(19) Deutsches Patent- und Markenamt

(10) **DE 198 18 437 B4** 2012.05.10

(12) # Patentschrift

(21) Aktenzeichen: **198 18 437.9**
(22) Anmeldetag: **24.04.1998**
(43) Offenlegungstag: **29.10.1998**
(45) Veröffentlichungstag der Patenterteilung: **10.05.2012**

(51) Int Cl.: ***A63F 13/00*** (2006.01)
A63F 13/10 (2006.01)

Innerhalb von drei Monaten nach Veröffentlichung der Patenterteilung kann nach § 59 Patentgesetz gegen das Patent Einspruch erhoben werden. Der Einspruch ist schriftlich zu erklären und zu begründen. Innerhalb der Einspruchsfrist ist eine Einspruchsgebühr in Höhe von 200 Euro zu entrichten(§ 6 Patentkostengesetz in Verbindung mit der Anlage zu § 2 Abs. 1 Patentkostengesetz).

(30) Unionspriorität:
9-123273 **25.04.1997** **JP**

(73) Patentinhaber:
Nintendo Co., Ltd., Kyoto, JP

(74) Vertreter:
Grosse, Schumacher, Knauer von Hirschhausen, 80335, München, DE

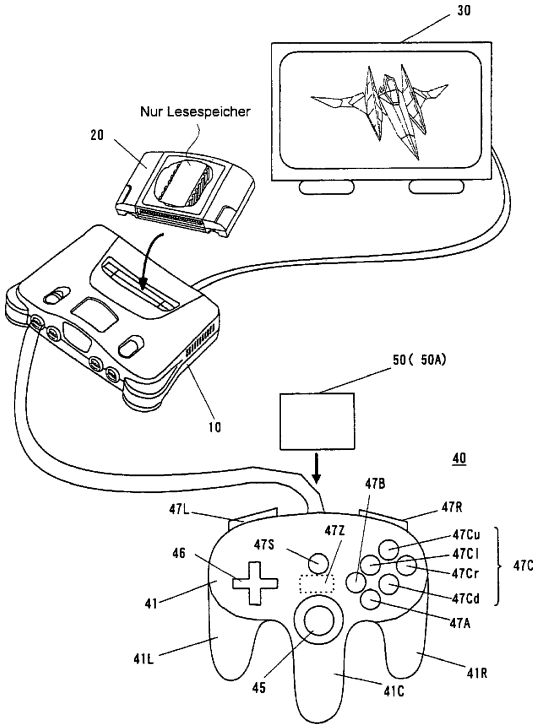
(72) Erfinder:
Miyamoto, Shigeru, Kyoto, JP; Shimizu, Takao, Kyoto, JP; Imamura, Takaya, Kyoto, JP; Morita, Kazuaki, Kyoto, JP; Kihara, Tsuyoshi, Kyoto, JP

(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:

DE	**32 11 019**	**C2**
DE	**43 37 728**	**A1**
DE	**195 20 986**	**A1**
US	**4 445 187**	**A**
US	**5 112 368**	**A**
WO	**94/ 27 677**	**A1**

(54) Bezeichnung: **Videospielvorrichtung**

(57) Hauptanspruch: Videospielvorrichtung zum Spielen eines Videospiels mit mehreren verschiedenen Ablaufen, durch die ein Spieler voranschreiten kann, indem dieser erfolgreich einen der Ablaufe löscht, wobei die Videospielvorrichtung umfasst,
einen Bildschirm, um ein Spielerobjekt (60) und ein anderes Objekt (72a–72n) anzuzeigen und um einen zu spielenden Ablauf aus mehreren Ablaufen auszuwählen, wenn das Videospiel gestartet wird oder nachdem ein Ablauf gelöscht worden ist;
eine erste Zähleinrichtung (11, 22, R10), die ausgebildet ist, eine Punktezahl in dem zu spielenden Ablauf zu zahlen abhängig von einer Beziehung des Spielerobjektes (60) zu dem anderen Objekt,
gekennzeichnet durch,
eine Steuereinrichtung (11, S19, 159), die ausgebildet ist, um eine Punktezahl eines gelöschten Ablaufs in eine Speichereinrichtung (11, R11–R1n) zu schreiben,
wobei die Steuereinrichtung ausgebildet ist, die Punktezahl für den gelöschten Ablauf auf eine anfängliche Punktezahl zurückzusetzen, wenn der Benutzer mit Hilfe des Bildschirms auswählt, den bereits gelöschten Ablauf...



Beschreibung

[0001] Diese Erfindung bezieht sich auf Videospielvorrichtungen und insbesondere auf eine Videospielvorrichtung zum Spielen eines Videospiels durch Auswählen eines Ablaufs und Anzeigen eines Spielerobjekts, welches sich auf Feinde zubewegt, die zu besiegen sind, wie in einem Schießspiel.

[0002] In einem herkömmlichen Videospiel ist es unmöglich, einen einmal gelöschten Ablauf auf halbem Weg des Spiels zu wiederholen. Der einmal gelöschte Ablauf kann nicht noch einmal durchgeführt werden, wenn nicht das Spielgerät zurückgesetzt wird, um wiederum das Spiel von Anfang an zu versuchen. Währenddessen sind andere Videospiele des Standes der Technik fast immer Videospiele, die ein Bewegen des Spielerobjekts anzeigen, bei welchem die Trefferentscheidung in einem Punkt desjenigen durchgeführt wird.

[0003] In einem Spiel, das mit einer Gesamtpunktzahl von Abläufen unter einer Vielzahl von Abläufen konkurriert, wird sogar, wenn ein Ablauf bei einer hohen Punktzahl, jedoch ein anderer Ablauf bei einer niedrigen Punktzahl gelöscht wird, die Gesamtpunktzahl niedrig sein. Aufgrund dieses Umstands ist eine hohe Punktzahl für einen Spieler schwierig zu erreichen. Währenddessen erfordert das Spielgerät, wenn ein Ablauf von Neuem versucht wird, ein Zurücksetzen, um es dem Spiel zu ermöglichen, wieder von Beginn an abgespielt zu werden. Der Ablauf muß, sogar wenn es einem widerstrebt, ihn noch einmal zu spielen, gezwungenermaßen zu Ende gespielt werden. Dementsprechend wird der Spieler mit ungewollten Mühen konfrontiert, was dazu führt, daß sein Interesse an dem Spiel verloren geht oder er ein Lästigkeitsgefühl hat.

[0004] Ebenso ist es unmöglich, wenn ein Spielerobjekt benutzt wird, welches konfiguriert ist, um Treffer an einem Punkt zu bestimmen, Änderungen der Anzeige an jedem bewirkten Anteil des Spielerobjekts wiederzugeben. Dieser Umstand macht es schwierig, das Spielerobjekt mit Diversifikationen anzuzeigen, wodurch Begrenzungen in der Bilddarstellung auferlegt werden. Das Interesse an einem Spiel könnte nicht mehr weiterhin vergrößert werden.

[0005] Aus der Druckschrift DE 195 20 986 A1 ist ein Videospielsystem mit einer Vorrichtung zur Anzeige eines Tastenprogrammierverfahrens bekannt. Das Videospielsystem umfasst eine Bedienungseinrichtung mit Richtungstasten für die Bewegung eines Spielerobjekts, ein Bildverarbeitungsgerät und eine Anzeigeeinheit.

[0006] Die Druckschrift DE 32 11 019 C2 offenbart ein elektronisches Spielgerät mit einer Anzeigeeinrichtung und einer logischen Steuereinrichtung, die in Abhängigkeit von Spielanweisungen, die eine vom Benutzer betätigbare Eingabeeinrichtung an die Steuereinrichtung weitergibt, in einem Speicher gespeicherte Spieldaten verarbeitet. Weiterhin ist eine Zähleinrichtung vorgesehen, die Spielerfolgspunkte zählt. Die Anzeigeeinrichtung dient dazu, das Ergebnis jeder Spieldatenverarbeitung anzuzeigen.

[0007] Deshalb ist es Aufgabe dieser Erfindung, eine neue Videospielvorrichtung zur Verfügung zu stellen, welche eine Vielzahl von Abläufen bereitstellt und es erlaubt sogar in einem schwierigen Spiel eine hohe Punktzahl zu gewinnen, ohne die Notwendigkeit ein Spiel auf halbem Weg zurückzusetzen und die Spieldaten zu löschen, wodurch das Spielamüsement vergrößert und das Leistungsgefühl des Spielers zufriedengestellt wird. Diese Aufgabe wird durch eine Videospielvorrichtung gemäß Anspruch 1 und ein Verfahren gemäß Anspruch 2 gelöst.

[0008] Gemäß einem Beispiel ist ein Videospielsystem vorgesehen, welches eine Bedienungseinrichtung, die durch einen Spieler zu bedienen ist, und ein Bildverarbeitungsgerat zum Liefern von Bilddaten an eine Anzeige hat, um ein Anzeigebild entsprechend einem Programm und abhängig von einer Bedienung der Bedienungseinrichtung anzuzeigen und zu variieren, worin die Bedienungseinrichtung eine Richtungsanzeigeeinrichtung beinhaltet, um eine Richtung der Bewegung eines Spieler Objekts zu bezeichnen, welches in seiner Bewegung auf einem Bildschirm durch den Spieler bedienbar ist, und eine Vielzahl von Bergungsbezeichnungsschaltern zum bezeichnen einer Bewegung des Spielerobjektes, umfasst: eine Erzeugungseinrichtung der Spierobjekt-Bilddaten zu Erzeugen der Daten, um ein Bild des Spierobjektes anzuzeigen; eine Erzeugungseinrichtung der Ablaufbilddaten zu Speichern einer Vielzahl von Ablaufdaten, um verschiedene Abläufe anzuzeigen, durch welche das Spierobjekt voranschreiten kann, und zum Anzeigen eines Ablaufes, welcher basierend auf einem Bedienungszustand der Bedienungseinrichtung durch den Spieler ausgewählt wird; eine Erzeugungseinrichtung der Objektbilddaten zum Anzeigen eines Bildes, um ein Objekt, welches ein anderes als das Spieler Objekt ist, in jedem der Abläufe, in dem das Spielerobjekt voranschreitet, anzuzeigen; eine Erzeugungseinrichtung der Auswahl-Bildschirmdaten zum Anzeigen eines Ablaufausfall-Bildschirmes, durch welche ein Ablauf, der als nächstes zu spielen ist, aus den Abläufen durch Bedienen der Bedienungseinrichtungen ausgewählt wird, wenn ein Spiel gestartet wird oder ein Ablauf gelöscht wird; eine Festsetzungseinrichtung des Punktezahl gebenden Zustandes zum Festsetzen eines Zustandes, um eine Punktezahl in einem Ablauf an die jeweiligen Abläufe zu geben; eine Detektionseinrichtung des Punktezahl gebenden Zustandes zum Detektieren, dass ein Punktezahl gebender Zustand, der durch die Festsetzungseinrichtung des Punktezahl geben-

den Zustand, der durch die Festsetzungseinrichtung de Punktezahl gebenden Zustandes festgesetzt wird, abhängig von der Bedienung zwischen dem Spielerobjekt und dem anderen Objekt getroffen wird, wenn das Spielerobjekt in dem ausgewählten Ablauf voranschreitet; eine erste Punktezahl-Zähleinrichtung zum Zählen einer Punktezahl in dem Ablauf abhängig von einem Ausgang der Detektionseinrichtung des Punktezahl gebenden Zustandes, wenn das Spielerobjekt in dem ausgewählten Ablauf voranschreitet; eine Detektionseinrichtung der Ablauflöschung zum Detektieren, dass ein Löschzustand für jeden Ablauf von dem Spielerobjekt getroffen wird; eine Speichereinrichtung der Ablauf-Punktezahl zum Speichern auf einer Ablauf-für-Ablauf-Basis einer Punktezahl, die für den Ablauf als eine unter den Abläufen durch die erste Punktezahl-Zähleinrichtung gezählt worden ist, eine Steuerungseinrichtung zum Schreiben einer Punktezahl des gelöschten Ablaufes, die von der Zähleinrichtung der Punktezahl gezählt worden ist, in einen korrespondierenden Bereich der Speichereinrichtung der Ablauf-Punkteszahl zu dem gelöschten Ablauf, wenn die Detektionseinrichtung der Ablauflöschung einen Ablauflöschung detektiert, und zum Zurücksetzen der Zähleinrichtung der ersten Punktezahl, um einen Anfangswert in einen korrespondierenden Bereich der Speichereinrichtung der Ablauf-Punktezahl zu einem Ablauf, der unmittelbar zuvor gelöscht wird, zu schreiben, basierend auf der Wahl, in dem gleichen Ablauf voranzuschreiten; und eine Zähleinrichtung der zweiten Punktezahl zum Bestimmen einer Summe der Ablauf-Punktezahl, welche in dem Ablauf-für-Ablauf-Bereich der Speichereinrichtung der Ablauf-Punktezahl gespeichert ist.

[0009] Gemäß einem weiteren Beispiel ist ein Videospielgerät mit einem externen Speichermedium, wie einem ROM, verbunden. Das externe Speichermedium umfasst: ein Erzeugungsprogramm der Bilddaten von dem Spielerobjekt zum Erzeugen der Daten, um ein Bild des Spielerobjektes anzuzeigen; ein Erzeugungsprogramm der Bilddaten des Ablaufes zum Speichern einer Vielzahl von Ablaufdaten, um verschiedene Abläufe anzuzeigen, durch welche das Spielerobjekt voranschreiten kann, und zum Anzeigen eines Ablaufes, der basierend auf einem Bedienungszustand der Bedienungseinrichtung durch den Spieler ausgewählt wird; ein Erzeugungsprogramm der Bilddaten des Objektes zum Anzeigen eines Bildes, um ein Objekt, welches ein anderes als das Spielerobjekt ist, auf jedem der Abläufe, in denen das Spielerobjekt voranschreitet, anzuzeigen; ein Erzeugungsprogramm der Auswahlbildschirmdaten zum Anzeigen eines Ablaufes, der auf dem Bildschirm ausgewählt wird, durch welchen ein nächster Ablauf, der zu spielen ist, ausgewählt wird unter den Abläufen durch das Bedienen der Bedienungseinrichtung, wenn ein Spiel startet oder ein Ablauf gelöscht wird; ein Festsetzungsprogramm des punktezahlgebenden Zustandes zum Festsetzen eines Zustandes, um eine Punktezahl in einem Ablauf zu den jeweiligen Abläufen zu vergeben; ein Detektionsprogramm des punktezahlgebenden Zustandes zum Detektieren, dass ein punktezahlgebender Zustand, der durch die Festsetzungseinrichtung des punktezahlgebenden Zustandes festgesetzt wird, abhängig von der Beziehung zwischen dem Spielerobjekt und dem anderen Objekt, wenn das Spielerobjekt in dem ausgewählten Ablauf voranschreitet, erreicht wird; ein Zählprogramm der ersten Punktezahl zum Zählen einer Punktezahl im Ablauf abhängig von einem Ausgang der Detektionseinrichtung des punktezahlgebenden Zustandes, wenn das Spielerobjekt in dem ausgewählten Ablauf voranschreitet; ein Detektionsprogramm der Ablauflöschung zum Detektieren, dass ein Löschungszustand für jeden Ablauf durch das Spielerobjekt erreicht wird; ein Speicherprogramm der Ablauf-Punktezahl zum Speichern auf einer Ablauf-für-Ablauf-Basis einer Punktezahl, die für den Ablauf unter den Abläufen von der Zähleinrichtung der ersten Punktezahl gezählt worden ist; ein Steuerungsprogramm zum Schreiben einer Punktezahl des gelöschten Ablaufes, die von der Zähleinrichtung der ersten Punktezahl gezählt worden ist, in einen korrespondierenden Bereich der Speichereinrichtung der Ablauf-Punktezahl zu dem gelöschten Ablauf, wenn die Detektionseinrichtung der Ablauflöschung eine Ablauflöschung detektiert und zum Zurücksetzen der Zähleinrichtung der ersten Punktezahl, um einen Anfangswert in einem korrespondierenden Bereich der Speichereinrichtung der Ablauf-Punktezahl zu einem Ablauf, der unmittelbar zuvor gelöscht wird, basierend auf der Auswahl, dass in einem gleichen Ablauf vorangeschritten wird, zu schreiben; und ein Zählprogramm der zweiten Punktezahl zum Bestimmen einer Summe der Ablaufpunktezahlen, die in dem Ablauf-für-Ablauf-Bereich der Speichereinrichtung der Ablaufpunktezahl gespeichert sind.

[0010] Das Speichermedium, z. B. ein externer ROM, kann vorher mit einem Erzeugungsprogramm der Bilddaten des Spielerobjektes, einem Erzeugungsprogramm der Bilddaten eines Ablaufes und einem Erzeugungsprogramm der Bilddaten des Objektes gespeichert werden, so dass das Videospielgerät ein Spielerobjektbild anzeigen kann und andere Objektbilder anzeigen kann, entsprechend den Bilddaten, die von diesem Programm auf der Anzeigevorrichtung gegeben werden.

[0011] Das Videospielgerät kann eine CPU und eine RCP (Reality Co-Processor) beinhalten, durch welche ein Auswahlbildschirm auf der Anzeigevorrichtung angezeigt wird, so dass dem Spieler ermöglicht wird, einen Ablauf bei einem Start eines Spieles oder Löschen eines Ablaufes auszuwählen. Abhängig davon kann der Spieler einen Ablauf dazu auswählen, voranzuschreiten. Somit wird das Spiel begonnen.

[0012] Während des Spielens eines Spieles detektiert die Detektionseinrichtung des Punktezahlzustandes, ob ein Zustandssatz des Punktezahl-Gebens (Treffer-detektion) im Hinblick auf die Beziehung zu dem Spielerobjekt zu anderen Objekten erfüllt ist oder nicht. Wenn der Zustand erfüllt ist, so zählt die erste Punktezahl-Zähleinrichtung eine Punktezahl für den Ablauf. Wenn eine Ablauf-Löschung durch die Detektionseinrichtung des Ablauflöschens detektiert wird, so wird eine erste Punktezahl, die durch die Zähleinrichtung der ersten Punktezahl gezählt wird, in ein Gebiet der Speichereinrichtung der Ablauf-Punktezahl geschrieben, und die Zähleinrichtung der ersten Punktezahl wird zurückgesetzt.

