイヤキャラクタ41および敵キャラクタ42を点線で示している。また、図7に示す軌跡ベクトルv1および基準ベクトルv2は、実際には画面に表示されない。

40

【0050】

軌跡ベクトルとは、プレイヤによって描かれた軌跡を示すベクトルである。ここでは、当該軌跡を1本の直線に近似することで軌跡ベクトルを算出する。具体的には、軌跡ベクトルは、軌跡の初期入力位置を始点とし、最後に検出された入力位置を終点とするベクトルとなる。

【0051】

なお、詳細は後述するが、軌跡ベクトルは、軌跡の全体を示すベクトルに必ずしもなるわけではなく、連続して入力された軌跡の少なくとも一部を示すベクトルであることもあ

50

る。例えば、上述したスクラッチ操作によって描かれる軌跡が描かれる場合には、当該軌跡の一部の直線部分を示すベクトルが軌跡ベクトルとして算出される。

#### 【 0 0 5 2 】

一方、基準ベクトル $v_2$ は、画面上におけるプレイヤーキャラクタ41の位置を始点とし、軌跡ベクトル $v_1$ によって決められる位置を終点とするベクトルである。ここで、「軌跡ベクトルによって決められる位置」とは、軌跡ベクトルが決まることによって一義的に決められる位置であり、例えば、軌跡ベクトルの始点や、中点や、終点の位置である。図7に示すように、本実施形態では、軌跡ベクトル $v_1$ によって決められる位置は、軌跡ベクトル $v_1$ の終点の位置であるとする。

#### 【 0 0 5 3 】

軌跡ベクトル $v_1$ および基準ベクトル $v_2$ を算出すると、ゲーム装置10は、軌跡ベクトル $v_1$ と基準ベクトル $v_2$ とのなす角度 $\theta_1$ を算出する。そして、算出した角度 $\theta_1$ に基づいてプレイヤーキャラクタの攻撃のアクションを決定する。図8は、プレイヤーキャラクタの攻撃のアクションを決定する方法を説明するための図である。図8では、軌跡ベクトル $v_1$ と基準ベクトル $v_2$ との始点の位置をそろえて記載している。本実施形態においては、軌跡ベクトル $v_1$ と基準ベクトル $v_2$ とのなす角度 $\theta_1$ は、基準ベクトル $v_2$ の方向を基準方向( $0^\circ$ )とし、軌跡ベクトル $v_1$ を右回りに見たときの角度として表されるものとする。 $\theta_1$ のとり得る範囲は、 $0^\circ \leq \theta_1 < 360^\circ$ とする。 $\theta_1$ のとり得る範囲は、攻撃のアクションに応じて、第1～第4範囲 $45^\circ \sim 48^\circ$ の4つに分割される。具体的には、各範囲は次のように設定される。

第1範囲(図8に示す領域45):  $0^\circ \leq \theta_1 < 45^\circ$  または  $315^\circ \leq \theta_1 < 360^\circ$

第2範囲(図8に示す領域46):  $45^\circ \leq \theta_1 < 135^\circ$

第3範囲(図8に示す領域47):  $135^\circ \leq \theta_1 < 225^\circ$

第4範囲(図8に示す領域48):  $225^\circ \leq \theta_1 < 315^\circ$

プレイヤーキャラクタ41から見て基準ベクトル $v_2$ の方向を正面方向としたとき、第1範囲は、軌跡ベクトル $v_1$ がほぼ正面方向を向いている場合における $\theta_1$ の範囲である。また、第2範囲は、軌跡ベクトル $v_1$ が右方向を向いている場合における $\theta_1$ の範囲である。第3範囲は、軌跡ベクトル $v_1$ が左方向を向いている場合における $\theta_1$ の範囲である。第4範囲は、軌跡ベクトル $v_1$ が背面方向を向いている場合における $\theta_1$ の範囲である。

#### 【 0 0 5 4 】

各範囲には、攻撃のアクションが1つずつ対応付けられる。すなわち、第1範囲には「突き」のアクションが対応付けられる。第2範囲には「左斬り」のアクションが対応付けられる。第3範囲には、「縦斬り」のアクションが対応付けられる。第4範囲には、「右斬り」のアクションが対応付けられる。

#### 【 0 0 5 5 】

ゲーム装置10は、算出した $\theta_1$ の値がどの範囲に属するかを判断する。そして、プレイヤーキャラクタが行うべき攻撃のアクションを、算出した $\theta_1$ の値が属する範囲に対応するアクションに決定する。例えば図8に示す場合においては $\theta_1$ は第2範囲に属するので、プレイヤーキャラクタが行うべき攻撃のアクションは、第2範囲に対応する「左斬り」に決定される。

#### 【 0 0 5 6 】

プレイヤーキャラクタの攻撃のアクションを決定すると、ゲーム装置10は、決定したアクションをプレイヤーキャラクタが行う画像を第2のLCDに表示する。図9および図10は、プレイヤーキャラクタが攻撃のアクションを行う際のゲーム画面を示す図である。例えば、上記角度 $\theta_1$ が第1範囲に属する場合、プレイヤーキャラクタ41は、図9に示す「突き」のアクションを行う。また、角度 $\theta_1$ が第4範囲に属する場合、プレイヤーキャラクタ41は、図10に示す「右斬り」のアクションを行う。このように、本実施形態では、プレイヤーが描いた軌跡の方向(軌跡ベクトル)とプレイヤーキャラクタから当該軌跡への方向(基準ベクトル)との関係に応じて、プレイヤーキャラクタが行うアクションが変化する。

## 【 0 0 5 7 】

なお、プレイヤーキャラクタ 4 1 が攻撃のアクションを行う方向は、軌跡ベクトルに基づいて決定される。具体的には、ゲーム装置 1 0 は、軌跡ベクトルの中点の方向に向かってアクションを行うようにプレイヤーキャラクタ 4 1 の動作を制御する。なお、他の実施形態においては、プレイヤーキャラクタ 4 1 は、現在の位置から軌跡ベクトルの始点の方向に向かってアクションを行うようにしてもよいし、軌跡ベクトルの終点の方向に向かってアクションを行うようにしてもよい。なお、軌跡ベクトルの中点または終点の方向に向かってアクションを行う場合には、敵キャラクタを斬りつけるように軌跡を描けば（図 4 参照）、その敵キャラクタに対して攻撃を行うことができる。つまり、この場合、プレイヤーは、スティック 1 6 を使って実際に敵キャラクタを攻撃するような感覚でゲーム操作を行えばよいので、ゲームの操作性を向上することができる。また、本実施形態では、プレイヤーキャラクタ 4 1 は画面の中央に表示されるので、プレイヤーは、プレイヤーキャラクタ 4 1 を基準に上下左右のどの方向にも軌跡を容易に描くことができる。つまり、プレイヤーは、プレイヤーキャラクタ 4 1 の全方向に攻撃を行わせることができる。

10

## 【 0 0 5 8 】

ゲーム装置 1 0 は、プレイヤーキャラクタ 4 1 が攻撃のアクションを行った結果、当該攻撃が敵キャラクタ 4 2 に命中したか否かを判定する。命中したと判定される場合、敵キャラクタ 4 2 にはダメージが与えられる。本実施形態では、敵キャラクタ 4 2 に与えられるダメージは、各アクションに応じて異なる。また、プレイヤーキャラクタ 4 1 が攻撃するまでの時間（攻撃のアクションが開始されてから敵キャラクタに攻撃が命中するまでの時間）は、各アクションに応じて異なる。例えば、「突き」のアクションについては、攻撃するまでの時間が短い（攻撃が速い）代わりに与えるダメージを小さくしたり、「縦斬り」のアクションは、攻撃するまでの時間が長い（攻撃が遅い）代わりに与えるダメージを大きくしたりすることができる。このように、各アクションによって異なるゲーム効果を設定することによって、ゲームの戦略性を向上することができる。

20

## 【 0 0 5 9 】

また、本実施形態では、プレイヤーキャラクタ 4 1 が攻撃のアクションを行っている間、プレイヤーキャラクタ 4 1 の移動動作は行われない。したがって、プレイヤーキャラクタ 4 1 が攻撃のアクションを行っている間にプレイヤーが移動操作を行っても、その移動操作は無視される。

30

## 【 0 0 6 0 】

以上のように、本実施形態では、プレイヤーキャラクタは 4 種類の通常攻撃を行うことが可能であり、プレイヤーは、画面上に描く軌跡を変化させることによって 4 種類の攻撃のアクションを使い分けることができる。これによって、タッチパネル 1 5 のようなポインティングデバイスを用いたゲームにおいても、プレイヤーキャラクタに多彩なアクションを容易に行わせることができる。