[0013] Wenn ein Ablauf, der bereits gelöscht ist, wiederum von dem Spieler ausgewählt wird, so wird eine anfängliche Punktezahl in ein relevantes Gebiet der Speichereinrichtung der Ablauf-Punktezahl geschrieben. Deshalb wird dieser Ablauf gestartet, um zu spielen, und eine erste Punktezahl wird gezählt. Die erste Punktezahl wird durch die Zähleinrichtung der zweiten Punktezahl aufsummiert.

[0014] Es ist ein Videospielsystem und ein Videospiel-Speichermedium erhältlich, welches das Realismusgefühl für ein Videospiel erhöhen kann und das Leistungsgefühl des Spieler befriedigen kann und das Interesse an dem Videospiel erhöht.

[0015] Ebenso ist es, wenn eine Vielzahl von Abläufen geliefert wird, möglich, eine hohe Punktezahl sogar in einem schwierigen Spiel zu erzielen und zu einem gleichen Ablauf (z. B. Wiederspielen) voranzuschreiten, sogar nachdem ein Ablauf oder eine Szene, die schwierig zu handhaben ist, gelöscht wurde, so dass der Spieler eine hohe und zufriedenstellende Punktezahl gewinnen kann, ohne die Notwendigkeit des Plazierens der Spieldaten auf halbem Wege zurück zu einem Anfangszustand, was das Spielamüsement erhöht und das Leistungsgefühl des Spielers befriedigt.

[0016] Gemäß einem Beispiel umfasst ein Videospielsystem: eine Erzeugungseinrichtung der Bilddaten des Spielerobjektes zum Erzeugen von Daten, um ein Bild des Spielerobjektes, welches aus einer Vielzahl von Anteilen, die in dem Spielerobjekt aufgeteilt sind, verbunden ist, anzuzeigen; Erzeugungseinrichtung der Bilddaten des Beeinflussungsobjektes zum Erzeugen der Bilddaten des Beeinflussungsobjektes, um ein Beeinflussungsobjektbild anzuzeigen, welches um das Spielerobjekt herum angezeigt wird, um einen Effekt auf das Spielerobjekt zu haben; eine Treffer-Bestimmungseinrichtung zum Bestimmen von Treffern für jedes der aufgeteilten Anteile des Spielerobjektes, wenn das Spielerobjekt durch das Beeinflussungsobjekt beeinflusst wird; eine Bild-Veränderungseinrichtung zum Verändern der Bilddaten, die von der Erzeugungseinrichtung der Bilddaten des Spielerobjektes erzeugt werden, um so in der Anzeige die aufgeteilten Anteile des Spielerobjektes, die als getroffen durch die Bestimmungseinrichtung der Treffer bestimmt worden sind, zu variieren; und eine Veränderungseinrichtung des Bewegungszustandes zum Verändern eines Bewegungszustandes des Spielerobjektes abhängig von einer Position des aufgeteilten Anteiles des Spielerobjektes, welches die Veränderung durch die Veränderungseinrichtung des Bildes wiedergibt.

[0017] Jeder geteilte Anteil des Spielerobjektes kann weiterhin aus einer Vielzahl von Unterteilungs-Anteilen aufgebaut sein. In solch einem Fall verändert die Bildveränderungseinrichtung die Bilddaten, welche durch die Erzeugungseinrichtung der Bilddaten des Spielerobjektes erzeugt werden, in solch einer Weise, dass eine Veränderung in der Anzeige an einen Unter-Teil-Anteil eines geteilten Anteiles des Spielerobjektes gegeben wird, bei dem die Trefferbestimmungseinrichtung einen Treffer bestimmt hat.

[0018] Anzumerken ist, dass das Spielerobjekt typischerweise links- und rechtsseitig symmetrisch liegend ausgewählt wird, wenn man in die Ebene hineinschaut. Die Trefferbestimmungseinrichtung detektiert, bei welchem Anteil auf der linken oder rechten Seite ein Treffer in dem Spielerobjekt detektiert wird. Die Bildveränderungseinrichtung veranlasst, dass eine Form auf einer Seite, die als getroffen durch die Trefferbestimmungseinrichtung bestimmt wird, verändert wird. Die Veränderungseinrichtung des Bewegungszustandes veranlasst, dass der Bewegungszustand in einer Weise verändert wird, der unterschiedlich zwischen einer Seite, die verändert ist und einer Seite, die unverändert ist, ist, durch die Bildveränderungseinrichtung.

[0019] Z. B. in dem Fall, dass das Spielerobjekt in Form eines Flugzeuges vorhanden ist, verändert, wenn der rechte Flügel als getroffen angenommen wird, die Bildveränderungseinrichtung die rechte Flügelform. In diesem Zustand fährt, wenn das Spielerobjekt durch den Joystick bewegt wird, die Veränderungseinrichtung des Bewegungszustandes langsam die Bewegungsgeschwindigkeit des rechten Flügels im Vergleich zu der Bewegungsgeschwindigkeit des linken Flügels herunter. Dementsprechend beginnt das Spielerobjekt (Flugzeug) in Richtung der rechten Seite auf eine natürliche Weise in einer Kurve zu fliegen.

[0020] In diesem Aspekt ist es möglich, ein dramatisches und interessantes Videospiel zu realisieren, wenn ein geteilter Anteil des Spielerobjektes getroffen wird, um einen Zustand der Anzeige des Spielerobjektes abhängig von einem Anteil, der beschädigt ist, zu verändern.

[0021] Wenn das Spielerobjekt so konfiguriert ist, dass es sich in seiner Bewegung in einer diversifizierten Weise verändern soll, so dass der Spieler eine Bedienungseinrichtung bedienen muss, um mit der Veränderung fertig zu werden, wird ein dramatisches und interessantes Videospiel geliefert, welches wirklichkeitsgetreu in der Bewegung des Spielerobjektes ist.

[0022] Wenn das Spielerobjekt in seiner Bewegung abhängig von einem Betrag des Angriffs für jeden geteilten Anteil des Spielerobjektes verändert wird, dann ist ein dramatisches und interessantes Videospiel möglich, welches wirklichkeitsgetreu in der Bewegung des Spielerobjektes ist.

[0023] Die oben beschriebenen Ziele und andere Ziele, Merkmale, Aspekte und Vorteile der vorliegenden Erfindung werden aus der folgenden detaillierten Beschreibung der vorliegenden Erfindung offensichtlicher werden, wenn sie in Verbindung mit den beigefügten Zeichnungen benutzt wird.

[0024] Fig. 1 ist eine externe Ansicht, die eine Struktur eines Videospielsystems von einer Ausführungsform der vorliegenden Erfindung zeigt;

[0025] **Fig.** 2 ist ein Blockdiagramm des Videospielsystems von der einen Ausführungsform der vorliegenden Erfindung;

[0026] Fig. 3 ist ein detailliertes Schaltungsdiagramm einer Steuerungseinheit-Schaltung **18**;

[0027] Fig. 4 ist ein Blockdiagramm einer Steuerungseinheit **40**;

[0028] Fig. 5 ist eine Speicherabbildung, die darstellend einen gesamten Speicherplatz von einem externen ROM **21** zeigt;

[0029] Fig. 6 ist eine Speicherabbildung, die im Detail den Teil des Speicherplatzes der externen ROM **21** zeigt;

[0030] Fig. 7 ist eine Speicherabbildung, die den gesamten Speicherplatz des RAM **15** zeigt;

[0031] Fig. 8 ist eine Speicherabbildung, die im Detail den Teil des Speicherplatzes des RAM **15** zeigt;

[0032] Fig. 9 ist eine Ansicht, die Spielabläufe von einem Beispiel, auf welches diese Erfindung angewendet wird, zeigt;

[0033] Fig. 10 ist ein Diagramm, welches einen Ablaufauswahlbildschirm in einem Spiel, welches in Fig. 10 gezeigt wird, zeigt;

[0034] Fig. 11 ist ein Diagramm, welches eine Spielgebietsabbildung zum Erklären eines Spielinhaltes von einem Beispiel, auf welches diese Erfindung angewendet wird, zeigt;

[0035] Fig. 12 ist ein Diagramm, welches darstellerisch einen Nachrichten-Ausgangsinhalt in einem Kommunikationsprozess mit Mitgliedern in dem Spiel von Fig. 11 zeigt;

[0036] Fig. 13 ist eine Ansicht, die ein Beispiel einer Auf-dem-Bildschirm-Anzeige zeigt, die einen Nachrichtenausgang hat, der basierend auf dem Kommunikationsprozess mit den Mitgliedern in dem Spiel von Fig. 11 dargestellt wird;

[0037] Fig. 14 ist eine Ansicht, die ein Beispiel einer Auf-dem-Bildschirm-Anzeige in einem Zustand zeigt, der gegen ein Knopfzeichen in dem Spiel der Fig. 11 ankämpft;

[0038] Fig. 15 ist ein Haupt-Flussdiagramm eines Spielablaufes von einer Ausführungsform der vorliegenden Erfindung;

[0039] Fig. 16 ist ein Subroutine-Flussdiagramm, welches einen detaillierten Ablauf für den Ablauf-Auswahl-Schirm zeigt;

[0040] Fig. 17 ist ein Subroutine-Flussdiagramm, welches einen detaillierten Prozess für das Modusshiften zeigt;

[0041] Fig. 18 ist ein Flussdiagramm zum Erklären des Datentransfers zwischen der Steuerungseinheit-Schaltung **18** und dem Videospiel-Maschinenhauptkörper;

[0042] Fig. 19, Fig. 20 und Fig. 21 sind Subroutine-Flussdiagramme für einen Kommunikationsprozess mit dem Mitglied entsprechend einem Beispiel eines Nachrichtenausgabeprozesses, um dem Spielfortschritt zu assistieren;

[0043] Fig. 22 und Fig. 23 sind Subroutine-Flussdiagramme von einem Auffrischungsmaterial-Versorgungsprozess entsprechend einem anderen Beispiel eines Nachrichtenausgabeprozesses, um dem Spielfortschritt zu assistieren;

[0044] Fig. 24, Fig. 25 und Fig. 26 sind Subroutine-Flussdiagramme für ein Mitgliedobjekt- Prozess;

[0045] Fig. 27 ist ein Subroutine-Flussdiagramm für einen Feindobjekt- Prozess;

[0046] Fig. 28 ist ein Subroutine-Flussdiagramm, welches im Detail die Funktionen des Teiles der Schritte, die in dem Feindobjekt-Prozess von Fig. 27 enthalten sind, zeigt;

[0047] Fig. 29 ist ein Subroutine-Flussdiagramm von einem stationären Objekt-Prozess;

[0048] Fig. 30 ist ein Subroutine-Flussdiagramm für einen Zurückgeb Prozess; und

[0049] Fig. 31 ist ein Subroutine-Flussdiagramm für einen Phon-Prozess.

[0050] Fig. 32 ist ein Subroutine-Flussdiagramm eines Spielerobjektprozesses;

[0051] Fig. 33 ist ein Subroutine-Flussdiagramm eines Spielerobjektprozesses;

[0052] Fig. 34 ist eine X(–Z)-Koordinate in einem dreidimensionalen Raum in einem Ein-Richtungs-Abrollmodus;

[0053] Fig. 35 ist eine externe Ansicht eines Spielerobjekts **60**;

[0054] Fig. 36 ist ein Subroutine-Flussdiagramm eines Spielerobjektprozesses;

[0055] Fig. 37 ist eine X(–Z)-Koordinate eines dreidimensionalen Raums in einem All-Bereichs-Modus;

[0056] Fig. 38 ist ein Subroutine-Flussdiagramm des Spielerobjektprozesses;

[0057] Fig. 39 ist ein Subroutine-Flussdiagramm eines Trefferbestimmungsprozesses;

[0058] Fig. 40 ist eine externe Ansicht des Spielerobjekts **60**, welches einen Teil von beiden Flügeln verliert;

[0059] Fig. 41 ist eine externe Ansicht des Spielerobjekts **60**, welches einen Teil eines linken Flügels verliert;

[0060] Fig. 42 ist ein Subroutine-Flussdiagramm des Spielerobjektprozesses;

[0061] Fig. 43 ist eine externe Ansicht des Spielerobjekts **60**, welches einen größeren Teil des linken Flügels verliert;

[0062] Fig. 44 ist ein Subroutine-Flussdiagramm eines Kameraprozesses;

[0063] Fig. 45 ist ein Subroutine-Flussdiagramm des Kameraprozesses;

[0064] Fig. 46 ist eine Ansicht, die ein Beispiel einer Bildschirmanzeige des Spielerobjekt zeigt;

[0065] Fig. 47 ist eine Ansicht, die ein anderes Beispiel einer Bildschirmanzeige des Spielerobjekts **60** zeigt, und

[0066] Fig. 48 ist ein Subroutine-Flussdiagramm eines Trefferbestimmungsprozesses für Feindobjekte.

[0067] Nun werden Erklärungen bezüglich der Strukturen eines Videospielsystems der Erfindung und eines Videospiel-Speichermediums, welches dafür benutzt wird, dargelegt. Nebenbei gesagt, kann, obwohl in den unten ausgeführten Ausführungsformen Erklärungen für den Fall eines exklusiven Videospielgerätes vorhanden sein werden, ein Bildverarbeitungsgerät ebenso als ein anderes Beispiel auf einen Personalcomputer oder dergleichen angewendet werden. Obwohl eine Bedienungseinrichtung als ein Fall einer spielexklusiven Steuerungseinrichtung erklärt werden wird, kann eine Eingabevorrichtung, wie eine Tastatur oder eine Maus, verwendet werden, vorausgesetzt, dass das Videospielsystem der Erfindung auf ein Bildverarbeitungsgerät, wie bei einem Personalcomputer, angewendet wird.

[0068] Fig. 1 ist eine externe Ansicht, die eine Struktur eines Videospielsystems entsprechend einer Ausführungsform der vorliegenden Erfindung zeigt. Das Videospielsystem ist aufgebaut, indem es einen Hauptkörper **10** des Videospielgerätes, eine ROM-Kassette **20** als ein Beispiel einer externen Speichervorrichtung, eine CRT-Anzeige **30** als ein Beispiel einer Anzeigevorrichtung, die mit dem Hauptkörper **10** des Videospielgerätes verbunden ist, und eine Steuerungseinrichtung **40** als ein Beispiel einer Bedienungseinrichtung (oder einer Bedienungseingabeeinrichtung) beinhaltet. Die Steuerungseinrichtung **40** ist lösbar befestigt, wenn erforderlich mit einer ROM-Kassette **50** (oder einer Vibrationskassette **50A**).

[0069] Die Steuerungseinrichtung **40** ist aufgebaut, indem sie mit einer Vielzahl von Schaltern oder Knöpfen auf einem Gehäuse **41**, die in einer derartigen Form sind, dass sie durch die jeweilige oder eine Hand ergreifbar sind, ausgestattet ist. Insbesondere ist die Steuerungseinrichtung **40** mit Handgriffen **41L**, **41C**, **41R** an den unteren Bereichen der jeweiligen linken, rechten und mittigen Seite des Gehäuses **41** ausgestattet, wodurch ein Bedienungsbereich auf einer oberen Oberfläche davon zur Verfügung gestellt wird. In dem Bedienungsbereich sind ein Analog-Eingabe-Joystick **47** an einem zentralen unteren Anteil davon, ein kreuzgeformter digitaler Richtungsschalter (hierin als „Kreuzschalter" bezeichnet) **46** auf der linken Seite und eine Vielzahl von Knopfschaltern **47A**–**47Z** auf der rechten Seite angeordnet. Der Joystick **45** wird benutzt, um eine Bewegungsrichtung und/oder Bewegungsgeschwindigkeit (oder den Bewegungsbetrag) des Spielerobjektes basierend auf dem Betrag und der Richtung der Joystick-Neigung

zu bestimmen oder einzugeben. Der Kreuzschalter **46** wird benutzt, um eine Bewegungsrichtung des Spielerobjektes anstelle des Joysticks **45** einzugeben oder zu bestimmen. Die Vielzahl der Knopfschalter **47** beinhaltet die Schalter **47A**, **47B** zum Bestimmen der Bewegung des Spielerobjektes, einen Schalter **47C**, um die Bildansicht eines Beobachters, wie durch eine Kamera gesehen, umzuschalten, einen Startschalter **47S**, einen Bewegungsschalter **47L**, der auf einer seitlichen Oberfläche eines linken oberen Anteiles des Gehäuses **41** angeordnet ist, einen Bewegungsschalter **47R**, der auf einer seitlichen Oberfläche eines rechten oberen Anteiles des Gehäuses **41** angeordnet ist, und einen Schalter **47Z**, der auf einer Rückseite des Handgriffes **41C** angeordnet ist. Der Schalter **47C** besteht aus vier Druckschaltern **47Cu**, **47Cd**, **47Cl**, **47Cr**, die an der oberen, unteren, linken und rechten Seite angeordnet sind, so dass sie ebenso bei einer anderen Anwendung als dem Wechseln der Kamera-Bildansicht der Beobachter benutzt werden können, um die Bewegungsgeschwindigkeit (z. B. Beschleunigung, Bremsen, etc.) in einem Schuss- oder Action-Spiel zu steuern. Die Druckschalter **47A**–**47Z** können aus Funktionen entsprechend einem Spielprogramm definiert werden.

[0070] **Fig.** 2 ist ein Blockdiagramm des Videospielsystems in einer Ausführungsform der Erfindung. Das Videospielgerät **10** beinhaltet darin eine zentrale Prozessoreinheit (hierin als „CPU" bezeichnet) **11** und einen Coprozessor (Realitäts-Coprozessor: hierin bezeichnet als „RCP") **12**. Der RCP **12** beinhaltet eine Bussteuerungsschaltung **121** zum Steuern von Bussen, eine Bildverarbeitungseinheit (Realitätssignal-Prozessor; hierin als „RSP") **122** zum Durchführen von polygonen Koordinatentransformationen, Schattenbehandlung, und so weiter, eine Bildverarbeitungseinheit (Realitätsanzeige-Prozessor; hierin als „RDP" bezeichnet) **123** zum Rastern der polygonen Daten in ein Bild, welches anzuzeigen ist, und zum Umwandeln desselben in eine Datenform **8** (Punktdaten), welche auf einem Rahmenspeicher speicherbar sind. Der RCP **12** ist mit einem Kassettenanschluss **13** zum lösbaren Verbinden mit einer ROM-Kassette **20**, an einem Plattenantrieb-Anschluss **14** zum lösbaren Verbinden mit einem Plattenantrieb **26**, und einem RAM **15** verbunden. Ebenso ist der RCP **12** mit einer Schaltung **16**, die ein Tonsignal erzeugt, zum Ausgeben eines Klangsignales, welches durch die CPU **11** verarbeitet wird, und einer Schaltung **17**, die ein Bildsignal erzeugt, zum Ausgeben eines Bildsignales verbunden. Weiterhin ist der RCP **12** mit einer Steuerungseinheit-Steuerungsschaltung **18** zum seriellen Übertragen von Bedienungsdaten für eine oder eine Mehrzahl von Steuerungseinheiten **40A**–**40D** und/oder Daten für eine RAM-Kassette **50** zur Erweiterung verbunden.

[0071] Die Bussteuerungsschaltung **121**, welche in dem RCP **12** enthalten ist, wandelt den Befehl, welcher in einem Parallelsignal von der CPU **11** über einen Bus geliefert wird, parallel-seriell um, um so ein serielles Signal an die Steuerungseinrichtung-Steuerungsschaltung **18** zu liefern. Ebenso wandelt die Bussteuerungsschaltung **121** das serielle Signal, welches von der Steuerungseinrichtung-Steuerungsschaltung **18** eingegeben wird, in ein Parallelsignal um, um einen Ausgang an die CPU **11** über einen Bus zu geben. Die Daten, die einen Bedienungszustand, welcher von der Steuerungseinrichtung **40A**–**40D** ausgelesen wird, darstellen, werden von der CPU **11** verarbeitet und zeitlich innerhalb eines RAM **15** usw. gespeichert. Mit anderen Worten: der RAM **15** beinhaltet einen Speicherbereich zum zeitlichen Speichern der Daten, die von der CPU **11** zu verarbeiten sind, so daß er für gleichmäßiges Lesen und Schreiben der Daten über die Bussteuerungsschaltung **121** benutzt wird.

[0072] Die Schaltung **16**, welche das Klangsignal erzeugt, ist mit einem Verbinder **195** verbunden, der auf einer Rückseite des Videospielgerätes **10** angeordnet ist. Die Schaltung **17**, welche das Bildsignal erzeugt, ist mit einem Verbinder **196** verbunden, welcher auf der Rückseite des Videospielgerätes **10** angeordnet ist. Der Verbinder **195** ist absteckbar mit einem Verbindungsteil einer klangerzeugenden Vorrichtung **32**, wie einem TV-Lautsprecher etc. verbunden. Der Verbinder **196** ist absteckbar mit einem Verbindungsteil einer Anzeige **31**, wie einem TV-Receiver oder CRT verbunden.

[0073] Die Steuerungseinrichtung-Steuerungsschaltung **18** ist mit Steuerungseinrichtung-Verbindern (hierin als „Verbinder" bezeichnet) **191**–**194** verbunden, die auf der Vorderseite des Videospielgerätes **10** angeordnet sind. Die Verbinder **191**–**194** sind abnehmbar mit einer Steuerungseinrichtung **40A**–**40D** über eine Verbindungssteckerbuchse verbunden. Die Verbindung einer Steuerungseinrichtung **40A**–**40D** mit einem Verbinder **191**–**194** auf diese Weise plaziert die Steuerungseinrichtung **40A**–**40D** in eine elektrische Verbindung mit dem Videospielgerät **10**, wodurch das Senden/Empfangen oder die Übertragung von Daten dazwischen ermöglicht wird.

[0074] Fig. 3 ist ein detailliertes Schaltungsdiagramm der Steuerungseinrichtungs-Steuerungsschaltung **18**. Die Steuerungseinrichtungs-Steuerungsschaltung **18** wird zum Senden und Empfangen von Daten in Serie zwischen dem RCP **12** und den Steuerungseinrichtungsverbindern **191**–**194** benutzt. Die Steuerungseinrichtungs-Steuerungsschaltung **18** beinhaltet eine Datentransferübertragungs-Steuerungsschaltung **181**, eine Sendeschaltung **182**, eine Empfangsschaltung **183** und ein RAM **184** zum zeitlichen Speichern der Sende- und Empfangsdaten. Die Datenübertragungs-Steuerungsschaltung **181** beinhaltet eine Schaltung zum parallel-seriellen Umwandeln und eine Schaltung zum seriellen-paral-

lelen Umwandeln, um ein Datenformat während des Datentransfers umzuwandeln und weiterhin führt sie die Steuerung durch, um in den RAM **184** zu schreiben und aus dem RAM **184** zu lesen. Die Schaltung zum seriellen-parallelen Umwandeln wandelt die seriellen Daten, welche von dem RCP **12** geliefert werden, in parallel Daten um, um sie an den RAM **184** oder die Sendeschaltung **182** zu liefern. Die Schaltung zum parallelen-seriellen Umwandeln wandelt die parallelen Daten, welche von dem RAM **184** oder der Empfangsschaltung **183** geliefert werden, in serielle Daten um, um sie an den RCP **12** zu liefern. Die Sendeschaltung **182** wandelt den Befehl zum Steuern des Steuerungseinrichtung-**40**-Signales, welches gelesen wird, das von der Datenül-Steuerungsschaltung **181** geliefert wird, und die Schreibdaten (parallele Daten) an die RAM-Kassette in serielle Daten um, um an die Kanäle CH1–CH4 jeweils korrespondierend zu der Steuerungseinrichtung **40A**–**40D** geliefert zu werden. Die Empfangsschaltung **183** empfängt in seriellen Daten die Bedienungszustandsdaten der Steuerungseinrichtung **40A**–**40D**, welche über die korrespondierenden Kanäle CH1–CH4 eingegeben werden, und liest die Daten von der RAM-Kassette **50** aus, um sie in parallele Daten umzuwandeln, die an die Datenübertragungs-Steuerungsschaltung **181** zu liefern sind. Die Datenübertragungs-Steuerungsschaltung **181** dient dazu, dass Schreiben in den RAM **184** auf den Daten, die von dem RCP **12** übertragen werden, den Bedienungszustandsdaten der Steuerungseinrichtung **40A**–**40D**, welche durch die Empfangsschaltung **183** empfangen werden, oder den Daten, welche aus der RAM-Kassette **50** ausgelesen werden, zu steuern und liest die Daten aus dem RAM **184** basierend auf einem Befehl von dem RCP **12** aus, um sie so an den RCP **12** zu übertragen.

[0075] Der RAM **184**, der so nicht gezeigt ist, beinhaltet die Speicherbereiche **184a**–**184h**. Der Bereich **184h** ist mit einem Befehl für den ersten Kanal gespeichert, während der Bereich **184b** mit des Sende- und Empfangsdaten für den ersten Kanal gespeichert ist. Ähnlich ist der Bereich **184c** mit einem Befehl für den zweiten Kanal gespeichert, während der Bereich **184d** mit den Sende- und Empfangsdaten für den zweiten Kanal gespeichert ist. Der Bereich **184e** ist mit einem Befehl für den dritten Kanal gespeichert, während der Bereich **184f** mit den Sende- und Empfangsdaten für den dritten Kanal gespeichert ist. Der Bereich **184g** ist mit einem Kanal für den vierten Kanal gespeichert, während der Bereich **184a** mit den Sende- und Empfangsdaten für den vierten Kanal gespeichert ist.

[0076] Fig. 4 ist ein detailliertes Schaltungsdiagramm der Steuerungseinrichtung **40** und der RAM-Kassette **50**. Das Gehäuse der Steuerungseinrichtung **40** beinhaltet eine Bedienungssignal-Verarbeitungsschaltung **44** etc., um einen Bedienungszustand des Joysticks **45**, der Schalter **46** und **47** etc. zu detektieren, um die detektierten Daten an die Steuerungseinrichtung-Steuerungsschaltung **18** zu übertragen. Die Bedienungssignal-Verarbeitungsschaltung **44** beinhaltet eine Empfangsschaltung **441**, eine Steuerungsschaltung **442**, eine Schaltersignal-Detektionsschaltung **443**, eine Zählerschaltung **444**, eine Joyport-Steuerungsschaltung **446**, eine Rücksetzschaltung **447** und ein NOR-Gate **448**. Die Empfangsschaltung **441** wandelt ein serielles Signal, wie ein Steuerungssignal, welches von der Steuerungseinrichtungs-Steuerungsschaltung **18** gesendet wird, oder Schreibdaten an die RAM-Kassette **50** in ein paralleles Signal um, um es an die Steuerungsschaltung **442** zu liefern. Die Steuerungsschaltung **442** erzeugt ein Rücksetzsignal, um (0) über das NOR-Gate **448** zurückzusetzen, zählt die Werte eines X-Achsen-Zählers **444X** und eines Y-Achsen-Zählers **444Y** innerhalb des Zählers **44**, wenn das Steuerungssignal, welches von der Steuerungseinrichtungs-Steuerungsschaltung **18** übertragen wird, ein Signal zum Zurücksetzen von X, Y-Koordinaten des Joysticks **45** ist.

[0077] Der Joystick **45** beinhaltet X-Achsen- und Y-Achsen-Fotounterbrecher, um eine Hebelneigung in die X-Achsen- und Y-Achsen-Komponenten aufzulösen, um Pulse in einer Zahl zu erzeugen, die proportional zu der Neigung ist, so daß er Pulssignale jeweils zu dem Zähler **444X** und dem Zähler **444Y** liefert. Der Zähler **444X** zählt die Zahl der Pulse, welche in Anwort auf den Neigungsbetrag erzeugt werden, wenn der Joystick in die X-Achsen-Richtung geneigt wird. Der Zähler **444Y** zählt die Zahl der Impulse, die Antwort auf den Neigungsbetrag erzeugt werden, wenn der Joystick **45** in die Y-Achsen-Richtung geneigt wird. Dementsprechend dient der resultierende X-Achsen- und Y-Achsen-Vektor, welcher durch die Zählwerte der Zähler **444X** und **444Y** bestimmt wird, dazu, eine Bewegungsrichtung und eine Koordinatenposition des Spielerobjektes oder des Herrscherzeichens oder eines Cursors zu bestimmen. Nebenbei gesagt, werden der Zähler **444X** und der Zähler **444Y** ebenso auf ihre Werte zurückgesetzt, wenn ein Rücksetzsignal von der Rücksetzsignal-Erzeugungsschaltung **447** aufgrund des Einschaltens der Versorgung geliefert wird oder ein Rücksetzsignal von der Schaltsignal-Detektionsschaltung **443** aufgrund des gleichzeitigen Drückens von zwei vorbestimmten Schaltern geliefert wird.

[0078] Die Schaltungssignal-Detektionsschaltung **443** antwortet auf einen Ausgangsbefehl eines Schalterzustandes, welcher in einer konstanten Periode (z. B. TV-Rahmenperiode von einem 1/30 zweiten Intervall) von der Steuerungsschaltung **442** geliefert wird, um ein Signal zu lesen, welches abhängig von einem Druckzustand des Kreuzschalters und des Schalters **47A**–**47Z** variiert, und liefert es dann an die Steuerungsschaltung **442**. Die Steuerungsschaltung **442** antwortet auf ein Auslese-Befehlssignal der Be-

dienungszustanddaten von der Steuerungseinheits-Steuerungsschaltung **18**, um ein vorbestimmtes Datenformat der Bedienungszustandsdaten auf den Schaltern **47A**–**47Z** zu liefern und die Zählwerte der Zähler **444X**, **444Y** an die Sendeschaltung **445** zu liefern. Die Sendeschaltung **445** wandelt das Parallelsignal, welches von der Steuerungsschaltung **442** ausgegeben wird, in ein serielles Signal um und überträgt es an die Steuerungseinrichtungs-Steuerungsschaltung **18** über ein Umwandlungsschaltung **43** und ein Signalleitung **42**. Die Steuerungsschaltung **442** ist mit einer Anschlußsteuerungsschaltung **446** über einen Adressenbus und einen Datenbus als auch über einen Anschlußverbinder **449** verbunden. Die Anschlußsteuerungsschaltung **446** führt Dateneingabe/Ausgabe(oder Sende/Empfang)-Steuerung entsprechend einem Befehl von der CPU **11** durch, wenn die RAM-Kassette **50** mit dem Anschlußverbinder **449** verbunden ist.

[0079] Die RAM-Kassette **50** ist so aufgebaut, daß der RAM **51** mit dem Adressenbus und dem Datenbus verbindet und den RAM **51** mit einer Batterie **52** verbindet. Der RAM **51** ist ein RAM, der eine Kapazität hat, die niedriger ist als die Hälfte einer maximalen Speicherkapazität, auf die über den Adressenbus zugegriffen werden kann (z. B. 256 k Bits). Der RAM **51** soll mit Backup-Daten mit Bezug auf ein Spiel gespeichert werden und er behält die Backup-Daten zurück durch ein Versorgt werden mit elektrischer Versorgung von der Batterie **52**, sogar wenn die RAM-Kassette **50** von dem Anschlußverbinder **449** zurückgezogen wird.

[0080] Nebenbei gesagt, ist es möglich, dort wo ein Stoßzustand mit Kollisionen, Stößen usw. durch Bilder oder Klänge, die in einer hohen realistischen Weise ausgegeben werden, dargestellt wird, eine RAM-Kassette zu benutzen, die eine Vibrations-Erzeugungsschaltung **52** beinhaltet oder eine Vibrationskassette **50A** zu benutzen, welche eine Vibrations-Erzeugungsschaltung **52** ohne RAM **51** umfaßt.

[0081] Die ROM-Kassette **20** ist aus einem externen ROM **21**, welcher auf einer Leiterplatte befestigt ist, aufgebaut, um so die Leiterplatte innerhalb eines Gehäuses davon unterzubringen. Der externe ROM **21** ist mit Bilddaten oder Programmdaten, die bildverarbeitend sind, für ein Spiel oder dergleichen und Klangdaten wie von Musik, Effektklängen oder Nachrichten, soweit erforderlich, gespeichert.

[0082] Fig. 5 ist eine Speicherabbildung, welche darstellerisch den gesamten Speicherplatz des externen ROM **21** darstellt, während Fig. 6 eine Speicherabbildung ist, die im Detail einen Teil (Bildanzeige-Datenbereich **24**) des Speicherplatzes des externen ROM **21** zeigt. Der ROM **21** beinhaltet eine Vielzahl von Speicherbereichen (hierin wird der „Speicherbereich" als „Bereich" bezeichnet, wenn er mit einem Datenartnamen, der davor gesetzt ist, benutzt wird), z. B. wie in Fig. 5 gezeigt, ein Programmbereich **22**, ein Zeichencodebereich **23**, ein Bilddatenbereich **24** und ein Klangspeicherbereich **25**, wodurch verschiedene Programme vorher fest gespeichert sind.

[0083] Der Programmbereich **22** ist mit Programmen gespeichert, die zum Durchführen der Bildverarbeitung erforderlich sind, wie für ein Spiel (Programme zum Realisieren der Funktionen der Flußdiagramme, die in Fig. 15 bis Fig. 31 gezeigt werden, die hierin aufgeführt sind, oder Spieldaten, die mit einem Spielinhalt konform sind, oder dergleichen). Insbesondere beinhaltet der Programmbereich **22** Speicherbereiche **22a**–**22p** zum vorherigen Speichern von Betriebsprogrammen für die CPU **11** in einer festgesetzten Weise. Ein Hauptprogrammbereich **22a** wird mit einem Hauptroutine-Verarbeitungsprogramm für ein Spiel, welches in Fig. 15 gezeigt wird, oder dergleichen gespeichert. Steuerungsauffülldaten (Bedienungszustand), die den Programmbereich **22b** bestimmen, sind mit einem Programm zum Verarbeiten der Daten, welche einen Bedienungszustand etc. der Steuerungseinrichtung **40** darstellen, gespeichert. Ein Schreibprogrammbereich **22c** ist mit einem Schreibprogramm gespeichert, durch welches die CPU **11** den RCP **12** dazu veranlaßt, in einen Rahmenspeicher und einen Z-Puffer zu schreiben. Z. B. wird der Schreibprogrammbereich **22c** mit einem Programm zum Schreiben von Chrominanz-Daten als Bilddaten basierend auf Textdaten für eine Vielzahl von bewegbaren Objekten oder Hintergrundobjekten, die in einer Hintergrundszene angezeigt werden, in einen Rahmenspeicherbereich (**152**, in Fig. 7 gezeigt) des RAM **15** gespeichert, und eine Programm zum Schreiben von Tiefendaten in einen Z-Pufferbereich (**153**, der in Fig. 7 gezeigt ist). Ein Bewegungsprogrammbereich **22d** ist mit einem Steuerungsprogramm gespeichert, durch welches die CPU **11** den RCP **12** dazu veranlaßt, die Position des bewegbaren Objektes in einem dreidimensionalen Raum zu verändern. Ein Kamera-Steuerungsprogramm-Bereich **22e** ist mit einem Kamerasteuerungsprogramm gespeichert, durch welches die Steuerung bezüglich welcher Position und Richtung das bewegbare Objekt, welches das Spielerobjekt oder das Hintergrundobjekt einschließt, in dem dreidimensionalen Raum photographiert wird, durchgeführt wird. Ein Ablauf-Wählprogramm-Bereich **22f** ist mit einem Ablauf-Wähl-Subroutine-Programm, wie in Fig. 16 gezeigt, um später darauf zurückzukommen, gespeichert. Ein Modusshift-Programmbereich **22g** ist mit einem Modusshift-Subroutine-Programm, wie in Fig. 17 gezeigt, um später darauf zurückzukommen, gespeichert. Das Programm, welches in dem Speicherbereich **22g** gespeichert ist, dient dazu, die Richtung und den Bereich des Scrolls durch Verschieben des Scroll-Modus zwischen einer eindimensionalen Scrollanzeige und dem Scrollen in alle Richtungen (alle Bereiche) zu verändern.