## 【 0 0 6 1 】

また、本実施形態では、プレイヤーによって描かれた軌跡をプレイヤーキャラクタ 4 1 から見たとき、当該軌跡が手前から奥に向けて描かれた場合、軌跡ベクトル v 1 は第 1 範囲に属することになる。したがって、この場合、プレイヤーキャラクタ 4 1 は「突き」のアクション（図 9 参照）を行う。また、プレイヤーキャラクタ 4 1 から見て左から右に向けて軌跡が描かれた場合、軌跡ベクトル v 1 は第 2 範囲に属することになり、プレイヤーキャラクタ 4 1 は「左斬り」のアクションを行う。同様に、プレイヤーキャラクタ 4 1 から見て奥から手前に向けて軌跡が描かれた場合、プレイヤーキャラクタ 4 1 は「縦斬り」のアクションを行い、プレイヤーキャラクタ 4 1 から見て右から左に向けて軌跡が描かれた場合、プレイヤーキャラクタ 4 1 は「右斬り」のアクション（図 1 0 参照）を行う。このように、本実施形態では、プレイヤーキャラクタ 4 1 が剣を振る軌道は、プレイヤーが描いた軌跡の軌道に近いものとなる。例えば、プレイヤーは、「突き」で敵キャラクタ 4 2 を攻撃したい場合、スティック 1 6 で敵キャラクタ 4 2 を実際に突くような感覚で、プレイヤーキャラクタ 4 1 から見て手前から奥に向けて軌跡を描けばよい。つまり、プレイヤーキャラクタ 4 1 は、プレイ

40

50

ヤが描いた軌跡のイメージに近いアクションを行うこととなる。したがって、プレイヤは攻撃操作の操作方法をイメージしやすく、プレイヤにとってわかりやすい操作方法を提供することができる。

#### 【0062】

以上に述べたように、本実施形態では、画面上に軌跡を描くことによって、プレイヤは、プレイヤキャラクタに通常攻撃を行わせることができる。ただし、本実施形態では、プレイヤが描いた軌跡が、予め定められた所定の長さ以上であり、かつ、予め定められた所定時間以内に入力されたことを条件とする。つまり、プレイヤが描いた軌跡がこの条件を満たさない場合、ゲーム装置10は、プレイヤキャラクタに通常攻撃を行わせない。なお、この場合には、プレイヤが軌跡を描いた操作は移動操作とみなされ、ゲーム装置10は、入力位置に応じた方向へプレイヤキャラクタを移動させる処理を行う。したがって、プレイヤは、攻撃操作を行う場合、所定長さ以上の直線状の軌跡を所定の速度で描かなければならない。

10

#### 【0063】

上記の条件を設定するのは、プレイヤが行った操作が移動操作であるかそれとも攻撃操作であるかを区別するためである。本実施形態では、移動操作および攻撃操作の両方がタッチパネル15を用いて行われる。そのため、もし上記の条件を設定しなければ、プレイヤが移動操作を行う目的で軌跡を描いた場合でも、当該軌跡を描く操作が攻撃操作とみなされて攻撃のアクションが行われてしまうことがある。そこで、本実施形態では、プレイヤが意図しない攻撃のアクションが行われることを防止するために上記条件を設定している。

20

#### 【0064】

次に、プレイヤキャラクタがスクラッチ操作を行う際の処理について図11および図12を用いて説明する。図11は、プレイヤがスクラッチ操作を行うときのゲーム画面を示す図である。図11においては、プレイヤキャラクタ41の右側の領域において、軌跡T2がプレイヤによって描かれている。軌跡T2は、直線状の軌跡を5つ連続して描いたような折れ線状の形状である。図11に示すように、スクラッチ操作は、直線状の軌跡を複数回反転させるように軌跡を描く操作である。

#### 【0065】

プレイヤによって画面上に軌跡が描かれると、ゲーム装置10は、当該軌跡を1以上の直線に近似する。図12は、プレイヤによって描かれた軌跡の一例を示す図である。図12では、点P5の位置が初期入力位置であり、点P13の位置が最後に検出された入力位置であるとする。ゲーム装置10は、検出された各入力位置を検出された順に結ぶ折れ線について、互いに接続される2つの線分のなす角度が鋭角となる部分を検出する（後述するステップS31）。具体的には、ゲーム装置10は、新たな入力位置が検出される度に、現在差分ベクトルと前差分ベクトルとの角度差を算出する。そして、当該角度差に基づいて、互いに接続される2つの線分のなす角度が鋭角となるか否かを判断する（後述するステップS32）。

30

#### 【0066】

例えば、図12に示す折れ線を例とすると、点P6から点P7までを結ぶ線分（差分ベクトル）と、点P7から点P8までを結ぶ線分とのなす角 $\theta_2$ が鋭角になっている。点P6から点P7までを結ぶ線分と点P7から点P8までを結ぶ線分とのなす角 $\theta_2$ が鋭角になる場合とは、点P6から点P7までを結ぶ差分ベクトルと点P7から点P8までを結ぶ差分ベクトルとの角度差 $\theta_2'$ が、 $90^\circ < \theta_2' < 270^\circ$ の範囲内の値となることである。なお、以下では、上記折れ線の鋭角部分の頂点となる入力位置を反転位置と呼ぶ。

40

#### 【0067】

プレイヤによって描かれた軌跡について反転位置が2つ以上検出されると、ゲーム装置10は、順に検出された2つの反転位置について、1カ所目の反転位置が検出されてから所定時間以内に2カ所目の反転位置が入力されたか否かを判定する（後述するステップS42）。なお、この所定時間は予め定められている。判定の結果、所定時間以内に入力さ

50

れた場合、当該2つの反転位置を結ぶベクトルが軌跡ベクトルとして算出される。図12を例にとって説明すると、点P7の入力位置（反転位置）が検出されてから所定時間以内に点P10の入力位置（反転位置）が入力された場合、点P7から点P10までを結ぶベクトルが軌跡ベクトルとして算出される。同様に、点P10の入力位置が検出されてから所定時間以内に点P12の入力位置が入力された場合、点P10から点P12までを結ぶベクトルが軌跡ベクトルとして算出される。

#### 【0068】

以上のように、軌跡を描く操作がスクラッチ操作と判断されるためには、次に示す2つの条件を満たす必要がある。第1の条件は、プレイヤーによって描かれた軌跡において2つ以上の反転位置が存在することである。第2の条件は、1カ所目の反転位置が検出されてから所定時間以内に2カ所目の反転位置が入力されることである。

10

#### 【0069】

ここで、第1の条件において反転位置を2カ所以上としているのは、スクラッチ操作と移動操作とを区別するためである。もし反転位置が1カ所であった場合にスクラッチ操作と判断すれば、プレイヤーキャラクタを方向転換させる場合における移動操作がスクラッチ操作と判断される場合が生じる。プレイヤーキャラクタを方向転換させる移動操作を行う場合には、プレイヤーによって描かれた軌跡に反転位置が含まれるからである。これを防止するため、本実施形態では、第1の条件において反転位置を2カ所以上としている。

#### 【0070】

また、第2の条件も第1の条件と同様、スクラッチ操作と移動操作とを区別するために設定されるものである。プレイヤーが移動操作を行う際、プレイヤーキャラクタを方向転換させた後しばらくして方向転換を再度行う場合も考えられる。第2の条件を設定することによって、このような場合における操作をスクラッチ操作と判断されることを防止している。

20

#### 【0071】

また、連続攻撃は、プレイヤーによるタッチパネル15への入力が続いている間に開始される。具体的には、ゲーム装置10は、第1および第2の条件が満たされたと判断した時点で、プレイヤーによるタッチパネル15への入力が続いている間であっても、プレイヤーキャラクタに1回の攻撃のアクションを行わせる。そして、プレイヤーによるタッチ入力が続き、第1および第2の条件が再度満たされたと判断されると、プレイヤーキャラクタに再度1回の攻撃のアクションを行わせる。したがって、プレイヤーは、スクラッチ操作を連続して（タッチ入力を継続したままで）行うことによって、プレイヤーキャラクタに連続攻撃を行わせることができる。

30

#### 【0072】

なお、本実施形態においては、連続攻撃におけるアクションは、通常攻撃におけるアクションと同様、軌跡ベクトルと基準ベクトルとのなす角に応じて決定される。また、他の実施形態においては、軌跡ベクトルが同じであっても、連続攻撃におけるアクションと通常攻撃におけるアクションとを異なるようにしてもよい。また、連続攻撃の回数に応じてアクションを変化させてもよい。例えば、所定回数以上の連続攻撃を行った場合には、特別なアクションを行うことができるようにしてもよい。