[0084] Ein Kommunikationsverarbeitungs-Verarbeitungsprogrammbereich **22h** ist mit einem Kommunikationsverarbeitungs-Subroutine-Programm, wie in Fig. 19 bis Fig. 21 gezeigt, worauf zurückgekommen wird, gespeichert. Ein Nachfüll-Verarbeitungsprogrammbereich **21e** ist mit einem Nachfüllverarbeitungs-Subroutine-Programm, wie in Fig. 22 bis Fig. 23 gezeigt, worauf zurückgekommen wird, gespeichert. Ein Spielerobjekt-Programmbereich **22j** ist mit einem Programm zum Anzeige-Steuern des Objektes, welches von dem Spieler bedient wird, gespeichert. Ein Mitglieder-Objektprogramm-Bereich **22k** ist mit einem Programm (siehe Fig. 24 bis Fig. 26) zum Anzeige-Steuern eines Mitgliedobjektes, welches mit einem Spiel in Kooperation mit dem Spielerobjekt einhergeht, gespeichert. Ein Hindernisobjekt-Programmbereich **221** ist mit einem Programm (siehe Fig. 27 und Fig. 28) zum Anzeige-Steuern eines Hindernisobjektes, welches das Spielerobjekt angreift, gespeichert. Ein Hintergrund-Programmbereich **22m** ist mit einem Hintergrund-Erschaffungsprogramm (siehe Fig. 29) gespeichert, durch welches die CPU **11** den RCP **12** dazu veranlaßt, ein dreidimensionales Hintergrundbild (oder Ablauf) zu erschaffen. Ein Klangverarbeitung-Programmbereich **22n** ist mit einem Programm (siehe Fig. 31) zum Erzeugen einer Nachricht von einem Effektklang oder einer Musik oder Stimmen gespeichert. Ein Spiel-Vorbei-Verarbeitungsprogrammbereich **220** ist mit einem Programm für einen Ablauf gespeichert, der auszuführen ist, wenn ein Spiel beendet ist, z. B. durch Detektieren eines Zustandes eines Spiel-Vorbei oder Sichern der Backup-Daten in einem Zustand des Spieles bis dahin, bevor das Spiel vorbei ist. Ein Nachrichten-Verarbeitungsprogramm-Bereich **22b** ist mit einem Subroutine-Programm zum Verarbeiten einer Nachricht (Kommunikationsprozeß in Fig. 19 bis Fig. 21, die Prozesse beinhalten einer Lieferung der Nachfüllmaterialien in Fig. 22 und Fig. 23) gespeichert, um durch Anzeigen von Zeichen oder Ausgeben von Klängen eine Nachricht auszugeben, die hilfreich zum Bedienen ist, welches geeignet für den Standort oder die Umgebung, in der das Spielerobjekt steht.

[0085] Ein Literalcodebereich **23** ist ein Bereich zum Speichern einer Vielzahl von Arten von literalen Codes, welche mit literalen Punktdaten in einer Vielzahl von Arten korrespondierend zu einem Code gespeichert sind. Die literalen Codedaten, welche in dem literalen Codebereich **23** gespeichert sind, werden zum Anzeigen einer erläuternden Anmerkung für den Spieler in dem Ablauf eines Spieles benutzt. In dieser Ausführungsform wird dieser Bereich benutzt, um eine literale Nachricht (oder Phrase) betreffend einer passenden Handhabungsart oder dem Antwortverfahren zur passenden Zeit in Konformität zu der umgebenden Umgebung, in der das Spielerobjekt (z. B. Standort, Hindernisart, Feindobjektart) oder der Situation, in der das Spielerobjekt plaziert ist, anzuzeigen. Ein Bilddatenbereich **24** beinhaltet Speicherbereiche **24a**–**24f** wie in Fig. 6 gezeigt. Der Bilddatenbereich **24** ist mit jeweiligen Koordinaten, Textdaten etc. von einer Vielzahl von Polygonen für jedes der Hintergrundobjekte und/oder der bewegbaren Objekte und einem Anzeigesteuerungsprogramm zum Anzeigen dieser Objekte stationär bei vorbestimmten Orten oder beim Umherbewegen gespeichert. Z. B. ist der Speicherbereich **24a** mit einem Programm zum Anzeigen der Spielerobjekte gespeichert. Der Speicherbereich **24b** ist mit einem Mitgliedsobjekt-Programm zum Anzeigen einer Vielzahl von Mitgliedsobjekten 1–3 gespeichert. Der Speicherbereich **24c** ist mit einem Hintergrundobjekt-Programm zum Anzeigen einer Vielzahl von Hintergrund(stationäre)-Objekte 1–n1 gespeichert. Der Speicherbereich **24d** ist mit einem Feindobjekt-Programm zum Anzeigen einer Vielzahl von Feindobjekten 1–n2 gespeichert. Der Speicherbereich **24d** ist mit einem Chefobjekt-Programm zum Anzeigen eines Chefobjektes gespeichert. Der Speicherbereich **24f** ist mit Daten zum Ausgeben einer Phrase oder Nachricht, z. B. wie in Fig. 12 gezeigt, auf welche später zurückgegriffen werden wird, gespeichert.

[0086] Ein Klangspeicherbereich **25** ist mit Klangdaten, wie Phrasen, Effektklängen und Spielmusik, zum Ausgeben einer Nachricht wie oben bei dem Klang in einer Weise passend zu der Szene, welche begleitet werden soll, gespeichert.

[0087] Nebenbei gesagt, kann die externe Speichervorrichtung andere Speichermedien wie eine CD-ROM und eine magnetische Diskette anstelle der ROM-Kassette **20** oder zusätzlich zu der ROM-Kassette **20** verwenden. In solch einem Fall ist ein Plattenantrieb (Record-Wiedergabevorrichtung) **26** angeordnet, um zu lesen und, falls erforderlich, verschiedene Daten zu schreiben (einschließlich Programmdaten und Bildanzeigedaten) aus einem optischen oder magnetischen Speichermedium vom Plattentyp, wie eine CD-ROM oder eine magnetische Platte. Der Plattenantrieb **26** liest Daten aus einer magnetischen oder optischen Platte, die magnetisch oder optisch Programmdaten ähnlich zu der von dem ROM **21** speichert, um die Daten an den RAM **15** zu übertragen.

[0088] Fig. 7 ist eine Speicherabbildung, die darstellerisch einen gesamten Speicherplatz des RAM **15** zeigt, während Fig. 8 eine Speicherabbildung ist, die im Detail einen Teil (Bildanzeige-Datenbereich **154**) des Speicherplatzes von dem RAM **15** zeigt. Der RAM **15** beinhaltet verschiedene Speicherbereiche **150**–**159**. Z. B. beinhaltet der RAM **15** einen Anzeigelistenbereich **150**, einen Programmbereich **151**, einen Rahmenspeicher (oder Bild-Pufferspeicher)-Bereich **152** zum zeitlichen Speichern- eines Rahmens von Bilddaten, einen Set-Puffer-Bereich **153** zum Speichern von Tiefendaten auf ei-

ner Punkt-für-Punkt-Basis in dem Rahmenspeicherbereich, einen Bilddatenbereich **154**, einen Klangspeicherbereich **155**, einen Bereich **156** zum Speichern von Bedienungszustands-Daten auf der Steuerungsanschlußfläche, einen Arbeitsspeicherbereich **157**, einen Mitglieddatenbereich **158** und einen Registermarkierungsbereich **159**. Die Speicherbereiche **151**–**159** sind Speicherplätze, die durch die CPU **11** über die Bussteuerungsschaltung **121** oder direkt durch den RCP **12** zugreifbar sind, so daß sie eine willkürliche Kapazität (oder Speicherplatz) haben, die abhängig von einem Spiel, welches verwendet wird, eingesetzt wird. Der Programmbereich **151**, der Bilddatenbereich **154** und der Klangspeicherbereich **155** speichern zeitlich einen Teil der Daten, z. B. ein Spielprogramm, welches für einen bestimmten Ablauf oder Abschnitt erforderlich ist, von einem Ein-Spiel-Alle-Szenen(oder Abschnitt)-Spielprogramm, welches in den Speicherbereichen **22**, **24**, **25** des ROM **21** gespeichert ist, wenn es dorthin übertragen wird. Wenn ein Teil der Programmdaten, die für eine bestimmte Szene erforderlich sind, in den Speicherbereichen **151**, **154** und **155** auf diese Weise gespeichert werden, wird die CPU **11** in ihrer Wirksamkeit mehr erhöht, als bei dem Verarbeiten durch direktes Auslesen der ROM **21** jedesmal, wenn es von der CPU **11** erforderlich ist, wodurch die Bildverarbeitungsgeschwindigkeit gesteigert wird. Insbesondere hat der Rahmenspeicherbereich **152** eine Speicherkapazität korrespondierend zu der Anzahl der Bildelemente (Pixel oder Punkte) auf der Anzeige **30** X die Anzahl der Chrominanz-Datenbits pro einem Bildelement, so daß er chrominante Daten für jeden Punkt korrespondierend zu den Bildelementen auf der Anzeige **30** speichert. Der Rahmenspeicherbereich **152** speichert zeitlich die Chrominanzdaten auf einer Punkt-für-Punkt-Basis eines Objektes, welches von dem Auge des Beobachters gesehen werden kann, basierend auf den dreidimensionalen Koordinaten zum Darstellen in einem Satz von einer Vielzahl von Polygonen, eines oder mehrere der stationären Objekte und/oder der beweglichen Objekte, welche in dem Bilddatenbereich **154** gespeichert sind, um in einer Hintergrundszene in einem Bildverarbeitungsmodus angezeigt zu werden. Ebenso speichert der Rahmenspeicherbereich **152** zeitlich in einem Anzeigemodus die Chrominanzdaten auf einer Punkt-für-Punkt-Basis, wenn verschiedene Objekte, welche in dem Bilddatenbereich **154** einschließlich der beweglichen Objekte, d. h. dem Spielerobjekt, den Mitgliedsobjekten, den Feindobjekten, den Chefobjekten oder den Hintergrund(oder stationären)-Objekten etc. gespeichert sind, angezeigt werden.

[0089] Der Z-Pufferbereich **153** hat eine Speicherkapazität korrespondierend zu der Anzahl der Bildelemente (Pixel oder Punkte) der Anzeige **30** X die Anzahl der Bits von den Tiefendaten pro einem Bildelement, so daß er die Tiefendaten auf einer Punkt-für-Punkt-Basis korrespondierend zu einer Weise der Bildelemente der Anzeige **30** speichert. Der Z-Pufferbereich **153** speichert in dem Bildverarbeitungsmodus zeitlich die Tiefendaten für jeden Punkt des Objektes, das von den Augen des Beobachters gesehen werden kann, basierend auf den dreidimensionalen Koordinatendaten zum Darstellen in einem Satz von einer Vielzahl von Polygonen, einem oder mehreren der stationären Objekte und/oder der beweglichen Objekte. Der Z-Pufferbereich **153** speichert zeitlich ebenso in dem Anzeigemodus die Tiefendaten für jeden Punkt der beweglichen und/oder stationären Objekte.

[0090] Der Bilddatenbereich **154** speichert die Koordinatendaten und die Textdaten für Polygone in einer Vielzahl von Sätzen, von welchen jeder aus den stationären und/oder beweglichen Objekten, die gespeichert sind, besteht, für die Spielanzeige in dem ROM **21**, so daß mindestens eines der Ablauf- oder Abschnittsdaten dorthin von dem ROM **21** übertragen werden, bevor eine Bildverarbeitungsfunktion stattfindet. Das Detail der gespeicherten Daten in dem Bilddatenbereich **154** wird mit Bezug auf Fig. 8 erklärt werden.

[0091] Der Klangspeicherbereich **155** wird durch einen Teil der Klangdaten (Daten der Phrasen, Musik und Effektklänge), die in dem Speicherbereich des ROM **21** gespeichert sind, übertragen, die zeitlich als Klangdaten, die durch die Klangerzeugungsvorrichtung **32** zu erzeugen sind, gespeichert sind.

[0092] Die Steuerungsanschlußdaten(Bedienungszustandsdaten)-Speicherbereich **156** speichert zeitlich Bedienungszustandsdaten, die einen Bedienungszustand darstellen, der von der Steuereinrichtung **40** ausgelesen wird.

[0093] Der Arbeitsspeicherbereich **157** speichert zeitlich Daten, wie Parameter, während des Ausführens eines Programms durch die CPU **11**.

[0094] Der Mitgliedsdatenbereich **158** speichert zeitlich Daten zum Anzeigesteuern des Mitgliedsobjektes, welches in dem Speicherbereich **22k** gespeichert ist.

[0095] Der Registermarkierungsbereich **159** beinhaltet eine Vielzahl von Registerbereichen **159R** und eine Vielzahl von Markierungsbereichen **159F**. Der Registerbereich **159R** beinhaltet Register R1–R3 zum Laden mit den Beträgen der jeweiligen Schäden an einem Hauptkörper, einem linken Flügel und einem rechten Flügel, ein Register R4 zum Laden des Schadens an dem Mitglied, ein Register R5 zum Laden des Schadens an dem Feind (Chef), ein Register R6 zum Laden der Anzahl der Spielerobjekte, ein Register R7 zum Laden der Lebenszahl der Spieler, ein Register R8 zum Laden der Anzahl der Feindobjekte, die auf einer Szene anzuzeigen sind, ein Re-

gister R9 zum Laden der Anzahl der stationären Objekte, ein Register R9 zum Laden der Punktezahl in einem Ablauf, der gespielt wird, die Register R11–R1n zum Laden der Punktezahlen für den Ablauf 1–n, ein Register R20 zum Laden der Gesamtpunkte und ein Register **21** zum Laden eines höchsten Punktes. Der Markierungsbereich **159F** ist ein Bereich zum Speichern einer Markierung, durch welche der Zustand des Spieles während des Ablaufs bekannt ist, und beinhaltet z. B. eine Mitgliedsmarkierung F1, eine Modusmarkierung F2 zum Unterscheiden eines Modus des Anzeigebereiches, die Phrasenmarkierungen F31–F3M zum Speichern, ob eine Phrase 1–m ausgegeben werden sollte oder nicht, eine Spiel-Vorbei-Markierung F4 zum Unterscheiden der Anwesenheit oder Abwesenheit der Detektion von einem Zustand des Erreichtseins eines Spielendes und Trefferzustandsmarkierung F5.

[0096] Fig. 9 ist eine Ansicht, die ein Beispiel von Spielabläufen zeigt, auf welche die vorliegenden Erfindung angewendet wird. Fig. 10 ist eine Ansicht, die einen Ablauf-Auswahlbildschirm für das Spiel, welches in Fig. 9 gezeigt wird, zeigt. Fig. 11 ist ein Diagramm, welches eine Spielbereichabbildung zum Erklären eines Beispieles eines Spielgehaltes zeigt, auf welchen die vorliegende Erfindung angewendet wird. Fig. 12 ist eine darstellerische Ansicht, welche einen Nachrichtenausgabeinhalt in einem Ablauf der Kommunikation mit einem Mitglied in dem Spiel von Fig. 11 zeigt. Fig. 13 ist ein Diagramm, welches ein Beispiel einer Darstellung auf dem Bildschirm von einer Nachrichtenausgabe zeigt, die basierend auf dem Kommunikationsablauf mit dem Mitglied in dem Spiel von Fig. 11 dargestellt wird. Fig. 14 ist eine Ansicht, die ein Beispiel von einer Darstellung auf einem Bildschirm in einem Zustand des Kriegführens gegen ein Chefzeichen in dem Spiel von Fig. 11 zeigt. Mit Bezug auf die Fig. 9 bis Fig. 14 werden Erklärungen auf der Außenseite des Videospieles gemacht, um eine hilfreiche Nachricht zum Vorwärtskommen in dem Spiel auszugeben, welches ein Merkmal der vorliegenden Erfindung darstellt. Obwohl der Spielgehalt des Videospieles durch das Programm bestimmt wird, welches in dem ROM **21** gespeichert wird, wird ein Beispiel eines Schießspieles in der Ausführungsform gezeigt werden. Als Start des Spieles werden die Abläufe, die in Fig. 9 gezeigt werden, angezeigt. In Fig. 9 ist die Anzeige für die deutlichen Ablauf-Anzeigebereiche **81a**–**81e** unterhalb eines Ablauf-Anzeigebereiches **80** gegeben, welches einen verdeutlichten Zustand auf einer Ablauf-für-Ablauf-Basis anzeigt. Ein Ablauf-Punkte-Anzeigebereich **82** und ein Höchstpunkte (Toppunkte)-Anzeigebereich **83** wird in einem oberen Anteil des Bildschirms angezeigt.

[0097] Bei einem anfänglichen Abschnitt des Spieles wird, wenn ein Ablauf 1 ausgewählt wird, eine Szene eines Startpunktes von Fig. 11 angezeigt, wie es in Fig. 12 gezeigt wird. Die lange Entfernung von einem Startpunkt zu einem Modusverschiebungspunkt, wie in Fig. 11 gezeigt (z. B hunderttausend in einer tiefen Koordinateneinheit; die Einheit ist willkürlich), wird als ein Anzeigebereich für einen Ein-Richtungs-Ablaufmodus ausgewählt. Der Anzeigebereich des Ein-Richtungs-Ablaufmodus hat eine Breite, die als dieselbe wie die Bildschirmgröße ausgewählt wird, die auf einem Anzeigebildschirm **31** der Anzeige **30** angezeigt werden kann, welche für die Ablaufanzeige von dem oberen zu dem unteren Ende verwendet wird. In dem Anzeigebereich des Ein-Richtungs-Ablauf-Modus wird die Anzeige in Reihenfolge durchgeführt für die Objekte **71**–**71n** (siehe Fig. 12), die die Hintergrundbilder darstellen, wie Gebäude, Bäume, Berge, Straßen und Himmel, welche die Hintergründe oder die stationären Objekte auf dem Ablauf darstellen. Bei vorbestimmten Punkten A–D in dem Mittelweg davon treten eine Vielzahl von Feindobjekten **72a**–**72n** auf, um ein Spielerobjekt **60** zu attakkieren oder das Spielerobjekt **60** vom Vorwärtskommen abzuhalten.