40

#### 【0073】

次に、ゲームプログラムを実行することによってゲーム装置10において行われるゲーム処理の詳細を説明する。まず、ゲーム処理において用いられる主なデータについて図13を用いて説明する。図13は、ゲーム装置10のRAM24に記憶される主なデータを示す図である。図13に示すように、RAM24には、現在入力位置データ51、前入力位置データ52、初期入力位置データ53、第1反転位置データ54、第2反転位置データ55、およびキャラクタ2次元座標データ56等の座標データが記憶される。また、RAM24には、現在差分ベクトルデータ57、前差分ベクトルデータ58、軌跡ベクトルデータ59、および基準ベクトルデータ60等のベクトルデータが記憶される。また、RAM24には、総フレームカウンタ61、反転フレームカウンタ62、反転フラグ63、

50

および攻撃フラグ 6 4 等のゲーム処理に用いられるデータが記憶される。なお、R A M 2 4 には、図 1 3 に示すデータの他、プレイヤーキャラクタに関するデータ（プレイヤーキャラクタの画像データや位置データ等）やゲーム空間に関するデータ（地形データ等）等、ゲーム処理に必要なデータが記憶される。

【 0 0 7 4 】

現在入力位置データ 5 1 は、上記現在入力位置の座標を示すデータである。現在入力位置データ 5 1 は、最新の入力位置を示し、タッチパネル 1 5 が入力位置を検出する毎に、すなわち、タッチパネル 1 5 から出力される座標データをゲーム装置 1 0 が読み込む毎に更新される。前入力位置データ 5 2 は、上記前入力位置の座標を示すデータである。前入力位置データ 5 2 は、現在入力位置データ 5 1 が更新される度に更新され、更新前の現在入力位置データ 5 1 により示される座標が、更新後の前入力位置データ 5 2 として記憶される。初期入力位置データ 5 3 は、上記初期入力位置の座標を示すデータである。

10

【 0 0 7 5 】

各反転位置データ 5 4 および 5 5 は、上記反転位置の座標を示すデータである。第 1 反転位置データ 5 4 は、最新の反転位置の座標を示し、第 2 反転位置データ 5 5 は、1 つ前の反転位置の座標を示す。例えば図 1 2 に示す軌跡が描かれる場合を例にとって説明すると、点 P 1 1 の位置が現在入力位置である場合、第 1 反転位置データ 5 4 は、点 P 1 0 の反転位置の座標を示し、第 2 反転位置データ 5 5 は、点 P 7 の反転位置の座標を示す。

【 0 0 7 6 】

キャラクタ 2 次元座標データ 5 6 は、プレイヤーキャラクタの画面上における位置（座標）を示すデータである。なお、図 1 3 には示していないが、キャラクタ 2 次元座標データ 5 6 の他に、プレイヤーキャラクタのゲーム空間内における 3 次元座標を示すデータも R A M 2 4 に記憶されている。

20

【 0 0 7 7 】

現在差分ベクトルデータ 5 7 は、上記現在差分ベクトルを示すデータである。現在差分ベクトルデータ 5 7 は、最新の差分ベクトルを示し、タッチパネル 1 5 が入力位置を検出する毎に更新される。前差分ベクトルデータ 5 8 は、上記前差分ベクトルを示すデータである。前差分ベクトルデータ 5 8 は、現在差分ベクトルデータ 5 7 が更新される度に更新され、更新前の現在差分ベクトルデータ 5 7 により示される座標が、更新後の前差分ベクトルデータ 5 8 として記憶される。また、軌跡ベクトルデータ 5 9 は、上記軌跡ベクトルを示すデータである。基準ベクトルデータ 6 0 は、上記基準ベクトルを示すデータである。

30

【 0 0 7 8 】

総フレームカウンタ 6 1 は、プレイヤーによるタッチ入力開始されてからのフレーム数を示すカウンタである。反転フレームカウンタ 6 2 は、プレイヤーにより描かれた軌跡が反転してからのフレーム数を示すカウンタである。図 1 2 を用いて説明すると、ゲーム装置 1 0 は、点 P 8 の入力位置が検出されると、プレイヤーにより描かれた軌跡が反転したと判断して、反転フレームカウンタ 6 2 による計測を開始する。また、軌跡の反転が再度行われる（図 1 2 に示す点 P 1 1 の入力位置が検出されると）、ゲーム装置 1 0 は、反転フレームカウンタ 6 2 の値をリセットし、再度最初から計測を行う。

40

【 0 0 7 9 】

また、反転フラグ 6 3 は、軌跡の反転が行われたか否かを示すフラグである。すなわち、軌跡の反転が行われたとき、反転フラグ 6 3 は“ O N ”に設定され、軌跡の反転が行われていないとき、反転フラグ 6 3 は O F F に設定される。図 1 2 を用いて説明すると、ゲーム装置 1 0 は、点 P 8 の入力位置が検出されると、プレイヤーにより描かれた軌跡が反転したと判断して、反転フラグ 6 3 を“ O N ”に設定する。なお、初期状態では反転フラグ 6 3 は“ O F F ”に設定されている。また、攻撃フラグ 6 4 は、プレイヤーキャラクタが攻撃のアクションを行っている途中であるか否かを示すフラグである。すなわち、プレイヤーキャラクタが攻撃のアクションを開始すると、攻撃フラグ 6 4 は“ O N ”に設定され、当該アクションが終了すると、攻撃フラグ 6 4 は O F F に設定される。

50

## 【 0 0 8 0 】

次に、ゲームプログラムを実行することによってゲーム装置 1 0 において行われるゲーム処理の詳細を、図 1 4 ~ 図 1 9 を用いて説明する。図 1 4 は、ゲーム装置 1 0 において実行されるゲーム処理の流れを示すフローチャートである。ゲーム装置 1 0 の電源が投入されると、ゲーム装置 1 0 の C P U コア 2 1 は、図示しないブート R O M に記憶されている起動プログラムを実行し、R A M 2 4 等の各ユニットが初期化される。そして、メモリカード 1 7 に格納されたゲームプログラムが R A M 2 4 に読み込まれ、C P U コア 2 1 によって当該ゲームプログラムの実行が開始される。図 1 4 に示すフローチャートは、以上の処理が完了した後に行われるゲーム処理を示すフローチャートである。なお、図 1 4 ~ 図 1 9 においては、ゲーム処理のうち、プレイヤーキャラクタに対する移動操作および攻撃操作に関する処理について詳細に示し、本願発明と関連しない他のゲーム処理については詳細な説明を省略する。

10

## 【 0 0 8 1 】

図 1 4 のステップ S 1 において、まず、ゲーム空間が構築され、プレイヤーキャラクタや敵キャラクタが初期位置に配置される。そして、プレイヤーキャラクタとその周囲のゲーム空間を撮影する位置に仮想カメラが設定され、仮想カメラから見たゲーム空間の画像が第 2 の L C D に表示される。このステップ S 1 以降の処理において、タッチパネル等を用いたゲーム操作が受け付けられ、ゲームが進行する。ステップ S 1 の次にステップ S 2 の処理が行われる。

## 【 0 0 8 2 】

20

ステップ S 2 において、タッチ入力があったか否かが判定される。具体的には、C P U コア 2 1 は、タッチパネル 1 5 から出力されるデータを読み込み、タッチパネル 1 5 から座標データが出力されているか否かを判定する。タッチパネル 1 5 から座標データが出力されている場合、タッチ入力があると判定されてステップ S 3 の処理が行われる。一方、入力がないことを示すデータがタッチパネル 1 5 から出力されている場合、タッチ入力がないと判定されて再度ステップ S 2 の処理が行われる。つまり、C P U コア 2 1 は、タッチ入力があるまでステップ S 2 の処理を繰り返し、タッチ入力があるとステップ S 3 の処理を行う。

## 【 0 0 8 3 】

ステップ S 3 において、各データおよび各フレームカウンタが初期化される。具体的には、R A M 2 4 に記憶されている各データ（各座標データ 5 1 ~ 5 6 および各ベクトルデータ 5 7 ~ 6 0 ）および各フレームカウンタ 6 1 および 6 2 の内容が初期化される。続くステップ S 4 において、R A M 2 4 に記憶されている反転フラグ 6 3 の値が“ O F F ”に設定される。ステップ S 4 の次にステップ S 5 の処理が行われる。

30

## 【 0 0 8 4 】

ステップ S 5 において、タッチパネル 1 5 に対する入力位置が検出される。具体的には、C P U コア 2 1 は、タッチパネル 1 5 から入力位置の座標データを取得し、取得した座標データを現在入力位置データ 5 1 として R A M 2 4 に記憶する。続くステップ S 6 において、ステップ S 5 で検出された入力位置が初期入力位置に設定される。具体的には、C P U コア 2 1 は、ステップ S 5 で取得した座標データを初期入力位置データ 5 3 として R A M 2 4 に記憶する。続くステップ S 7 において、ステップ S 5 で取得された座標データを前入力位置データ 5 2 として R A M 2 4 に記憶する。ステップ S 7 の処理は、後述するステップ S 1 3 において現在差分ベクトルを算出する際に必要となる前入力位置データ 5 2 を記憶しておく処理である。ステップ S 7 の次にステップ S 8 の処理が行われる。

40

## 【 0 0 8 5 】

ステップ S 8 においては、メイン処理が行われる。メイン処理では、タッチパネル 1 5 に対する入力位置が検出され、検出された入力位置に基づいてプレイヤーキャラクタが移動したり攻撃のアクションを行ったりする。以下、メイン処理の詳細を図 1 5 ~ 図 1 9 を用いて説明する。

## 【 0 0 8 6 】

50

図 1 5 は、図 1 4 に示すステップ S 8 の詳細な処理の流れを示すフローチャートである。メイン処理においてはまずステップ S 1 1 において、タッチパネル 1 5 に対する入力位置が検出される。ステップ S 1 1 の処理は上記ステップ S 5 の処理と同様である。なお、ステップ S 1 1 において、現在入力位置データ 5 1 は R A M 2 4 に更新して記憶される。また、メイン処理においては、1 フレーム毎にステップ S 1 1 の処理が行われる。ステップ S 1 1 の次にステップ S 1 2 の処理が行われる。

【 0 0 8 7 】

ステップ S 1 2 においては、R A M 2 4 に記憶されている現在差分ベクトルデータ 5 7 が前差分ベクトルデータ 5 8 に代入される。つまり、R A M 2 4 に記憶されている現在差分ベクトルデータ 5 7 の内容が前差分ベクトルデータ 5 8 として記憶される。続くステップ S 1 3 において、現在差分ベクトルが新たに算出される。上述のように、現在差分ベクトルは、前入力位置を始点とし現在入力位置を終点とするベクトルである（図 6 参照）。現在差分ベクトルは、R A M 2 4 に記憶されている現在入力位置データ 5 1 および前入力位置データ 5 2 を用いて算出される。続くステップ S 1 4 において、R A M 2 4 に記憶されている総フレームカウンタ 6 1 の値が 1 加算される。

【 0 0 8 8 】

ステップ S 1 4 の次のステップ S 1 5 において、第 1 行動判定処理が行われる。第 1 行動判定処理は、プレイヤーによるタッチ入力に基づいてプレイヤーの操作が移動操作であるか攻撃操作（スクラッチ操作）であるかを判断し、プレイヤーの操作に従ってプレイヤーキャラクタを動作させる処理である。以下、図 1 6 ~ 図 1 8 を用いて第 1 行動判定処理の詳細を説明する。

【 0 0 8 9 】

図 1 6 は、図 1 5 に示すステップ S 1 5 の詳細な処理の流れを示すフローチャートである。第 1 行動判定処理においてはまずステップ S 3 1 において、現在差分ベクトルと前差分ベクトルとの角度差が算出される。当該角度差は、R A M 2 4 に記憶されている現在差分ベクトルデータ 5 7 および前差分ベクトルデータ 5 8 を用いて算出される。続くステップ S 3 2 において、ステップ S 3 1 で算出された角度差が所定範囲内の値をとるか否かが判定される。この所定範囲は、現在差分ベクトルと前差分ベクトルとの角度差が、 $90^{\circ} < \angle < 270^{\circ}$  となる範囲である。当該角度差が  $90^{\circ} < \angle < 270^{\circ}$  となる場合、プレイヤーによって描かれた軌跡を表す各入力位置を検出順に結んだ折れ線は、鋭角の部分

【 0 0 9 0 】

ステップ S 3 2 の判定において、上記角度差が所定範囲内の値をとると判定される場合、ステップ S 3 5 の処理が行われる。ステップ S 3 5 以降の処理においては、所定の条件が満たされればプレイヤーによる操作がスクラッチ操作であると判断され、プレイヤーキャラクタの攻撃処理が行われる（後述するステップ S 4 4）。なお、所定の条件が満たされなければプレイヤーによる操作が移動操作であると判断され、プレイヤーキャラクタの移動処理が行われる（後述するステップ S 3 4 または S 4 1）。一方、ステップ S 3 2 の判定において、上記角度差が所定範囲内の値をとらないと判定される場合、ステップ S 3 3 の処理が行われる。ステップ S 3 3 以降の処理においては、プレイヤーキャラクタの移動処理が行われる（後述するステップ S 3 4 または S 4 1）。

【 0 0 9 1 】

ステップ S 3 3 においては、R A M 2 4 に記憶されている反転フラグ 6 3 が “ O N ” に設定されているか否かが判定される。ステップ S 3 3 の処理は、現在入力中の軌跡が反転位置を有するか否かを判定するための処理である。例えば図 1 2 に示す軌跡が描かれる場



合を例にとって説明すると、点 P 8 の位置が現在入力位置である時点で反転フラグ 6 3 が “ O N ” に設定される（後述するステップ S 3 7）。したがって、点 P 8 よりも後の位置が現在入力位置である場合には、ステップ S 3 3 の判定において、反転フラグ 6 3 が “ O N ” に設定されていると判定される。ステップ S 3 3 の判定において、反転フラグ 6 3 が “ O N ” に設定されていると判定される場合、後述するステップ S 3 8 の処理が行われる。一方、反転フラグ 6 3 が “ O N ” に設定されていない（すなわち、“ O F F ” に設定されている）と判定される場合、ステップ S 3 4 の処理が行われる。

【 0 0 9 2 】

ステップ S 3 4 においては、移動処理が行われる。移動処理は、ゲーム空間内においてプレイヤーキャラクタを移動させる処理である。以下、図 1 7 を用いて移動処理の詳細を説明する。

10

【 0 0 9 3 】

図 1 7 は、図 1 6 に示すステップ S 3 4 の詳細な処理の流れを示すフローチャートである。移動処理においてはまずステップ S 5 0 において、プレイヤーキャラクタが攻撃のアクションを行っている途中であるか否かが判定される。具体的には、R A M 2 4 に記憶されている攻撃フラグ 6 4 が “ O N ” に設定されている場合、プレイヤーキャラクタが攻撃のアクションを行っている途中であると判定される。この場合、ステップ S 5 1 ~ 5 4 の処理はスキップされ、C P U コア 2 1 は移動処理を終了する。したがって、本実施形態では、プレイヤーキャラクタが攻撃のアクションを行っていないことを条件として、プレイヤーキャラクタに移動動作を行わせる処理（ステップ S 5 1 ~ 5 4）が行われる。一方、R A M 2 4 に記憶されている攻撃フラグ 6 4 が “ O F F ” に設定されている場合、プレイヤーキャラクタが攻撃のアクションを行っている途中ではないと判定される。この場合、ステップ S 5 1 の処理が行われる。

20

【 0 0 9 4 】

ステップ S 5 1 において、画面上におけるプレイヤーキャラクタの位置を示す 2 次元座標が算出される。具体的には、C P U コア 2 1 は、プレイヤーキャラクタのゲーム空間における位置を示す 3 次元座標を仮想カメラの投影面の 2 次元座標へ変換し、当該 2 次元座標からタッチパネル 1 5 上の座標系の 2 次元座標を算出する。算出された座標のデータは、キャラクター 2 次元座標データ 5 6 として R A M 2 4 に記憶される。ステップ S 5 1 の次にステップ S 5 2 の処理が行われる。