[0098] Die Plätze A, B, C, D in dem Ablauf des Anzeigebereiches des Ein-Richtungs-Ablaufmodus werden als Standorte bestimmt, um in der Anzeige oder als Klang eine Nachricht (oder Phrase) auszugeben, um den Spieler wissen zu lassen, welches ein passender Handhabungsweg ist, oder dem Spielerobjekt **60** zu assistieren, um die Feindobjekte **72a**–**72n** zu vertreiben oder ihre Attacken geschickt zu vermeiden. Wie in Fig. 12 gezeigt wird, wird eine Nachricht auf dem Anzeigebereich **31a** angezeigt. Das Gesicht eines Mitgliedes, welches die Nachricht sendet, wird auf dem Anzeigebereich **31b** angezeigt. Die Punktezahl während des Spielens wird in dem Anzeigebereich **31c** angezeigt und das Leben (der Betrag, der durch das Widerstehen gegenüber den Zerstörungen erhältlich ist) wird auf dem Anzeigebereich **31d** angezeigt.

[0099] Die Nachrichten werden als konkrete Beispiele darstellerisch in Fig. 13 gezeigt.

[0100] Unter einer Vielzahl von Nachrichten wird ein Nachrichtensatz für einen Standort durch das Programm in dem Anzeigebereich **31a** angezeigt. In diesem Spielbeispiel wird ein Fall gezeigt, in dem Phrasen in einem Klang oder Bild auf verschiedene Arten abhängig von der Art und der Szene einer Person oder dem Eintrittszeichen ausgegeben werden, so daß eine Nachricht durch ein Mitglied bezüglich eines Handhabungsverfahrens gelehrt wird, welches für die Situation im Bezug auf das Phrasenauftreten geeignet ist. Die Phrasen 1–9 werden in Prioritätsordnung bestimmt, so daß, wenn ein Zustand des Erzeugens einer Vielzahl von Phrasen zu der gleichen Zeit detektiert wird, eine höhere Präferenzordnung einer Phrase erzeugt wird. In Beziehung auf die Anzeige einer Nachricht wird ein Gesicht des Mitgliedsobjektes **73**, welches die Nachricht sendet, angezeigt.

Die Nachricht beinhaltet ein Steuerungsverfahren (eine Nachricht „darüber hinwegkommen mit Bremsen" für Durchführen der Bremsung), worin das Spielerobjekt **60** als ein Kämpfer angenommen wird, und ein Handhabungsweg, der besagt, welcher Schalter in welcher Weise auf der Steuerungseinrichtung **40** zum Erreichen des Steuerungsverfahrens bedient werden sollte (eine Nachricht „C-Knopf nach unten" zeigt das Drücken des Knopfes **47Cd** an: vorzugsweise Anzeigen einer unterschiedlichen Farbe der niedrigeren Knöpfe unter den vier Knöpfen, die an der oberen, unteren, linken und rechten Seite angeordnet sind). Zusätzlich wird zu der Nachrichtenanzeige das Ausgeben des Klanges („darüber hinwegkommen mit Bremsen") ebenso durchgeführt, falls erforderlich. An dem Standort C wird eine Nachricht erzeugt „zweimal drücken von entweder Z oder R", welches besagt, daß der Schalter **47Z** oder **47R** zweimal gedrückt wird. In dieser Weise ist der Nachrichteninhalt durch den Standort A–D abhängig von der Form oder der Bewegung des Feindobjektes unterschiedlich. Wenn der Spieler den Joystick **45** handhabt, um die Position und die Richtung des Spielerobjektes **60** zu steuern und einen Schaltern von den Schaltern **47A**–**47Z** bedient, in Konformität zu dem Nachrichtenausgang, so kann er leicht eine passende Bedienung durchführen, sogar wenn die Zahl der Schalter eine hohe ist und die passende und schnelle Schalterbedienung schwierig oder unmöglich ist. Es ist möglich, den Feind zu attackieren oder Krisen durch schnelles Durchführen der angezeigten Bedienung zu vermeiden. Der Spieler wird, sogar wenn nicht getötet, leicht zu den nächste Szenen vorwärtsgebracht.

[0101] Wenn das Spielerobjekt **60** einen Modusverschiebe-Punkt erreicht, so wird der Anzeigemodus in einen All-Bereich-Modus verschoben, in welchem das Ablaufen in jede Richtung möglich ist. In dem All-Bereich-Modus ist ein Chef-Zeichen (Chef-Objekt) in einem Zentrum eines anzeigbaren Bereichs plaziert, so daß das Spielerobjekt **60** Angriffe auf das Chef-Zeichen **74** machen kann, während es sich darum herumdreht.

[0102] Der Bereich, über welchen das Spielerobjekt **60** sich ungefähr bewegen kann, wird in einer kurzen Entfernung ausgewählt (z. B. 100.000) in Richtung der oberen, unteren, linken und rechten Seite mit Hinblick auf das Chef-Zeichen **74**. Wenn das Spielerobjekt **60** nahe einer Grenze des Bewegungsbereichs kommt, so wird die Kamera, die das Spielerobjekt **60** fotografiert, in der Richtung wechseln, um dadurch automatisch die Bewegungsrichtung des Spielerobjekts **60** zu verändern. Zu dieser Zeit wird eine größenreduzierte Abbildung in dem Abbildungs-Anzeigebereich **31c** in der unteren rechten Seite des Anzeigebildschirms **31** angezeigt, so daß der Spieler leicht die Position, an der das Spielerobjekt **60** steht, erfahren kann. Die Abbildung beinhaltet die Anzeige der Symbole der Chef-Zeichen **74**, des Spielerobjekts **60** und des Mitgliedobjekts **73**.

[0103] Fig. 15 ist ein Hauptflußdiagramm eines Videospielsystems von einer Ausführungsform der vorliegenden Erfindung. Mit Bezug auf Fig. 9–Fig. 15 werden Erklärungen kurz über das Prinzip dieser Erfindung mit Hilfe des Hauptflußdiagramms in Fig. 15 gemacht. Durch Einschalten einer Versorgung setzt die CPU **11** während des Startens das Videospielgerät **10** auf einen vorbestimmten Anfangszustand. Z. B. überträgt die CPU **11** ein Startprogramm unter den Spielprogrammen, die in dem Programmbereich des ROM **21** gespeichert sind, an den Programmbereich **151** des RAM **15**, um Parameter auf ihre Anfangswerte zu setzen, und danach führt sie in Reihenfolge die Abläufe des Flußdiagramms von Fig. 15 durch.

[0104] Der Flußbetrieb von Fig. 15 wird bei jedem 1-Rahmen (1/30 Sekunden) ausgeführt. Bevor der Ablauf gelöscht wird, werden ein Schritt 1 (mit „S" in der Figur bezeichnet), ein Schritt 2 und ein Schritt 3–Schritt 17 ausgeführt und danach ein Schritt 3–Schritt 17 in der Funktion wiederholt. Wenn das Spiel ohne Erfolg des Löschens des Ablaufs beendet wird, so wird ein Spiel-Vorbei-Prozeß von einem Schritt 18 ausgeführt. Wenn das Ablauflöschen erfolgreich durchführt wird, so geht der Prozeß von dem Schritt 16 zu dem Schritt 1 zurück.

[0105] Das heißt, obwohl bei dem Schritt 1 die Anzeige für die Spiel-Ablaufszene und/oder Spiel-Auswahlszene durchgeführt wird, wird eine Ablaufszene, wie in Fig. 9 gezeigt, angezeigt, wenn das Spiel gestartet wird, nachdem die Versorgung eingeschaltet wird. Nebenbei gesagt, wird nachdem zu dem Ablauf 2 vorangeschritten wird, durch Löschen des Ablaufs 1, wie es in Fig. 9 gezeigt wird, und der Ablauf 2 ebenso gelöscht wird, ein Ablauf-Auswahl-Bildschirm, wie in Fig. 10 gezeigt, angezeigt. Wenn ein Ablauf auf dem Ablauf-Auswahl-Bildschirm ausgewählt wird, so wird eine Ablauf-Auswahl Subroutine, wie in Fig. 16 gezeigt, (Funktionen von den Schritten 101–116), ausgeführt. Dies stellt jedoch nicht einen wesentlichen Teil der Erfindung dar, und deshalb wird ein Flußdiagramm nur gezeigt, um dadurch detaillierte Bedienungserklärungen wegzulassen.

[0106] Da das Spiel von dem Ablauf 1 sofort nach dem Start durchgeführt wird, wird der Spiel-Start-Prozeß für den Ablauf bei Schritt 2 durchgeführt. Z. B. wird der Registerbereich **159R** und der Markierungsbereich **159F** gelöscht (anfängliche Werte werden für die Register R6, R7 gesetzt) und verschiedene Daten, die zum Durchführen des Spiels für den Ablauf 1 (oder einem ausgewählten Ablauf) erforderlich sind, werden von dem ROM **21** aus gelesen und zu den Speicherbereichen **151**–**155** des RAM **15** übertragen.

[0107] Bei dem Schritt 3 wird ein Modus-Verschiebe-Subroutinen-Prozeß durchgeführt. Das Spielerobjekt **60** ist an dem Startpunkt in Fig. 11 unmittelbar nach dem Starten des Spiels anwesend. Jedoch liegt die Zeitperiode zwischen dem Startpunkt (Z-Koordinate = 0) und dem Modus-Verschiebungspunkt (Z-Koordinate = –hunderttausend) in dem Ein-Richtungs-Ablaufmodus. Dementsprechend wird bei einem Schritt 121 in Fig. 17 bestimmt, daß das Spielerobjekt nicht an der All-Bereichs-Modusposition anwesend ist, also daß eine Markierung F2 bei einem Schritt 122 zurückgesetzt wird, um die Verschiebung zu dem Ein-Richtungs-Ablaufmodus durchzuführen, und dann fährt der Prozeß fort zu dem nächsten Schritt 4. Die detaillierte Funktion davon wird später mit Bezug auf Fig. 17 beschrieben werden.

[0108] Bei dem Schritt 4 wird ein Steuerungseinrichtungsprozeß durchgeführt. Dieser Prozeß wird durch Detektieren bezüglich ob entweder der Joystick **45**, der Kreuzschalter **46** oder der Schalter **47A**–**47Z** der Steuerungseinrichtung **40** bedient wird oder nicht, durchgeführt. Die detektierten Daten (Steuerungseinrichtungsdaten) in dem Bedienungszustand werden gelesen, und die Lese-Steuerungseinrichtungsdaten werden geschrieben. Die detaillierte Funktion wird später mit Bezug auf Fig. 18 beschrieben werden.

[0109] In dem Schritt 5 wird ein Prozeß der Kommunikation mit einem Mitglied durchgeführt. Dieser Prozeß wird durch Anzeigen oder Klangausgeben einer Nachricht durchgeführt, welche einen erfassenden Handhabungsweg mitteilt, welcher ein Merkmal der vorliegenden Erfindung darstellt. Daß heißt, der Spieler ist im Wissen eines passenden Handhabungswegs an dem Standort A–D in der Ein-Richtungs-Ablaufperiode, die in Fig. 11 gezeigt wird, durch Angezeigt werden oder Klangausgeben von einer Nachricht oder einer Phrase, wie es in Fig. 13 gezeigt wird. Die detaillierte Funktion davon wird mit einem Beispiel später mit Bezug auf die Fig. 19–Fig. 21 beschrieben werden. Nebenbei gesagt, ist es hervorzuheben, daß der Inhalt oder der Erscheinungszustand einer Nachricht nur ein Beispiel ist und abhängig von dem Inhalt oder der Art eines Spiels unterschiedlich ist und für den Gebrauch modifiziert werden kann.

[0110] In dem Schritt 6 wird ein Nachfüllprozeß durchgeführt, um Material von einem Hauptquartier zu liefern. In diesem Prozeß werden Punkte zu Assistieren des Spielers (z. B. Teile zum Reparieren eines Flügels eines Fliegers, Feuerwaffen, Gewehre etc.) von dem Hauptquartier oder einem Mitglied gesendet, da, wenn das Spielerobjekt **60** von einem Feind angegriffen wird oder ein Rumpf Schaden erleidet und der normale Flug unmöglich ist. Wenn ein solcher Punkt auf dem Bildschirm angezeigt wird, wenn der Spieler eine Bedienung durchführt, um dasselbe zu erwerben (Überlappen des Rumpfes über den Punkt, Anstoßen des Punkts durch Schuß etc.), so kann der beschädigte Anteil in einem früheren Zustand wiederhergestellt werden oder ein Punkt, der zum Angreifen des Feinds vorteilhaft ist, kann dafür angeboten werden. In diesem Fall, wird, da sich der Punkt, welcher durch den Spieler erforderlich ist, abhängig von einem Zustand des Schadens auf das Spielerobjekt, unterscheidet, die Art des Punkts automatisch entsprechend einer vorbestimmten Referenzordnung bestimmt. Die detaillierte Funktion wird später mit Bezug auf die Fig. 22 und Fig. 23 beschrieben werden.

[0111] In dem Schritt 7 wird ein Prozeß zum Anzeigen des Spielerobjekts **60** ausgeführt. Obwohl der Prozeß abhängig davon, ob das Spielerobjekt **60** in einem der Einrichtungs-Ablaufbereiche und dem All-Bereichs-Bereich existiert, sich unterscheidet, ist er grundsätzlich ein Prozeß des Änderns der Richtung oder der Form basierend auf dem Bedienungszustand des Joysticks **45**, der von dem Spieler bedient wird, und der An- oder Abwesenheit von einem Angriff des Feinds. Z. B. wird die Steuerung in der Anzeige des Spielerobjekts **60** durch Berechnen nach dem Verändern der polyphonen Daten, die auf dem Programm, welches von dem Speicherbereich **22j** übertragen wird, basieren, der polygonen Daten des Spielerobjekts, welche von dem Speicherbereich **24a** übertragen werden, und des Bedienungszustands des Joysticks **45** durchgeführt. Die Chrominanzdaten werden in Adressen in dem Speicherbereich **154a** korrespondierend zu einer Oberfläche einer Vielzahl von Dreiecken, die eine Vielzahl von Polygonen darstellen, die somit erhalten werden, um so dadurch ein Muster oder Farbpapier, welches durch die Textdaten bezeichnet wird, darauf zu legen, geschrieben.

[0112] In dem Schritt 8 wird ein Kameraprozeß durchgeführt. Z. B. werden die Koordinatenberechnungen für einen Winkel, in welchem das Objekt gesehen wird, durchgeführt, so daß das Auge des Beobachters oder Sichtstandpunkt, wie er von einem Finder einer Kamera gesehen wird, einen Winkel einnimmt, welcher durch den Spieler bezeichnet wird.

[0113] In dem Schritt 9 wird ein Mitgliedsobjekt-Prozeß durchgeführt. Das Mitgliedsobjekt wird für eine vorbestimmte Positionsrelation zu dem Spielerobjekt in dem Ein-Richtungs-Ablaufbereich berechnet. Z. B. wird das Mitgliedsobjekt nicht angezeigt, wenn es hinter dem Spielerobjekt **60** fliegt, während, wenn das Spielerobjekt **60** abgebremst wird, ein Berechnungsprozeß durchgeführt wird, um anzuzeigen, ob es davor geflogen war. In dem All-Bereichs-Bereich wird, wenn das Mitgliedsobjekt vor dem Spielerobjekt **60** fliegt, es zusammen mit dem Rumpf des Mitglieds durch Symbole in einer größenreduzierten Abbildung angezeigt. Wenn es auf einer Rückseite fliegt, wird es nur durch ein Symbol in der größenreduzierten Abbildung angezeigt. Das Detail wird später mit Bezug auf die Fig. 24–Fig. 26 genauer beschrieben werden.

[0114] In dem Schritt 10 wird ein Prozeß des Feindobjekts durchgeführt. In diesem Prozeß wird die Anzeigeposition und/oder die Form des Feindobjekts **72a**–**72n** durch Berechnen von Polygondaten bestimmt, um ein verändertes Bild anzuzeigen, so daß es sich bewegt, um das Spielerobjekt **60** anzugreifen oder das Voranschreiten desjenigen zu behindern, während die Bewegung des Spielerobjekts **60** beurteilt wird, basierend auf dem Programm, welches teilweise von den Speicherbereichen **221** und **24d** übertragen wird. Aufgrund dieses Umstands wirkt das Feindobjekt in einer Weise, daß es einen bestimmten Effekt auf das Spielerobjekt **60** hat. Das Detail wird später mit Bezug auf die Fig. 27 und Fig. 28 beschrieben werden.

[0115] In dem Schritt 11 wird ein Prozeß des Hintergrunds-(stationären)-Objekts durchgeführt. In diesem Prozeß wird die Anzeigeposition und die Form des stationären Objekts **71a**, **71n** durch Berechnen basierend auf dem Programm, welches teilweise von dem Speicherbereich **22n** übertragen wird und den Polygondaten des stationären Objekts, welche von dem Speicherbereich **24c** übertragen werden, bestimmt. Das Detail wird später mit Bezug auf Fig. 29 beschrieben werden.

[0116] In dem Schritt 12 führt der RSP **122** einen Wiedergabeprozeß durch, d. h. der RCP **12** führt unter der Steuerung der CPU **11** einen Umformungsprozeß (Koordinationumformungsprozeß und Rahmenspeicher-Wiedergabeprozeß) auf den Bilddaten zum Anzeigen des beweglichen Objekts und des stationären Objekts basierend auf den Textdaten für das bewegliche Objekt, wie dem Feind, dem Spieler, dem Mitglieder- und dem stationären Objekt, wie dem Hintergrund, der in dem Bilddaten-Bereich **154** des RAM **15** gespeichert ist, durch. Insbesondere werden die Chrominanzdaten in Adresse in dem Speicherbereich **154d** korrespondieren zu der Ebene des Dreiecks, welches aus einer Vielzahl von Polygonen besteht, für das bewegliche Objekt oder das stationäre Objekt geschrieben, um eine Farbe, die durch die Textdaten, welche für das Objekt bestimmt sind, bezeichnet, wird, festzulegen. Das Detail wird später mit Bezug auf die Fig. 30 näher beschrieben werden.

[0117] In dem Schritt 13 führt der RCP **12** einen Klangprozeß basierend auf den Klangdaten, wie auf einer Nachricht, auf einer Musik und einem Effektklang durch. Das Detail wird später mit Bezug auf Fig. 31 näher beschrieben werden.