30

【 0 0 9 5 】

ステップ S 5 2 において、ステップ S 5 1 で算出されたプレイヤーキャラクタの 2 次元座標を始点とし、現在入力位置を終点とするベクトルが算出される。このベクトルは、R A M 2 4 に記憶されている現在入力位置データ 5 1 およびキャラクター 2 次元座標データ 5 6 を用いて算出される。続くステップ S 5 3 において、ステップ S 5 2 で算出されたベクトルの角度が算出される。具体的には、予め定められた基準方向を 0 ° としたときにおける当該ベクトルの角度が算出される。続くステップ S 5 4 において、ステップ S 5 3 で算出された角度に基づいてプレイヤーキャラクタがゲーム空間において移動される（図 3 参照）。具体的には、C P U コア 2 1 は、ステップ S 5 3 で算出された角度に基づいてプレイヤーキャラクタの移動方向を決定する。プレイヤーキャラクタは、決定された移動方向へ所定距離だけ移動する。なお、この所定距離は、1 フレームの間にプレイヤーキャラクタが移動可能な距離である。所定距離は、予め定められた一定値であってもよいし、ステップ S 5 2 で算出されたベクトルの大きさに応じて決定されてもよい。以上に示したステップ S 5 1 ~ S 5 4 の処理が終了すると、C P U コア 2 1 は移動処理を終了する。

40

【 0 0 9 6 】

図 1 6 の説明に戻り、ステップ S 3 4 の処理が終了すると、C P U コア 2 1 は第 1 行動判定処理を終了する。以上のように、プレイヤーの描いた軌跡に反転位置がない場合（ステップ S 3 2 の判定結果が否定である場合）、プレイヤーキャラクタはゲーム空間内を移動する動作を行う。

【 0 0 9 7 】

50

一方、上述したように、プレイヤーの描いた軌跡に反転位置がある場合（ステップS32の判定結果が肯定である場合）、ステップS35の処理が行われる。ステップS35においては、RAM24に記憶されている各反転位置データ54および55が更新される。具体的には、RAM24に記憶されている第1反転位置データ54の内容を第2反転位置データ55として記憶し、前入力位置データ52の内容を新たな第1反転位置データ54として記憶する。ステップS35の次にステップS36の処理が行われる。

【0098】

ステップS36においては、RAM24に記憶されている反転フラグ63が“ON”に設定されているか否かが判定される。ステップS36の処理はステップS33の処理と同じである。ステップS36の判定において、反転フラグ63が“ON”に設定されていると判定される場合、後述するステップS42の処理が行われる。一方、反転フラグ63が“ON”に設定されていない（すなわち、“OFF”に設定されている）と判定される場合、ステップS37の処理が行われる。例えば図12に示す軌跡が描かれる場合を例にとって説明すると、点P11の位置が現在入力位置である場合、ステップS32の判定結果が肯定となり、ステップS36の判定結果が肯定となる。したがって、この場合、ステップS42の処理へ進む。また、点P8の位置が現在入力位置である場合、ステップS32の判定結果が肯定となり、ステップS36の判定結果が否定となる。したがって、この場合、ステップS37の処理へ進む。

【0099】

ステップS37においては、RAM24に記憶されている反転フラグ63が“ON”に設定される。続くステップS38においては、RAM24に記憶されている反転フレームカウンタ62の値が1加算される。続くステップS39において、反転フレームカウンタ62の値が予め定められた所定値以上であるか否かが判定される。ステップS39の処理は、スクラッチ操作の上記第2の条件が満たされているか否かを判定するための処理である。反転フレームカウンタ62の値が上記所定値以上であると判定される場合、ステップS40の処理が行われる。一方、反転フレームカウンタ62の値が上記所定値よりも小さいと判定される場合、上記ステップS34の移動処理が行われる。

【0100】

ステップS40においては、RAM24に記憶されている反転フラグ63が“OFF”に設定される。ステップS40の処理が行われる場合には、スクラッチ操作の上記第2の条件が満たされていないからである。続くステップS41において、移動処理が行われる。したがって、スクラッチ操作の上記第2の条件が満たされていない場合には、プレイヤーによる操作は移動操作と判断される。なお、ステップS41の移動処理は、上記ステップS34の移動処理と同じである。ステップS41の次に、後述するステップS45の処理が行われる。

【0101】

一方、ステップS42においては、反転フレームカウンタ62の値が予め定められた所定値以上であるか否かが判定される。ステップS42の処理は、上記ステップS39の処理と同じである。反転フレームカウンタ62の値が上記所定値以上であると判定される場合、上記ステップS40の処理が行われる。一方、反転フレームカウンタ62の値が上記所定値よりも小さいと判定される場合、ステップS43の処理が行われる。

【0102】

ステップS43においては、軌跡ベクトルが算出される。この軌跡ベクトルは、上記第2反転位置を始点とし、上記第1反転位置を終点とするベクトルである。したがって、この軌跡ベクトルは、RAM24に記憶されている各反転位置データ54および55を用いて算出することができる。ステップS43の次にステップS44の処理が行われる。

【0103】

ステップS44においては、攻撃処理が行われる。攻撃処理は、ゲーム空間内においてプレイヤーキャラクタに攻撃のアクションを行わせる処理である。以下、図18を用いて攻撃処理の詳細を説明する。

## 【0104】

図18は、図16に示すステップS44の詳細な処理の流れを示すフローチャートである。攻撃処理においてはまずステップS61において、プレイヤーキャラクタの画面上における位置を示す2次元座標が算出される。ステップS61の処理は、図17の上記ステップS51の処理と同じである。続くステップS62において、基準ベクトルが算出される。上述したように、基準ベクトルは、ステップS51で算出されたプレイヤーキャラクタの2次元座標を始点とし、現在入力位置を終点とするベクトルである。したがって、基準ベクトルは、RAM24に記憶されている現在入力位置データ51およびキャラクタ2次元座標データ56を用いて算出される。算出された基準ベクトルのデータは、基準ベクトルデータ60としてRAM24に記憶される。ステップS62の次にステップS63が行われる。

10

## 【0105】

ステップS63においては、軌跡ベクトルと基準ベクトルとのなす角度が算出される。この角度は、RAM24に記憶されている軌跡ベクトルデータ59および基準ベクトルデータ60を用いて算出される。続くステップS64において、ステップS63で算出された角度に基づいて、プレイヤーキャラクタが行うべきアクションが決定される。具体的には、ステップS63で算出された角度が上記第1～第4範囲(図8参照)のいずれに属するかによって、プレイヤーキャラクタが行うべきアクションが決定される。

## 【0106】

続くステップS65において、CPUコア21は、ステップS64で決定されたアクションをプレイヤーキャラクタに実行させる。すなわち、プレイヤーキャラクタが当該アクションを行うゲーム画像を第2のLCDに表示させる。なお、プレイヤーキャラクタが行うアクションのアニメーションは、複数のフレームにわたって表示される。ここで、ステップS65の時点において、以前に行われた攻撃のアクションがまだ終了していない場合(すなわち、攻撃フラグ64が“ON”の場合)、CPUコア21は、当該アクションが終了した後に続けて、直前のステップS64で決定されたアクションをプレイヤーキャラクタに実行させるようにしてもよい。これによって、プレイヤーキャラクタの攻撃のアクション中においてプレイヤーによってさらに攻撃操作が行われた場合、連続して攻撃のアクションが行われることになる。

20

## 【0107】

続くステップS66において、RAM24に記憶されている攻撃フラグ64の内容が“ON”に設定される。ステップS66の後、CPUコア21は攻撃処理を終了する。

30

## 【0108】

図16の説明に戻り、ステップS44の攻撃処理の次に、ステップS45の処理が行われる。すなわち、ステップS45においては、RAM24に記憶されている反転フレームカウンタ62の値が初期化される(すなわち、0に設定される)。ステップS45の後、CPUコア21は第1行動判定処理を終了する。以上で、第1行動判定処理の説明を終了する。

## 【0109】

図15の説明に戻り、ステップS15の行動判定処理の次にステップS16の処理が行われる。すなわち、ステップS16において、CPUコア21は、RAM24に記憶されている現在入力位置データ51の内容を前入力位置データ52としてRAM24に記憶する。ステップS16の処理は、上記ステップS7の処理と同じ処理である。ステップS16の次にステップS17の処理が行われる。

40

## 【0110】

ステップS17においては、プレイヤーキャラクタの攻撃アクションが終了したか否かが判定される。すなわち、上記攻撃処理のステップS65において開始された攻撃アクションのアニメーションが終了したか否かが判定される。当該アニメーションの最後のフレームが画面に表示されている場合、プレイヤーキャラクタの攻撃アクションが終了したと判定され、ステップS18およびS19の処理が行われる。一方、当該アニメーションがまだ

50

終了していない場合、または、当該アニメーションが表示されていない場合には、ステップS 18およびS 19の処理がスキップされてステップS 20の処理が行われる。

【0111】

ステップS 18においては、CPUコア21は、敵キャラクタのダメージを計算する処理を行う。つまり、上記ステップS 65で実行されたアクションに応じたダメージが敵キャラクタに与えられる。具体的には、まず、CPUコア21は、ステップS 65において実行されたアクションによる攻撃が敵キャラクタに命中したか否かを判定し、命中した場合にはどの敵キャラクタに命中したかを検出する。当該攻撃が命中していない場合は、ステップS 18の処理を終了する。ここで、RAM 24には、プレイヤーキャラクタのアクションと、そのアクションによる敵キャラクタのダメージとを対応付けたテーブルが記憶されている。上記攻撃が命中した場合、CPUコア21は、当該テーブルを参照して、ステップS 65で実行されたアクションに対応するダメージを敵キャラクタに与える。すなわち、敵キャラクタの体力を示すパラメータをダメージに応じて減少させる。以上のステップS 18の後、ステップS 19の処理が行われる。ステップS 19においては、RAM 24に記憶されている攻撃フラグ64が“OFF”に設定される。ステップS 19の後、ステップS 20の処理が行われる。

10

【0112】

ステップS 20においては、プレイヤーがタッチオフしたか否か、すなわち、タッチ入力が無くなったか否かが判定される。ステップS 20の判定は、ステップS 3と同様の方法で行われる。すなわち、CPUコア21は、タッチパネル15から出力されるデータを読み込み、タッチパネル15から座標データが出力されているか否かを判定する。タッチパネル15から座標データが出力されている場合、タッチ入力があると判定されてステップS 11の処理が行われる。以降、タッチ入力がないと判定されるまでステップS 11～S 20の処理が繰り返される。なお、ステップS 20の次のステップS 11では、ステップS 20で読み込まれた座標データが現在入力位置データ51としてRAM 24に記憶される。一方、ステップS 20の判定において、入力がないことを示すデータがタッチパネル15から出力されている場合、タッチ入力が無くなったと判定されてステップS 21の処理が行われる。

20

【0113】

ステップS 21においては、第2行動判定処理が行われる。第2行動判定処理は、直前のステップS 11～S 20のループにおいて検出されたプレイヤーの操作が通常攻撃の攻撃操作であるか否かを判定し、通常攻撃の攻撃操作である場合にはプレイヤーキャラクタに通常攻撃を行わせる処理である。以下、図19を用いて第2行動判定処理の詳細を説明する。

30

【0114】

図19は、図15に示すステップS 21の詳細な処理の流れを示すフローチャートである。第2行動判定処理においてはまずステップS 71において、RAM 24に記憶されている総フレームカウンタ61の値が、予め定められた所定値以下であるか否かが判定される。総フレームカウンタ61の値が所定値以下である場合、ステップS 72の処理が行われる。一方、総フレームカウンタ61の値が所定値よりも大きい場合、CPUコア21は第2行動判定処理を終了する。ステップS 71の処理は、直前のステップS 11～S 20においてプレイヤーによって描かれた軌跡が、予め定められた所定時間以内に入力されたか否かを判定するための処理である。

40

【0115】

ステップS 72においては、軌跡ベクトルが算出される。この軌跡ベクトルは、ステップS 11～S 20においてプレイヤーによって描かれた軌跡の始点から終点までを結ぶベクトルである。すなわち、軌跡ベクトルは、初期入力位置を始点とし、最後の入力位置を終点とするベクトルである。したがって、CPUコア21は、RAM 24に記憶されている初期入力位置データ53および現在入力位置データ51を用いて軌跡ベクトルを算出する。

50

## 【0116】

続くステップS73においては、ステップS72で算出された軌跡ベクトルの大きさが、予め定められた所定値以上であるか否かが判定される。軌跡ベクトルの大きさが所定値以上である場合、ステップS74の処理が行われる。一方、軌跡ベクトルの大きさが所定値よりも小さい場合、CPUコア21は第2行動判定処理を終了する。ステップS73の処理は、直前のステップS11～S20においてプレイヤーによって描かれた軌跡が、予め定められた所定の長さ以上であるか否かを判定するための処理である。

## 【0117】

ステップS74においては、攻撃処理が行われる。ステップS74の攻撃処理は、上記ステップS34の攻撃処理と同じである。ただし、第1行動判定処理における攻撃処理と、第2行動判定処理における攻撃処理とで、プレイヤーキャラクタが行うアクションを異なるようにしてもよい。続くステップS75において、CPUコア21は、敵キャラクタのダメージを計算する処理を行う。ステップS75の処理は上記ステップS18の処理と同じである。続くステップS76において、ステップS19と同様のステップS77の処理が行われる。ステップS77の後、CPUコア21は、図15に示す第2行動判定処理を終了する。また、図15において、ステップS18の第2行動判定処理が終了すると、CPUコア21はメイン処理を終了する。

## 【0118】

図14の説明に戻り、ステップS8のメイン処理の次にステップS9の処理が行われる。すなわち、ステップS9において、ゲームを終了するか否かが判定される。ステップS9の判定は、例えば、プレイヤーキャラクタの体力が0になったか否かや、プレイヤーキャラクタが全ての敵キャラクタを倒したか否か等によって行われる。ステップS9の判定において、ゲームを終了すると判定された場合、CPUコア21は、図14に示すゲーム処理を終了する。一方、ゲームを終了しないと判定された場合、ステップS3の処理が行われ、以降、ステップS9においてゲームを終了すると判定されるまでステップS3～S9の一連の処理が繰り返される。以上で、ゲーム処理の詳細な説明を終了する。

## 【0119】

なお、以上に説明したゲーム処理においては、敵キャラクタの動作の制御については特に触れていないが、CPUコア21は、1フレーム毎に敵キャラクタの動作を制御する処理を行う。すなわち、予め定められたアルゴリズムに従って各敵キャラクタを移動させたり、攻撃させたりする。さらに、ゲーム処理中においては、敵キャラクタによるプレイヤーキャラクタへの攻撃が命中したか否かが判定され、命中した場合には攻撃に応じたダメージをプレイヤーキャラクタに与える処理が行われる。

## 【0120】

また、以上に説明したゲーム処理において、ある入力位置とその次に検出された入力位置との距離が所定長さよりも短い場合、当該次に検出された入力位置を無視するようにしてもよい。具体的には、ステップS13において算出される現在差分ベクトルの大きさが、予め定められた所定長さよりも小さい場合、ステップS16の処理を行わないようにしてもよい。また、この場合、現在差分ベクトルデータは更新されず、直前のステップS12で更新された前差分ベクトルデータは更新前の内容に戻される。このような短い差分ベクトルがある場合、現在差分ベクトルと前差分ベクトルとの角度差を算出する処理(ステップS31)において反転位置を正確に選出することができないおそれがあるからである。

## 【0121】

以上のように、本実施形態によれば、プレイヤーキャラクタは、軌跡ベクトルと基準ベクトルとの角度に応じて異なる攻撃アクションを行う。したがって、プレイヤーは、タッチパネル15を用いてプレイヤーキャラクタに多彩なアクションを容易に行わせることができる。

## 【0122】

なお、本実施形態においては、画面上の任意の位置を指定することができるタッチパネ

10

20

30

40

50

ルを入力装置として用いて、プレイヤーキャラクタの表示位置を基準ベクトルの始点位置とした。ここで、他の実施形態では、入力装置は、タッチパッドのように、入力面上の任意の位置を指定することができる装置であってもよい。このとき、基準ベクトルの始点位置は、タッチパッドの入力面上における予め定められた所定位置（例えば、入力面の中心位置）に設定される。

【産業上の利用可能性】

【0123】

以上のように、本発明は、ポインティングデバイスの入力を用いたゲームにおいて、プレイヤーキャラクタに多彩なアクションを行わせること等を目的としてゲームプログラムおよびゲーム装置に利用することが可能である。

10

【図面の簡単な説明】

【0124】

【図1】本発明の一実施形態に係るゲーム装置の外観図

【図2】ゲーム装置の内部構成図

【図3】ゲーム装置10の第2のLCD12に表示されるゲーム画面の例を示す図

【図4】プレイヤーが通常攻撃の攻撃操作を行うときのゲーム画面を示す図

【図5】図4に示す軌跡T1を示す図

【図6】図5に示す現在入力位置および前入力位置を示す図

【図7】軌跡ベクトルおよび基準ベクトルを示す図

【図8】プレイヤーキャラクタの攻撃のアクションを決定する方法を説明するための図

20

【図9】プレイヤーキャラクタが「突き」の攻撃アクションを行う際のゲーム画面を示す図  
【図10】プレイヤーキャラクタが「右斬り」の攻撃アクションを行う際のゲーム画面を示す図