[0118] Als ein Ergebnis des Wiedergabeprozesses in dem Schritt 12 durch den RCP **12** werden die Bilddaten, die in dem Rahmenspeicherbereich **152** gespeichert sind, ausgelesen, wodurch das Spielerobjekt, das bewegliche Objekt, das stationäre Objekt, das Feindobjekt etc. in dem Schritt 14 auf dem Anzeigebildschirm **31** angezeigt werden.

[0119] In dem Schritt 15 wird ein Klang, wie eine Musik, ein Effekt-Sound oder eine Sprache durch Lesen der Klangdaten, die durch den Klangprozeß in dem Schritt 13 von dem RCP **12** erhalten werden, ausgegeben.

[0120] In dem Schritt 16 wird bestimmt, ob der Ablauf gelöscht wird oder nicht (Ablauf- Löschungsdetektion). Wenn der Ablauf nicht gelöscht wird, dann wird in dem Schritt 17 bestimmt, ob das Spiel nicht vorbei ist oder nicht. Wenn das Spiel nicht vorbei ist, dann geht der Prozeß zurück zum Schritt 3, wodurch die Schritte 3–17 wiederholt werden, bis der Zustand des Spielvorbeiseiens detektiert wird. Wenn die Detektion für einen Zustand des Spielvorbeiseiens durchgeführt wird, daß z. B. die Zahl der Fehler, die an den Spieler vergeben werden, eine vorbestimmte Anzahl wird oder das Leben des Spielerobjekts durch einen vorbestimmten Betrag verbraucht ist, so wird ein Spielvorbei-Prozeß in dem folgenden Schritt 18 mit einem selektiven Prozeß des Weiterführens des Spiels oder des Sicherns der Backup-Daten ausgeführt. Nebenbei gesagt, wird in dem Schritt 16, wenn der Zustand des Löschens des Ablaufs (z. B. der Chef wird gestürzt etc.) detektiert wird, wird der Ablauf-Löschungsprozeß in dem Schritt 19 durchgeführt und dann geht der Ablauf zurück zu dem Schritt 1. Hier involviert der Ablauf-Löschungsprozeß z. B. das Laden einer registergespeicherten Ablauf-Punktezahl, die in dem Spiel gewonnen wird, unmittelbar vorher auf ein korrespondierendes Ablauf-Punktezahlregister geladen, um so die Ablauf-Punktezahl als ein Ablaufpunkt in Fig. 11 darzustellen. Wenn eine Vielzahl von Abläufen gelöscht wird, so werden die Gesamtpunkte bestimmt und angezeigt. Nebenbei gesagt, kann die Berechnung einer Ablauf-Punktzahl addiert werden, soweit erforderlich, durch einen Bonuspunkt für ein Ablauf-Löschen.

[0121] Erklärungen werden hierin für die detaillierte Funktion für jede Subroutine gemacht werden.

[0122] Mit Bezug auf Fig. 16 werden Erklärungen im Detail zur Ablaufauswahl (Schritt 1 der Hauptroutine), die durch die vorliegende Erfindung gekennzeichnet ist, gemacht werden. In einem Schritt 101 führt die CPU **11** einen Anzeigeprozeß zum Anzeigen der Ablauf-Punktezahl durch, die für jeden Ablauf von dem Spieler erreicht werden und speichert sie in dem Registerbereich **159R** des RAM **15** ab, wie es in Fig. 9 gezeigt wird, in einen Ablauf-Punktezahl-Anzeigebereich **72a**–**72e**. Insbesondere trägt die CPU **11** die Bilddaten, die die Punktezahlen darstellen, in die Anzeigeliste ein. Danach bezieht sich, wenn nichts anderes beschrieben wird, der Anzeigeprozeß als Bildeingang auf die Anzeigeliste. In einem Schritt 102 summiert die CPU **11** über die Ablauf-Punktezahl auf, um einen Anzeigeprozeß durchzuführen, um eine Gesamtpunktzahl in einem Anzeigebereich **72** der Gesamtpunktezahl anzuzeigen. In ei-

nem Schritt 103 führt die CPU **11** einen Anzeigeprozeß aus, um die Anzahl der verwendbaren vorherigen Objekte in einem Anzeigebereich **74** der Flugzeugrumpfzahl anzuzeigen.

[0123] Nebenbei bemerkt, werden ein Ablauf-Anzeigeprozeß oder ein Anzeigeprozeß der hohen Punktezahl und dergleichen, wie es so nicht in der Fig. 16 gezeigt wird, ausgeführt, neben den Anzeigeprozessen in dem Schritt 101, dem Schritt 102 und dem Schritt 103. Der Ablauf-Anzeigeprozeß soll solche Bilddaten, wie gezeigt, in den Ablauf-Anzeigebereich **80** in Fig. 9 durch die CPU **11** eintragen.

[0124] In einem Schritt 104 bestimmt die CPU **11**, ob der Startschalter **47S** gedrückt ist oder nicht. Wenn der Startschalter **47S** nicht gedrückt ist, so schreitet der Prozeß zu einem Schritt 105 voran. Bei dem Schritt 105 bestimmt die CPU **11**, ob der Druckschalter **47A** gedrückt oder nicht gedrückt ist. Wenn der Druckschalter **47A** nicht gedrückt ist, so wird eine Bildverarbeitung und eine Klangverarbeitung, die durchzuführen ist, bei jedem Rahmen durchgeführt. Nachdem ein vorbestimmtes Bild auf der Anzeige **30** angezeigt und ein Klang ausgegeben wird, geht der Prozeß zurück zu dem Schritt 104. Wenn der Druckschalter **47** gedrückt ist, so geht der Prozeß zu einem Schritt 110 voran.

[0125] Bei dem Schritt 104 schreitet auf der anderen Seite, wenn der Startschalter **47S** gedrückt ist, der Prozeß zu einem Schritt 106 voran. Bei dem Schritt 106 führt die CPU **11** einen Anzeigeprozeß für ein Ablauf-Auswahlfenster **75**, wie es in Fig. 10 gezeigt wird, durch. Das Ablauf-Auswahlfenster **75** zeigt mit den drei Punkten „FÜHRE ABLAUF FORT", „ÄNDERE DEN ABLAUF" und "VERSUCHE NOCH EINMAL DEN ABLAUF" mit einem Cursor (Bezeichnungseinrichtung), der zum Bezeichnen, welcher Punkt auszuwählen ist, beweglich angezeigt wird, an. Obwohl der Cursor durch einen Pfeil in Fig. 10 dargestellt wird, kann die Farbe des ausgewählten Punkts invertiert werden oder die Bezeichnung von einem Punkt kann durch ein Bildmuster ebenso praktiziert werden.

[0126] In einem Schritt 107 führt die CPU **11** einen Bewegungsprozeß für das vertikale Bewegen der Bezeichnungseinrichtung durch. Insbesondere detektiert die CPU **11**, ob der Joystick **45** in seine Richtung gezogen oder zurückgedrückt wird. Wenn er gezogen wird, so wird die Bezeichnungseinrichtung abwärts bewegt, während, wenn er gedrückt wird, die Bezeichnungseinrichtung aufwärts bewegt wird. In einem Schritt 108 bestimmt die CPU **11**, ob der Druckschalter **47A** gedrückt ist oder nicht. Wenn der Druckschalter **47A** nicht gedrückt ist, so wird der Bildprozeß und der Klangprozeß für jeden Rahmen durchgeführt. Nachdem ein vorbestimmtes Bild auf der Anzeige **30** angezeigt und ein Klang ausgegeben wird, geht der Prozeß zurück zu dem Schritt 107. Wenn der Druckschalter **47A** gedrückt wird, so fährt der Prozeß zu dem Schritt 109 fort. Bei dem Schritt 109 bestimmt die CPU **11**, welcher Punkt die Bezeichnungseinrichtung durch Drücken des Druckschalters **47A** ausgewählt hat, um zu bestimmen, ob der ausgewählte Punkt „FÜHRE DEN ABLAUF FORT" oder nicht ausgewählt wurde. Wenn der Punkt „FÜHRE DEN ABLAUF FORT" ausgewählt wird, so schreitet der Prozeß zu dem Schritt 110 voran.

[0127] In dem Schritt 110 speichert die CPU **11** die Daten über das Mitgliedsobjekt die gesamten Abläufe in einen Speicherbereich **158** der Mitgliedsdaten von dem RAM **15**. Die Daten des Mitgliedsobjekts werden bei einer Anwesenheit des Mitgliedsobjekts in einem Ablauf unmittelbar zuvor gelöscht. In der vorliegenden Erfindung ist das Mitgliedsobjekt so konfiguriert, daß es von der Kriegsfront abläßt, wenn es schweren Schaden während einer Schlacht erlitten hat. Im Falle, daß das Mitgliedsobjekt von dem Kriegsfront abläßt, kann es, wenn der Schaden schwer ist, nicht an einem nächsten Ablauf teilnehmen. Wenn der Schaden niedrig ist, kann es in dem nächsten Ablauf teilnehmen, wobei dieser Zustand, ob es möglich ist, an dem nächsten Ablauf teilzunehmen oder nicht, ein Mitgliedsobjektszustand ist.

[0128] In einem Schritt 111 führt die CPU **11** ein anfängliches Festsetzen für den nächsten Ablauf durch, welches z. B. das Rahmen-Zurücksetzen, verschiedener Parameter Sätze, das Schreiben verschiedener Objektdaten, die anzuzeigen sind, in den RAM **15** etc. beinhaltet. Wenn der Schritt 111 endet, so geht der Prozeß zurück zu der Hauptroutine, um zu einem Schritt 2 voranzuschreiten.

[0129] Auf der anderen Seite schreitet der Prozeß in dem Schritt 109, wenn der Auswahlpunkt nicht „SCHREITE IN DEM ABLAUF FORT" ist, zu einem Schritt 112 vor. Bei dem Schritt 112 bestimmt die CPU **11**, ob der Wahlpunkt „ÄNDERE DEN ABLAUF" oder nicht ist. Wenn „ANDERE DEN ABLAUF" ausgewählt wird, so schreitet der Prozeß zu einem Schritt 113 fort. Bei dem Schritt 113 führt die CPU **11** ein Festsetzen durch, um einen unterschiedlichen Kurs zu einem Kurs, der als nächstes zu beschreiten ist, auszuwählen, wie es in Fig. 9 gezeigt wird. Z. B. wird das Festsetzen des Voranschreitens von dem Kurs 3 zu Kurs 4 verändert, zu einem Festsetzen des Vorwärtsgehens von Kurs 3 zu Kurs 7 o. dgl. Der Prozeß geht dann zu dem Schritt 104 zurück.

[0130] Währenddessen schreitet in dem Schritt 112, wenn der Wahlpunkt nicht „ÄNDERE DEN ABLAUF" ist, der Prozeß zu einem Schritt 114 voran. In dem Schritt 114 subtrahiert die CPU **11** 1 von der Spielerobjekt-Zahl, die in dem Registerbereich **159R** des RAM gespeichert ist. In dem Schritt 115 werden die Mitgliedsobjektsdaten aus dem Speicherbereich **158** der Mitgliedsdaten von dem RAM ausgelesen. In dem

Schritt 116 wird ein anfängliches Festsetzen für den Ablauf, der unmittelbar vorher gelöscht wird, durchgeführt. Wenn der Schritt 111 geendet hat, so geht der Prozeß zu der vorherigen Routine zurück, um zu dem Schritt 2 voranzuschreiten.

[0131] Mit Bezug auf Fig. 17 wird die Funktion einer Subroutine für den Modus-Verschiebungsprozeß (den Schritt 3 der Hauptroutine) erklärt werden. Wenn das Spielerobjekt den Modus-Verschiebungspunkt in Fig. 11 erreicht, so wird in einem Schritt 121 bestimmt (oder detektiert), daß er in der All-Bereichs-Modusposition vorhanden ist, und in einem Schritt 123 wird bestimmt, ob ein Demonstrationsprozeß (hierin bezeichnet als „Demo") in dem All-Bereichs-Modus enden wird oder nicht. Es wird für die erste Zeit bestimmt, daß der Demo-Prozeß nicht geendet hat, und ein Bildprozeß für die Demo-Anzeige in dem All-Bereichs-Modus wird in einem Schritt 124 durchgeführt. Ein Klangprozeß wird in einem Schritt 125 zum Erzeugen eines Demo-Klangs in dem All-Bereichs-Modus ausgeführt und dann schreitet der Prozeß auf den oben erwähnten Wiedergabeprozeß in dem Schritt 12 voran.

[0132] Auf der anderen Seite wird, wenn in dem Schritt 123 bestimmt wird, daß der Demo-Prozeß geendet hat, ein Verschiebungsprozeß zu dem All-Bereichs-Modus (Verändern einer Modusmarkierung F2 zu dem All-Bereichs-Modus) in einem Schritt 126 durchgeführt, und dann geht der Prozeß an die Hauptroutine zurück.

[0133] Dies liefert einen Vorteil, daß der Abrollbereich verschoben werden kann, ohne einen fremdartigen Eindruck zu geben, wie, daß sich die Bildschirm-Abrollvorrichtung abrupt beim Verschieben von dem Ein-Richtungs-Abrollmodus zu dem All-Bereichsmodus verändert. Ebenso löst das Verschieben des Abrollbereichs die Last von der CPU während der Ein-Richtungs-Abrollperiode, im Vergleich zu dem Fall, daß der Abrollbereich in den gesamten Bereich über die gesamte Ablaufdauer plaziert ist. Es ist ebenso möglich, eine Abrollanzeige in einer diversifizierten Art im Vergleich zu dem Ein-Richtungs-Abrollen über die gesamte Ablaufperiode zur Verfügung zu stellen. Somit sind eine Vielzahl von Bilddarstellungen für ein Spiel möglich, welches einen Vorteil beim Erhöhen weiterer Spielerinteressen liefert.

[0134] Mit Bezug auf Fig. 18 werden Erklärungen für die Funktion einer Subroutine für den Steuerungseinrichtungsprozeß (Schritt 4) durchgeführt werden. Es wird in dem Schritt 131 bestimmt, ob es ein Lese-Anfrage-Befehl für die Steuerungseinrichtungsdaten gegeben hat oder nicht. Wenn nicht, so wartet ein Lese-Befehl in dem Schritt 131 auf seine Erzeugung. Wenn die Anwesenheit eines Lese-Anfrage-Befehls bestimmt wird, so wird ein Befehl an die Steuerungseinrichtungs-Steuerungsschaltung **18** in dem Schritt 132 geliefert. In Antwort darauf führt die Steuerungseinrichtungs-Steuerungsschaltung **18** einen Prozeß des Lesens der Bedienungszustands-Daten auf der Steuerungseinrichtung **40A**–**40D** durch. In einem Schritt 133 wird bestimmt, ob das Lesen der Bedienungszustandsdaten durch die Steuerungseinrichtungs-Steuerungsschaltung **18** für alle Steuerungseinrichtungen **40A**–**40D** geendet hat oder nicht. Wenn sie nicht geendet hat, so wird auf das Ende gewartet. Wenn das Ende detektiert wird, so werden die Bedienungszustands-Daten für jede Steuerungseinrichtung **40A**–**40D** in einem Schritt 134 in den Speicherbereich **156** in dem RAM **14** über die Bus-Steuerungsschaltung **121** von der Steuerungseinrichtungs-Steuerungsschaltung **18** geschrieben.

[0135] Mit Bezugs auf die Fig. 19 bis Fig. 21 werden Erklärungen über die Funktion einer Subroutine für den Kommunikationsprozeß (Schritt 5) mit dem Mitglied gemacht werden. Es wird bestimmt in einem Schritt 141a, ob das Spielerobjekt den Standort A erreicht hat oder nicht. Wenn der Standort A als nicht erreicht bestimmt wird, so geht der Prozeß zurück zu der Hauptroutine, nach dem Verarbeiten in den Schritten 141b, 141c, 151a, 151b, 151c, 151d. Währenddessen, wenn das Spielerobjekt den Standort A erreicht hat, welcher im dem Schritt 141a bestimmt wird, dann wird in einem Schritt 142b bestimmt, ob ein Mitglied 1 vorne ist oder nicht. Wenn das erste Mitglied existiert, so wird dann in einem Schritt 143a bestimmt, ob eine Phrase oder Nachricht jetzt bei dem Verarbeiten dabei ist oder nicht. Wenn bestimmt wird, daß eine Phrase bei dem Verarbeiten dabei ist, so wird eine korrespondierende Markierung unter den Phrasenmarkierungen F31–F3n an die Phrase eingeschaltet und ein Vergleich in der Prioritätsordnung wird in dem Schritt 144a durchgeführt, aufgrund der Notwendigkeit des Auswählens irgendeiner aus einer Vielzahl von Phrasen. Es wird in dem Schritt 145a bestimmt, ob die Prioritätsordnung einer Phrase 1 höher ist als die Phrase, die momentan in einem Schritt 150a beim Verarbeiten ist. Wenn sie höher ist, so schreitet der Prozeß zu einem Schritt 146a voran. In dem Schritt 146a wird ein Anzeigeprozeß für die Phrase 1 durchgeführt. Z. B. ist die Phrase 2 eine Nachricht (Darüberhinwegkommen mit Bremsen) von dem ersten Kameraden an das Spielerobjekt, so daß sie von Angriffen durch einen Feind, der an dem Standort A auftritt, abwendet. Eine Nachricht wird angezeigt, um das Drücken des unteren Knopfes (Schalters **47cd**) von dem Schalter **47c** als eine Bedienungsart davon anzuweisen. Bei einem Schritt 147a wird ein Prozeß zum Ausgeben der Phrase 1 in einem Klang durchgeführt. Nebenbei angemerkt, gibt es, wenn es in dem Schritt 143a bestimmt wird, daß keine Phrase während der Verarbeitung ist, keine Notwendigkeit des Beurteilens der Präferenzordnung, so daß der Prozeß zu dem Schritt 146a voranschreitet. Wenn die Abwesenheit eines ersten Mitglieds in dem Schritt 142a bestimmt wird oder die

Phrase, die in Prozeß ist, eine niedrigere Präferenzordnung als die Phrase 1 hat, und diese in dem Schritt 145a bestimmt wird, so geht der Prozeß zurück zur Hauptroutine.