【図11】プレイヤーがスクラッチ操作を行うときのゲーム画面を示す図

【図12】プレイヤーによって描かれた軌跡の一例を示す図

【図13】ゲーム装置10のRAM24に記憶される主なデータを示す図

【図14】ゲーム装置10において実行されるゲーム処理の流れを示すフローチャート

【図15】図14に示すステップS8の詳細な処理の流れを示すフローチャート

【図16】図15に示すステップS15の詳細な処理の流れを示すフローチャート

【図17】図16に示すステップS34の詳細な処理の流れを示すフローチャート

30

【図18】図16に示すステップS44の詳細な処理の流れを示すフローチャート

【図19】図15に示すステップS21の詳細な処理の流れを示すフローチャート

【符号の説明】

【0125】

10 ゲーム装置

11, 12 LCD

21 CPUコア

24 RAM

41 プレイヤーキャラクタ

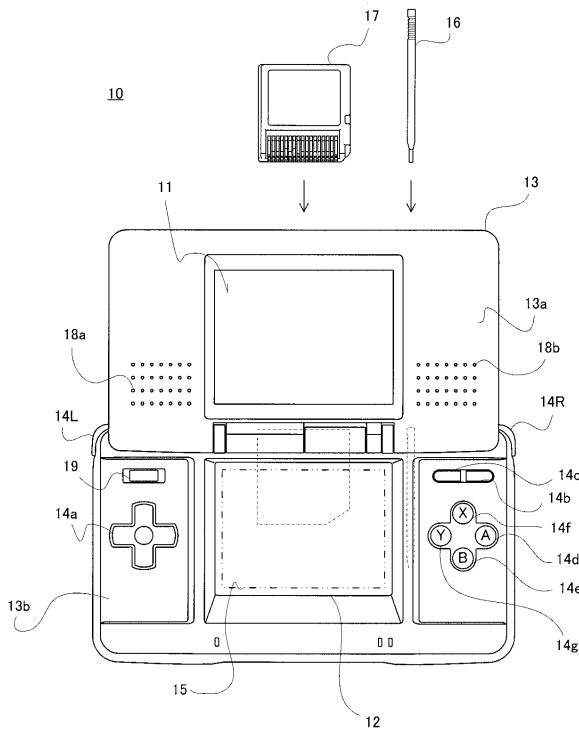
42 敵キャラクタ

40

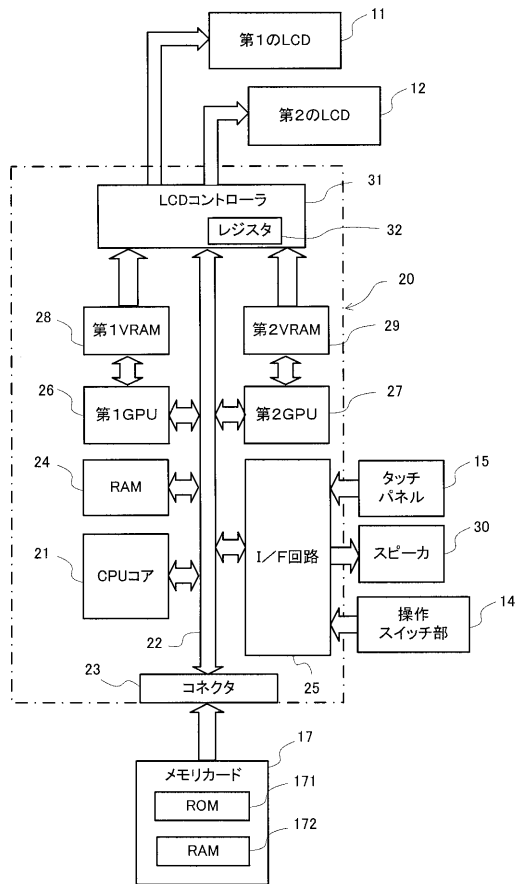
59 軌跡ベクトルデータ

60 基準ベクトルデータ

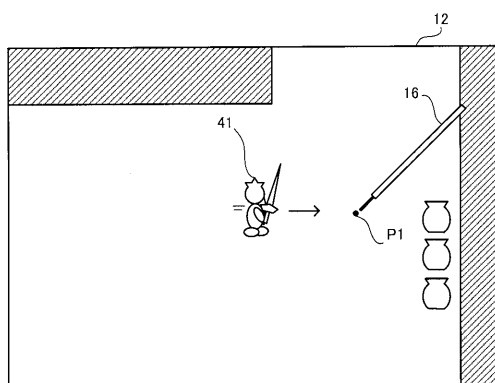
【 図 1 】



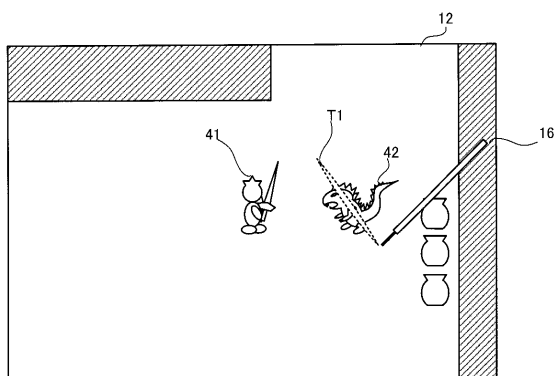
【 図 2 】



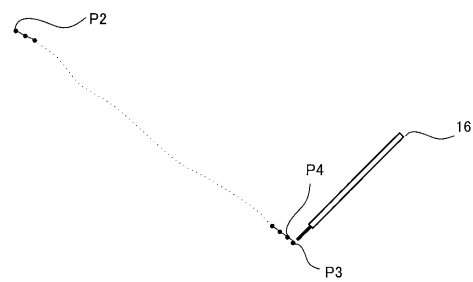
【圖 3】



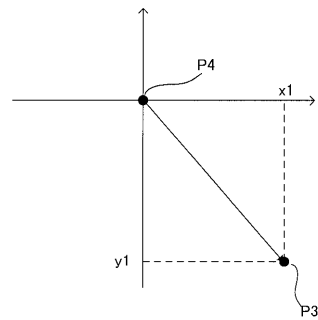
【 図 4 】



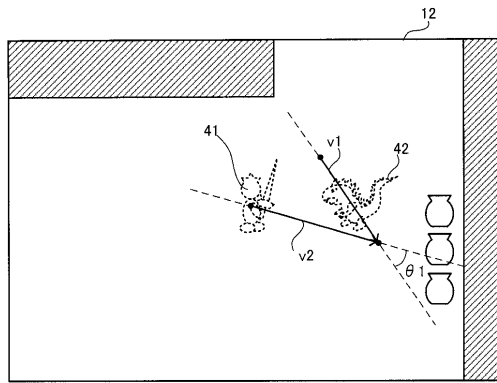
【 図 5 】



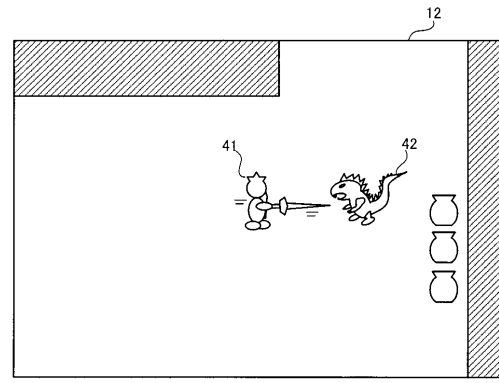
【 図 6 】



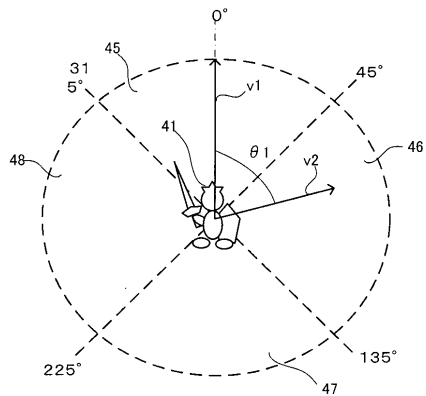
【図 7】



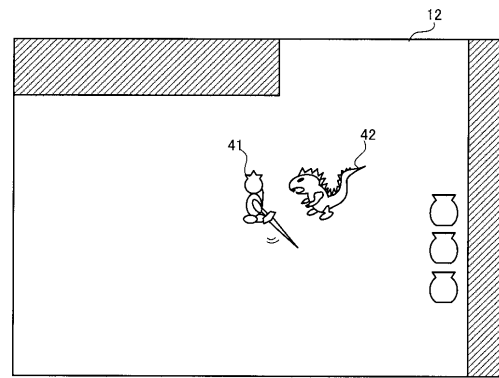
【図 9】



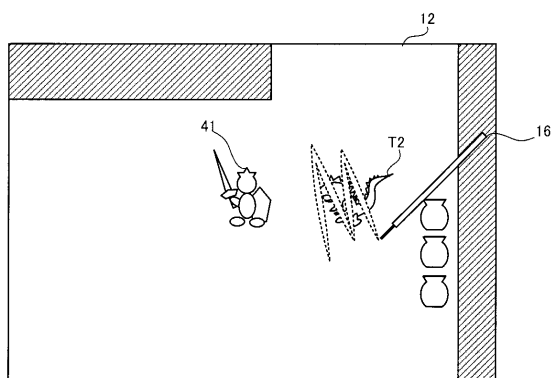
【図 8】



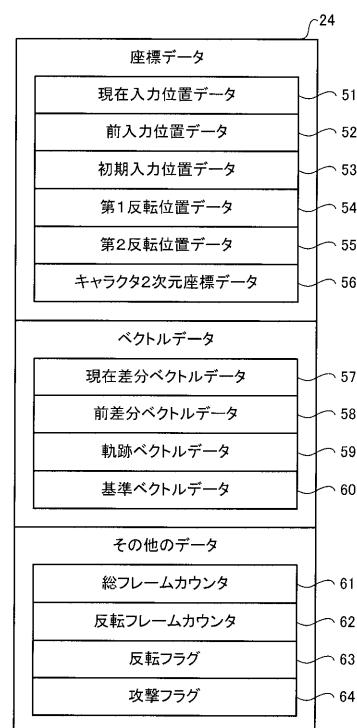
【図 10】



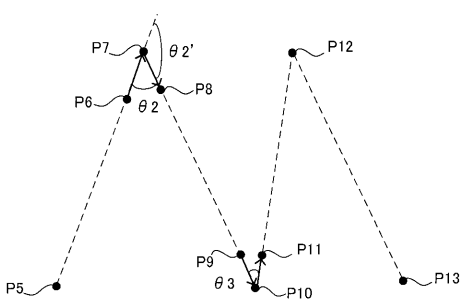
【図 11】



【図 13】

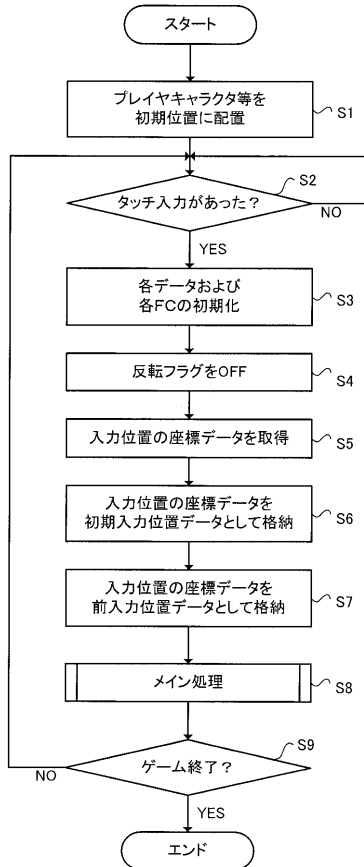


【図 12】

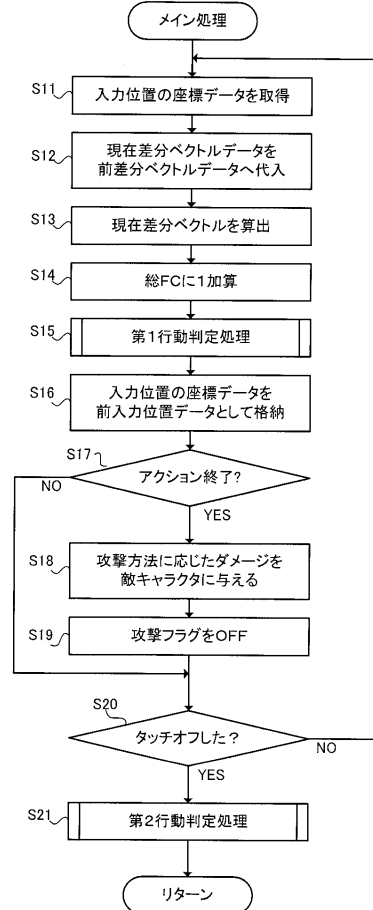




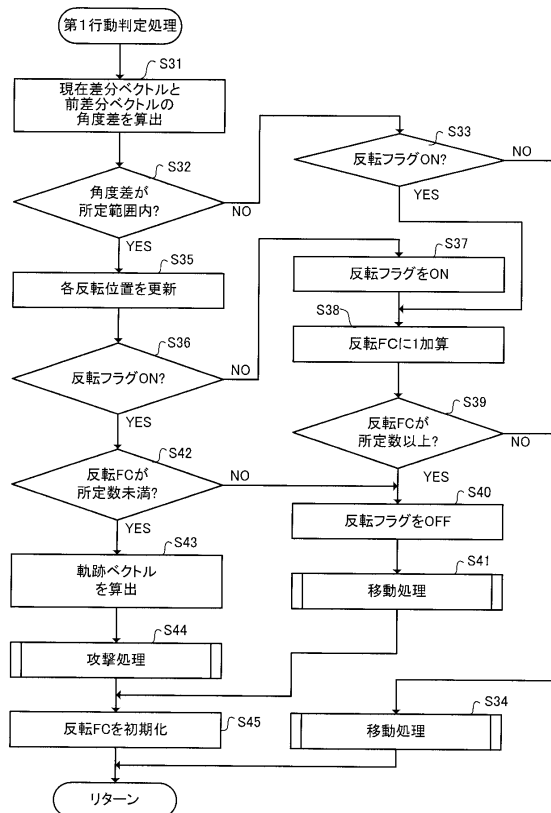
【図 14】



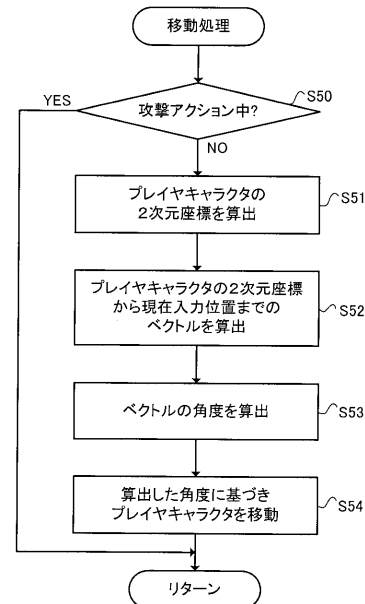
【図 15】



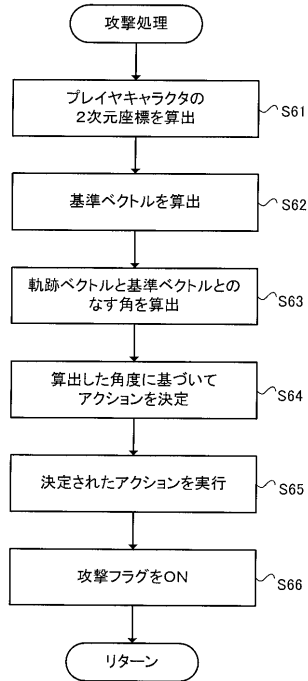
【図 16】



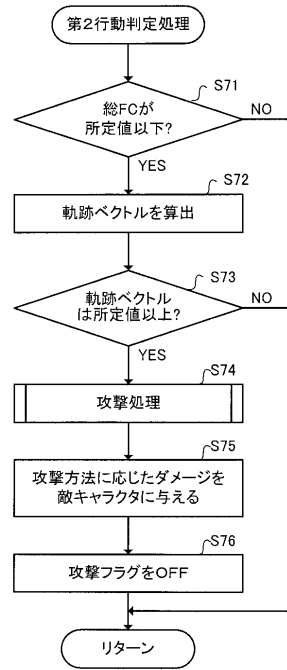
【図 17】



【図 18】



【図 19】



---

フロントページの続き

審査官 宇佐田 健二

- (56)参考文献 特開2002-000939(JP,A)  
特開平11-095650(JP,A)  
特開2002-035411(JP,A)  
特開平11-085375(JP,A)  
特開2001-232056(JP,A)  
特開2005-204754(JP,A)

- (58)調査した分野(Int.Cl., DB名)  
A63F 13/00-13/12, 9/24,  
G06F 3/041,  
G06T 13/00