[0136] Auf der anderen Seite werden, wenn bestimmt wird, daß das Spielerobjekt an der Position B anstelle der Position A existiert, die Funktionen der Schritte 141b, 147b ausgeführt. Die Schritte 141b, 147b sind zum Ausgeben einer Phrase 2, welche ähnlich zu den Funktionen der Schritte 141a–147a sind. Dementsprechend wird die korrespondierende Schrittnummer gefolgt durch ein Symbol „b" anstelle des Symbols „a" dargestellt, welches Erklärungen im Detail erübrigt.

[0137] Währenddessen wird, wo der Phrasenausgabezustand abhängig von der Zeit ist, z. B. wo er von einer Zeitperiode A des Findens des Chefs abhängig ist, die Zeitperiode A in einem Schritt 141c bestimmt. Wenn ein zweiter Kamerad in der Nähe anwesend ist und dies in einem Schritt 142c bestimmt wird, werden die Funktionen der Schritte 143c–147c ausgeführt. Diese Schritte 143c–147c sind mit der Funktion zusammenhängend, daß der zweite Kamerad eine Nachricht sendet (Phrase 3 in Fig. 13), die mitteilt, wie der Chef zu stürzen ist (Sturmangriffsart), von welchem die Funktionen dieselben wie die Funktionen der Schritte 142a–147a sind, mit der Ausnahme des Unterschieds in dem Mitglied und der Phrase, und die detaillierten Erklärungen werden weggelassen.

[0138] Mit Bezug auf den Phrasenausgangszustand wird ein drittes Mitglied durch einen Feind im Auge behalten, wobei dasselbe in einem Schritt 151a bestimmt wird, um dadurch die Funktionen der Schritte 152a–156a zu bewirken. Die Schritte 152a–156a sind für die Funktion eine Nachricht auszugeben (Phrase 5 in Fig. 13), daß ein Mitglied unterrichtet, wie der Chef zu besiegen ist, und ihre Funktionen sind dieselben wie die Funktionen der Schritte 142a–146a mit der Ausnahme des Unterschieds in der Phrase.

[0139] Wenn der Phrasen-Ausgabezustand ein solcher ist, daß das dritte Mitglied gerettet worden ist, so wird in einem Schritt 151b dasselbe bestimmt, um dadurch die Funktion der Schritte 152b–156b auszuführen. Die Schritte 152b–156b sind die Funktion, um eine Phrase 6 zu erzeugen, wenn das dritte Mitglied gerettet wird, und die Funktionen sind ähnlich zu den Funktionen der Schritte 125a–156a, mit Ausnahme des Unterschieds in der Phrase.

[0140] Währenddessen wird, wenn der Phrasen-Ausgabezustand ein solcher ist, daß eine Phrase 8 durch das Spielerobjekt, welches von einem Feind überfallen worden ist, auszugeben ist, dasselbe in einem Schritt 151c bestimmt, um die Funktionen der Schritte 151c–155c zu bewirken. Wenn der Phrasen-Ausgabezustand das Ausgeben einer Phrase 9 betrifft, in welcher der Chef besiegt ist, so wird dasselbe in einem Schritt 151d bestimmt, um die Funktionen der Schritte 152d–156d durchzuführen.

[0141] Wie oben erwähnt, wird eine Nachricht (die Phrasen 1–4 in dem Beispiel der Fig. 13) als Anzeige-Klang ausgegeben, so daß der Spieler assistiert wird, um passende Handhabungen durchzuführen. Mit einem passenden Bedienungsverfahren von einem Ratschlag ist das Spiel leicht dadurch voranzutreiben, wodurch dem Spieler ein Leistungs- und Zufriedenstellungsgefühl gegeben wird, sogar wenn die Bedienungsart schwierig ist. Somit sind die Szenen und Abläufe leicht zu überschauen. Wenn eine genaue Nachricht (die Phrasen 5–9 in dem Beispiel der Fig. 13) in der Anzeige und/oder dem Klang in Konformität zu einer Szene oder Situation des Spiels ausgegeben wird, so sind sehr realistische Ausdrücke in dem Fortlauf des Spiels erhältlich, welches weiterhin das Amüsement in dem Spiel erhöht.

[0142] Nebenbei bemerkt, ist die Anzeige- oder die Klangnachricht, welche zum Helfen des Spielers erzeugt wird, um sauber in Konformität zu einer Nachricht oder Situation zu handhaben, nicht auf die Fig. 13 und der darin enthaltenen Ausführungsform eingeschränkt, sie jedoch kann abhängig von der Art oder dem Inhalt eines Spieles passend modifiziert werden. Somit ist sie nicht auf die Beschreibung der Ausführungsform eingeschränkt. Z. B., obwohl das Bedienungsverfahren der Schalter in dem Fall erklärt worden ist, daß irgendeiner einer Vielzahl von Schaltern gedrückt wird, um die Erklärung zu vereinfachen, ist es ebenso möglich, zu bestimmen, das ein gleicher Schalter eine Vielzahl von Malen gedrückt wird oder eine vorbestimmte Kombination von einer Vielzahl von Schaltern gedrückt wird.

[0143] Mit Bezug auf die Fig. 22–Fig. 23, werden Erklärungen über die Funktion einer Subroutine für den Materialnachfüllprozeß (Schritt 8) gemacht werden. Bevor das Spielerobjekt zu einem vorbestimmten Standort oder einer Position kommt, bei der ein Punkt erhältlich ist, wird in einem Schritt 161 bestimmt, daß das Spielerobjekt nicht den Standort betreten hat. Bei einem Schritt 163 wird bestimmt, daß eine Zeitperiode (T1) zum Anzeigen eines Punkts nicht gesetzt ist (T1 = 0). Es wird in einem Schritt 170 bestimmt, daß es nicht in einem Zustand des Darstellens einer Marke ist (Punktbox), die darstellt, daß man ein Recht hat, einen Punkt zu erwerben. In einem Schritt 172 wird bestimmt, daß kein Punkt gesetzt wird, und dann geht der Prozeß zurück zu der Hauptroutine. Danach wird der Prozeß der Hauptroutine in einer Rahmenperiode ausgeführt.

[0144] Wenn das Spielerobjekt zu einem Platz kommt, worin es einen Punkt bekommen kann, so wird dasselbe in einem Schritt 161 bestimmt. In ei-

nem Schritt 162 wird eine konstante Zeitperiode (T1) auf ein Zeitregister festgesetzt, als eine Zeitperiode, für welche die Anzeige für eine Markierung des Mitteilens, daß sie in einem Zustand ist, der einen Punkt erhältlich macht, gemacht wird. Es wird in einem Schritt 163 bestimmt, daß die Zeitperiode T1 > 0 ist. In einem Schritt 164 wird die Subtraktion (T1 – 1) durch eine Einheitszeit (z. B. 1 Sekunde) durchgeführt. In einem Schritt 165 wird die Anzeige für eine Markierung durchgeführt, durch welche der Spieler einen Punkt beantragen kann, wenn er einen Punktanzeige-Beantragungsknopf (z. B. 47Cr) drückt. Es wird in einem Schritt 166 bestimmt, ob der Punktantragsschalter gedrückt ist oder nicht. Wenn kein Drücken bestimmt wird, dann werden die Schritte 170–172 ausgeführt, und dann geht der Prozeß zurück zur Hauptroutine. Die Schritte 161, 163–166, 170, 172 werden für jede Rahmenperiode wiederholt, wodurch der Schalter **47Cr** darauf wartet, innerhalb einer bestimmten Zeitperiode gedrückt zu werden.

[0145] Während der Wiederholung der Wartefunktion, wie oben beschrieben, wird in dem Schritt 166 bestimmt, daß der Anzeige-Antragsschalter gedrückt ist, so wird 0 (zurücksetzen) auf das Zeitregister gesetzt, und eine Herstellung in einem Schritt 168 zum Ausgeben einer Phrase wird durchgeführt, die einen Punkt, der von einem Mitglied beantragt wird, darstellt. Diese Phrase wird in den Schritten 14 und 15 in Bild und Klang ausgegeben. In einem Schritt 169 wird ein Prozeß (Punkt-Ersetzprozeß) zum Anzeigen einer Markierung (Punktbox) durchgeführt, die in einem Zustand zeigt, daß es möglich ist, einen Punkt zu erhalten. In einem Schritt 170 wird in dem Zustand bestimmt, daß eine Punktbox angezeigt werden kann. In einem Schritt 171 wird ein Prozeß zum Anzeigen einer Punktbox durchgeführt. Wenn in einem Schritt 172 bestimmt wird, daß die Anzeige von der Punktbox durchgeführt worden ist, wird in einem nächsten Schritt 173 bestimmt, ob der Spieler eine Funktion zum Erhalten der Punktbox durchführte oder nicht (z. B. Funktion des Feuerns bei der Punktbox oder Funktion des Überlappens des Spielerobjekts über die Punktbox oder dgl.). Wenn bestimmt wird, daß die Punktbox erhalten worden ist, so wird ein Prozeß in den Schritten 173–180 zum Liefern eines Punkts, der abhängig von einem Zustand des Spielerobjekts erforderlich ist, durchgeführt. Z. B. wird, wenn das Spielerobjekt ein Kämpfer oder ein Schußspiel ist, in einem Spiel 174 bestimmt, ob die Flügel in einem vorbestimmten Zustand sind. Wenn ein vorbestimmter Flügel nicht existiert, so wird ein Flügel als ein Nachfüllpunkt in einem Schritt 175 angeboten. Wenn ein vorbestimmter Flügel existiert, so wird in einem Schritt 176 bestimmt, ob das Leben oder ein schadenswiderstehender Betrag nicht größer als ein konstanter Wert (128) ist oder nicht. Wenn das nicht größer bestimmt wird, so wird ein Punkt zum Wiederherstellen des Lebens in einem Schritt 177 geliefert. Wenn das Leben größer als der konstante Wert (128) ist, so wird in einem Schritt 178 bestimmt, ob es zweistrahlige Artillerien (twin beam) gibt oder nicht. Wenn eine Existenz bestimmt wird, so wird ein twin beam in einem Schritt 179 geliefert. Wenn die Anwesenheit bestimmt wird, so wird eine Bombe in einem Schritt 180 angeboten.

[0146] Auf diese Weise wird der Punkt effektiv zum Voranschreiten des Spiels durch den Spieler geliefert, abhängig von dem Zustand des Spielerobjekts, so daß es dem Spieler erleichtert wird, das Spiel auf einer Erfolgsszene fortzufahren oder über den Ablauf hinweg zu löschen. Somit erhält der Spieler leicht ein Leistungs- und Befriedigungsgefühl in dem Spiel. Ebenso kann der Spieler mit einem Gefühl spielen, wenn er Anweisungen empfängt, während er eigentlich einen Kämpfer steuert oder einen Flug durchführt, während er die Assistenz empfängt, mit steigendem Spielinteresse. Nebenbei gesagt, ist der Lieferungspunkt verschieden, abhängig von der Art oder dem Inhalt des Spiels, so daß der Software-Entwickler des Spiels verschiedene Modifikationen mit Bezug auf die technische Idee, die in dieser Ausführungsform beschrieben wird, durchführen könnte.

[0147] Mit Bezug auf die Flußdiagramme in Fig. 32, Fig. 33, Fig. 36, Fig. 38, Fig. 39 und Fig. 42 wird eine Subroutine des Verarbeitens des Spielerobjekts erklärt werden. Als erstes liest in einem Schritt 101 die CPU **11** die Joystick-Daten, die in den Datenbereich der Steuerungs-Anschlußfläche gespeichert sind, aus, und korrigiert die Daten. Insbesondere werden die Daten für einen zentralen Teil des Joysticks gelöscht, so daß die Daten bei „0” sind, wenn der Stick in eine Position nahe oder um das Zentrum herum kommt (z. B. 10 Zahlen im Radius). Dies ermöglicht den Joystick- Daten genau „0” anzunehmen, sogar wenn es einen Fehler aufgrund der Herstellung des Joysticks gibt oder wenn ein Finger des Spielers empfindlich zittert. Ebenso werden Korrekturen für einen vorbestimmten Bereich bei einem äußeren peripheren Anteil in einem Joystick-Bedienungsbereich durchgeführt. Insbesondere werden Korrekturen nicht durchgeführt, um einen ungewollten Anteil der Daten beim Spielen eines Spiels auszugeben.

[0148] In einem Schritt 302 bestimmt die CPU **11** die Joystick-Daten X_j, Y_j, die für ein Spiel zu verwenden sind. Da die Daten, die in dem Schritt 301 bestimmt werden, einen Wert des Zählers **444** haben, werden sie in eine Form umgewandelt, die leicht in dem Spiel zu verarbeiten ist. Insbesondere wird X_j „0”, wenn der Stick nicht betätigt wird, „+60”, wenn er maximal in eine –X-Achsen-Richtung (linksseitig) ausgelenkt wird, und „–60”, wenn er maximal in eine +X-Achsen-Richtung (rechtsseitig) ausgelenkt wird. Y_j wird „0”, wenn der Stick nicht ausgelenkt wird, „+60”, wenn er maximal in eine +Y-Achsen-Richtung (vorwärts) ausgelenkt wird, und „–60”, wenn er maximal in eine –Y-Achsen-Richtung (rückwärts) ausgelenkt wird.

[0149] In einem Schritt 303 setzt die CPU **11** eine Basisgeschwindigkeit As0 des Spielerobjekts in einem dreidimensionalen Spielraum fest. As0 ist eine Spielraum-Entfernung, die das Spielerobjekt in einem 1-Rahmen voranschreitet. Dieses As0 kann entsprechend einem Ablauf oder einer Szene, die gespielt wird, frei festgesetzt werden. In einem Schritt 304 liest die CPU **11** die Daten des Knopfschalters C1, welche in dem Steuerungsanschluß-Flächenbereich gespeichert sind, aus, um zu bestimmen, ob der Spieler den Knopfschalter C1 gedrückt hat oder nicht. Dieser Knopfschalter C1 wird als ein Verstärkungsschalter benutzt, um eine Voranschreitungsgeschwindigkeit des Spielerobjekts in dem Spiel zu erhöhen. Wenn der Knopfschalter C1 gedrückt ist, so wird das Spielerobjekt, wenn es z. B. ein Flugzeug ist, bildverarbeitet, um den Düsenausstoß oder das Düsenaufheulen zu steigern. Wenn bestimmt wird, daß der Spieler den Knopfschalter C1 gedrückt hat, so schreitet der Prozeß zu einem Schritt 305 voran. Bei dem Schritt 305 verarbeitet die CPU **11** das Liefern eines Vibrations-Erzeugungssignals an die Erzeugungsschaltung **53** der Vibration von der Steuerungseinrichtung. Als ein Ergebnis vibriert die Steuerungseinrichtung, wenn der Knopfschalter C1 gedrückt worden ist. Wenn der Knopfschalter C1 nicht gedrückt wird, so wird die Vibration gestoppt. Die Vibration der Steuerungseinrichtung während der Verstärkung erhöht auf diese Weise das Realismusgefühl des Spiels, wodurch das Interesse des Spielers wesentlich gesteigert wird.

[0150] In einem Schritt 306 führt die CPU **11** eine Berechnung As = As0 + Asα durch. As ist eine Geschwindigkeit des Spielerobjekts, während Asα eine erhöhte Geschwindigkeit des Spielerobjekts während der Verstärkung ist. Diese Berechnung erhöht eine Bewegungsentfernung in einer dreidimensionalen Position des Spielerobjekts in einen nächsten Rahmen. Dann schreitet der Prozeß zu einem Schritt 309 voran.

[0151] Auf der anderen Seite, wenn in einem Schritt 304 bestimmt wird, daß der Spieler nicht den Knopfschalter C1 gedrückt hat, so schreitet der Prozeß zu einem Schritt 307 voran. In dem Schritt 307 liest die CPU **11** die Daten über den Knopfschalter **47Cd**, die in dem Datenbereich der Steuerungsanschlußfläche gespeichert sind, aus, um zu bestimmen, oh der Spieler den Knopfschalter **47Cd** gedrückt hat oder nicht. Dieser Knopfschalter **47Cd** wird als ein Bremsschalter benutzt, um die voranschreitende Geschwindigkeit des Spielerobjekts in dem Spiel abzubremsen. Wenn der Knopfschalter **47Cd** gedrückt ist, so wird das Spielerobjekt, wenn es z. B. ein Flugzeug ist, bildverarbeitet, wie für das Bremsfeuern, welches einen Bremsfeuerklang erzeugt. Wenn bestimmt wird, daß der Spieler den Knopfschalter **47Cd** gedrückt hat, so schreitet der Prozeß zu einem Schritt 308 voran. In dem Schritt 308 führt die CPU **11** eine Vibrationsverarbeitung ähnlich zum Schritt 305 durch.

[0152] In dem Schritt 309 führt die CPU **11** eine Berechnung As = As0 – Asβ durch. Asβ ist eine erniedrigte Geschwindigkeit des Spielerobjekts während des Anwendens einer Bremse. Die Berechnung erniedrigt die Bewegungsgeschwindigkeit in einer dreidimensionalen Position des Spielerobjekts in den nächsten Rahmen. Der Prozeß schreitet dann zu einem Schritt 310 voran.

[0153] Währenddessen wird in dem Schritt 307, wenn bestimmt wird, daß der Spieler nicht den Knopfschalter **47Cd** gedrückt hat, der Prozeß zu dem Schritt 310 voranschreiten. In dem Schritt 310 bestimmt die CPU **11**, ob man in dem All-Bereichs-Modus ist oder nicht. Insbesondere detektiert die CPU **11**, ob eine Markierung des All-Bereichs-Modus in dem Markierungsbereich **159f** des RAM **15** auf (gesetzt) oder (zurückgesetzt) ist. Wenn die Markierung des All-Bereichs-Modus gesetzt ist, so wird der momentane Spielmodus in dem All-Bereichs-Modus bestimmt. Wenn die Markierung des All-Bereichs-Modus in einem zurückgesetztem Zustand ist, so wird der momentane Spielmodus als der Ein-Richtungs-Abrollmodus bestimmt. Wenn die Bestimmung nicht der All-Bereichs-Modus ist, so schreitet der Prozeß zu einem Schritt 320 voran.

[0154] In dem Schritt 320 funktioniert die CPU **11** nach den folgenden Gleichungen. Anzumerken ist, daß die Multiplikation in der vorliegenden Beschreibung durch „Sternchenzeichen" angezeigt wird.

$$As \cdot \sin\theta x1 = 0{,}68 \cdot X_j$$

$$\theta y1 = 0{,}68 \cdot Y_j$$

$$Xs = -As \cdot \sin\theta x1$$

$$Zs = -As \cdot \cos\theta x1$$

$$Ys = -As \cdot \sin\theta y1$$

[0155] Hierin wird die Bedeutung von θx1, θy1, Xs, Zs, Ys im Detail mit Bezug auf Fig. 34 erklärt werden. θx1 ist ein Winkel eines As-Richtungsvektors des Spielerobjekts **60** im Hinblick auf eine –Z-Achsen-Richtung auf einer XZ-Koordinate (ein ebenes Koordinatensystem des Spielerobjekts **60** wie es aus dem Obigen ersichtlich ist), und hat einen Wert, der in eine +-Richtung erhöht wird, wenn der Stick des Joysticks linksseitig ausgelenkt wird und in eine –-Richtung erhöht wird, wenn er rechtsseitig ausgelenkt wird. Aus den Gleichungen wird ersichtlich, daß der Wert θx1 sich erhöht und abnimmt von –40,8 (0,680·60) auf 40, 8 (0,68·(–60)) in Anteilen von einer Auslenkung des Sticks. Da das Spielerobjekt **60** in Richtung auf eine As-Richtung voranschreitet, wird es in der Richtung

in einem Bereich von rechts und links in einem Bereich von –40,8 (0,68·60) auf 40,8 (0,68·(–60)) Grad verändert. Z. B. ist diese Richtung eine Rumpfnasen-Richtung, wenn ein Flugzeug wie in Fig. 35 als das Spielerobjekt **60** benutzt wird. Sie kann eine Richtung einer Kanone sein, wenn ein Panzer benutzt wird, während sie eine Richtung eines Gesichts sein kann, wenn ein Tier benutzt wird.

[0156] θy1 ist ein Winkel des As-Richtungsvektors des Spielerobjekts **60** im Hinblick auf eine –Z-Achsen-Richtung auf einer YZ-Koordinate (ein ebenes Koordinatensystem, wenn das Spielerobjekt **60** in der Ebene von dem linken in Richtung der rechten Seite gesehen wird), welche einen Wert hat, der sich in die +-Richtung erhöht, wenn der Stick des Joysticks vorwärts ausgelenkt wird und sich in die —Richtung erniedrigt, wenn der Joystick rückwärts ausgelenkt wird. Aus den Gleichungen wird ersichtlich, daß sich der Wert θx1 von –40,8 (0,68·60) auf 40,8 (0,68·(–60)) in Anteilen einer Auslenkung des Sticks erhöht und erniedrigt. Da das Spielerobjekt **60** in Richtung der As-Richtung voranschreitet, wird es auf eine vertikale Richtung in einem Bereich von –40,8 (0,68·60) auf 40,8 (0,68·(–60)) Grad verändert.

[0157] Xs, Zs und Ys sind jeweils eine X-Achsen-Komponente, eine Y-Achsen-Komponente und eine Z-Achsen-Komponente des As.

[0158] In einem Schritt 321 bestimmt die CPU **11**, ob ein linker Anteil des Spielerobjekts **60** vorhanden ist oder nicht. Da in dieser Ausführungsform ein Flugzeug als ein Beispiel des Spielerobjekts zur Erklärung benutzt wird, wird die Anwesenheit oder Abwesenheit eines linken Flügels bestimmt.

[0159] Anzumerken ist, daß, wenn ein Panzer als Spielerobjekt benutzt wird, es eine Kette anstelle eines Flügels ist. Wenn ein Tier verwendet wird, wird es ein Körperglied sein. Wenn bestimmt wird, daß der linke Anteil des Spielerobjekts **60** anwesend ist, so schreitet der Prozeß zu einem Schritt 322 voran. In dem Schritt 322 bestimmt die CPU **11**, ob ein rechter Anteil des Spielerobjekts **60** existiert oder nicht. Wenn bestimmt wird, daß der rechte Anteil des Spielerobjekts **60** anwesend ist, so schreitet der Prozeß zu einem Schritt 323 voran.

[0160] Im Schritt 323 funktioniert die CPU **11** mit den folgenden Gleichungen:

Xa = Xa + Xs

Ya = Ya + Ys

Za = Za + Zs

[0161] Die Berechnung addiert eine Geschwindigkeit (Xs, Ys, Zs) des Spielerobjekts **60** zu den Koordinaten (Xa, Ya, Za) (auf der rechten Seite der obigen Gleichung) in dem vorhergehenden Rahmen. Somit wird eine Koordinate (Xa, Ya, Za)(linke Seite der obigen Gleichung) für einen Folgerahmen bestimmt. Der Prozeß schreitet dann zu einem Schritt 330 in Fig. 36 voran.

[0162] Auf der anderen Seite, wenn in dem Schritt 323 bestimmt wird, daß es keinen linken Anteil des Spielerobjekts **60** gibt, schreitet der Prozeß zu einem Schritt 324 voran. Die CPU **11** funktioniert in dem Schritt 324 mit den folgenden Gleichungen:

Xs = Xs – 2,5

Ys = Ys – 2,5

[0163] Die Berechnung reduziert die Geschwindigkeit in die X-Achsen-Richtung und die Geschwindigkeit in die Y-Achsen-Richtung. Deshalb wird das Spielerobjekt **60**, wenn das Spiel voranschreitet, angezeigt, wenn er in eine linke Abwärtsbewegung gezogen wird. Z. B. wird, sogar wenn der Joystick nicht ausgelenkt wird, das Spielerobjekt **60** anzeigt, wenn er sich allmählich in Richtung der linken Abwärtsrichtung, wie beim Betrachten des Bildschirms zu sehen ist, bewegt.

[0164] Währenddessen schreitet der Prozeß zu einem Schritt 325 voran, wenn in dem Schritt 322 bestimmt wird, daß das Spielerobjekt **60** keinen rechten Anteil hat. In dem Schritt 325 funktioniert die CPU **11** nach den folgenden Gleichungen:

Xs = Xs + 2,5

Ys = Ys – 2,5

[0165] Die Berechnung erhöht die Geschwindigkeit in der X-Achsen-Richtung, jedoch erniedrigt sich die Geschwindigkeit in der Y-Achsen-Richtung. Dementsprechend wird das Spielerobjekt **60**, wenn es in dem Spiel voranschreitet, angezeigt, wenn es in eine rechte Abwärtsbewegung gezogen wurde, wie es gesehen wird, wenn man den Bildschirm betrachtet. Z. B. wird, sogar wenn der Stick nicht ausgelenkt wird, das Spielerobjekt **60** in einer Weise angezeigt, daß es sich allmählich in Richtung der rechten Abwärtsbewegung bewegt. Dann schreitet der Prozeß zu dem Schritt 323 voran.

[0166] Auf der anderen Seite schreitet, wenn in dem Schritt 310 bestimmt wird, daß die CPU **11** nicht in dem All-Bereichs-Modus ist, der Prozeß zu einem Schritt 330 voran.

[0167] Als erster wird zu dem Unterschied zwischen dem Ein-Richtungs-Abrollmodus und dem All-Bereichs-Modus Stellung genommen. Der Ein-Richtungs-Abrollmodus hat ein Koordinatensystem, wie

es in Fig. 34 gezeigt wird, während der All-Bereichs-Modus zwei Koordinatensystem hat, wie es in der Fig. 37 gezeigt wird. Bei der Detailerklärung der jeweiligen Koordinatensysteme für den All-Bereichs-Modus ist eine X1(–Z1)-Koordinate von Fig. 37, eine Koordinate zum Darstellen eines dreidimensionalen Raums, worin eine Vielzahl von Objekten existiert, welche immer auf eine festgesetzte Weise während des Spielens eines Spiels geliefert werden. (In dieser Ausführungsform bezieht sich die Koordinate auf eine „Spielraum-Koordinate".) Eine X2(–Z2)-Koordinate ist eine Koordinate, die basierend auf der Bewegung des Spielerobjekts verändert wird. Da die –Z2-Achsen-Richtung mit einer Kamerafotografierungsrichtung, wie später bei dem Kameraverfahren erklärt wird, korrespondiert, stellt die Z2-Achse eine Richtung senkrecht zu dem Anzeigebildschirm dar. (In der vorliegenden Ausführungsform wird sie als eine „Spieler-Koordinate" bezeichnet.) Das Spielerobjekt **60** führt in dieser Spieler-Koordinate eine Funktion durch, die im allgemeinen ähnlich zu der des Ein-Richtungs-Abrollmodus ist. Der Gebrauch der zwei Koordinatensysteme auf diese Art macht es möglich, ein All-Bereichs-Modussystem aufzubauen, welches in jede Richtung abrollen kann, während ein Modussystem des Ein-Richtungs-Abrollens benutzt wird. Deshalb ist, sogar wenn der Modus verändert wird, die Bedienungsweise annähernd unverändert. Deshalb wird es dem Spieler leicht gemacht, sich an dem Spiel zu erfreuen.

[0168] Bei einem Schritt 330 funktioniert die CPU **11** mit den folgenden Gleichungen:

$\theta x1 = 0{,}68 \cdot X_j$

$\theta y1 = 0{,}68 \cdot Y_j$

θx2 = 0,03·θx1 + θx2(0,03·θx1 erhöht pro 1-Rahmen)

Xs = –As·sin(θx1 + θx2)

Zs = –As·cos(θx1 + θx2)

Ys = –As·sinθy1

[0169] Mit Bezug auf Fig. 37 werden Erklärungen für θx1, θy1, θx2, Xs, Zs und Ys gemacht werden. θx1 ist ein Winkel des As-Richtungsvektors des Spielerobjekts **60** im Hinblick auf die –Z2-Achsen-Richtung in einer XZ-Koordinate (ein Spieler- Koordinatensystem, wenn das Spielerobjekt **60** in der Ebene von dem Obigen gesehen wird), und hat einen Wert, der in die +- Richtung erhöht wird, wenn der Stick des Joysticks linksseitig ausgelenkt wird und in die –-Richtung erniedrigt wird, wenn er rechtsseitig ausgelenkt wird.

[0170] θy1 ist ein Winkel des As-Richtungsvektors des Spielerobjekts im Hinblick auf die –Z2-Achsen-Richtung in einer YZ-Koordinate (ein Spieler-Koordinatensystem, wenn das Spielerobjekt **60** in einer Ebene von der linken in Richtung auf die rechte Seite gesehen wird), und hat einen Wert, der sich in die +-Richtung erhöht, wenn der Stick des Joysticks vorwärts gedrückt wird und sich in die –-Richtung erniedrigt, wenn er rückwärts gedrückt wird.

[0171] θx2 ist ein Winkel zwischen der –Z1-Achsen-Richtung und der –Z2-Achsen-Richtung des Spielerobjekts **60** auf einer XZ-Koordinate (ein Spieler-Koordinatensystem, wenn das Spielerobjekt **60** in einer Ebene von dem Obigen gesehen wird), und hat einen Wert, der sich erhöht und erniedrigt in Anteilen zu dem θx1 auf jedem Rahmen. D. h., er erhöht sich in die +-Richtung, wenn der Stick des Joysticks linksseitig ausgelenkt wird und erhöht sich in die –-Richtung, wenn er rechtsseitig ausgelenkt wird. Aus den Gleichungen ist ersichtlich, daß der Wert θx2 sich in Anteilen zu der Auslenkung des Sticks verändert und in einem Bereich von –224 (= 0,03·0,68·(–60)) auf 1, 224 (= 0,03·0,68·60) in jedem 1- Rahmen erhöht und erniedrigt.

[0172] Xs, Zs und Ys sind jeweils eine X-Achsen-Komponente, eine Y-Achsen-Komponente und eine Z-Achsen-Komponente des As in der Spielraum-Koordinate.

[0173] In einem Schritt 332 bestimmt die CPU **11** ob ein linker Anteil des Spielerobjekts **60** existiert oder nicht. Wenn der linke Anteil des Spielerobjekts **60** als anwesendseiend bestimmt wird, so schreitet der Prozeß zu einem Schritt 333 voran.

[0174] In dem Schritt 333 bestimmt die CPU **11**, ob ein rechter Anteil des Spielerobjekts **60** vorhanden ist oder nicht. Wenn der rechte Anteil des Spielerobjekts **60** als vorhandenseiend bestimmt wird, so schreitet der Prozeß zu einem Schritt 334 voran. In dem Schritt 334 funktioniert die CPU **11** nach den folgenden Gleichungen, ähnlich zu dem Schritt 323:

Xa = Xa + Xs

Ya = Ya + Ys

Za = Za + Zs

[0175] Die Berechnung addiert eine Geschwindigkeit (Xs, Ys, Zs) des Spielerobjekts **60** in dem Koordinatenraum des Spielraums zu den Spielraum-Koordinaten (Xa, Ya, Za) auf den vorherigen Rahmen, wodurch eine Spielraum-Koordinate (Xa, Ya, Za) auf dem nachfolgenden Rahmen bestimmt wird. Der Prozeß schreitet dann zu einem Schritt 350 in Fig. 38 voran.

[0176] Auf der anderen Seite, wenn in dem Schritt 332 bestimmt wird, daß der linke Anteil des Spieler-

objekts **60** nicht vorhanden ist, so schreitet der Prozeß zu dem Schritt 335 voran. Die CPU **11** funktioniert in dem Schritt 335 nach den folgenden Gleichungen:

$$Xs = Xs - 2{,}5 \cdot \cos\theta x2$$

$$Zs = Zs - 2{,}5 \cdot \sin\theta x2$$

$$Ys = Ys - 2{,}5$$

[0177] Diese Berechnung erhöht die Geschwindigkeit in die X-Achsen-Richtung und die Geschwindigkeit in die Y-Achsen-Richtung in dem Spielraum. Dementsprechend wird das Spielerobjekt **60**, wenn es in dem Spiel voranschreitet, angezeigt, wenn es in die linke Abwärtsrichtung gezogen wird, wie es bei Betrachten des Bildschirms gesehen wird. Z. B. wird das Spielerobjekt **60**, sogar wenn der Stick nicht ausgelenkt wird, angezeigt, sich allmählich in Richtung der linken Abwärtsrichtung, wie es bei Betrachten des Bildschirms gesehen wird, zu bewegen.

[0178] Dann schreitet, wenn in dem Schritt 333 bestimmt wird, daß der rechte Anteil des Spielerobjekts **60** vorhanden ist, der Prozeß zu einem Schritt 336 voran. In dem Schritt 336 funktioniert die CPU **11** nach den folgenden Gleichungen:

$$Xs = Xs + 2{,}5 \cdot \cos\theta x2$$

$$Zs = Zs - 2{,}5 \cdot \sin\theta x2$$

$$Ys = Ys - 2{,}5$$

[0179] Die Berechnung erhöht die Geschwindigkeit in die X-Achsen-Richtung und erniedrigt die Geschwindigkeit in die Y-Achsen-Richtung in dem Spielerraum. Dementsprechend wird das Spielerobjekt **60**, wenn es in dem Spiel voranschreitet, angezeigt, wenn es in die rechte Abwärtsrichtung, wie es bei Betrachten des Bildschirms gesehen wird, gezogen wird. Z. B. wird das Spielerobjekt **60**, sogar wenn der Stick nicht ausgelenkt wird, als allmählich sich bewegend in Richtung der rechten Abwärtsbewegung angezeigt. Der Prozeß schreitet dann zu dem Schritt 334 voran.

[0180] Nebenbei bemerkt wird, obwohl in der Figur zur Vereinfachung beschrieben wurde, daß der Winkel in der Steuerungseinrichtung direkt auf dem Spielerobjekt für jeden Raum reflektiert wird, insbesondere ein Filterprozeß während des Ableitens von $\theta x1$ aus $0{,}68 \cdot Xj$ durchgeführt. Ähnlich wird der Filterungsprozeß während des Reduzierens von $\theta y1$ aus $0{,}68 \cdot Yj$ durchgeführt. Bei Erklären des Filterungsprozesses im Detail wird ersichtlich, daß die CPU **11** den $\theta x1$-Wert dazu veranlaßt, sich allmählich im $0{,}68 \cdot Xj$ über eine Vielzahl von Rahmen anzunähern, um so $\theta x1 = 0{,}68 \cdot Xj$ zu einer Zeit zu erhalten, bei der die Rahmen verarbeitet worden sind, ohne den Wert $0{,}6 \cdot Xj$ unmittelbar $\theta x1$ in einem 1-Rahmen werden zu lassen. Z. B. wird $0{,}68 \cdot Xj$ durch Nf dividiert (vorbestimmte natürliche Zahl), und $0{,}68 \cdot Xj/Nf$ wird zu $\theta x1$ während einer Nf-Rahmenperiode addiert, um dadurch $\theta x1 = 0{,}68 \cdot Xj$ zu erhalten. Der Filterungsprozeß zum Reduzieren vom $\theta y1$ aus $0{,}68 \cdot Yj$ ist ähnlich zu dem Filterungsprozeß zum Herleiten von $\theta x1$ aus $0{,}68 \cdot Xj$. Durch diesen Filterungsprozeß ist es möglich, das Spielerobjekt **60** davon abzuhalten, schreckliche Bewegungen aufgrund abrupter Bedienung des Joysticks oder grausamer Veränderungen der Szenen, die das Auge des Spielers belasten oder die Steuerung schwierig machen, aufkommen zu lassen.

[0181] In einem Schritt 350 führt die CPU **11** eine Trefferbestimmung für das Spielerobjekt **60** durch. Das Detail dieser Trefferbestimmung wird durch Verwenden der Schritte 351–354 erklärt werden. Die Schritte 351–354 sind normale Trefferbestimmungen. Die Trefferbestimmungen an dem Mitgliedsobjekt, welches später erklärt werden wird, sind ähnlich derjenigen.

[0182] In einem Schritt 351 in Fig. 39 bestimmt die CPU **11** ob $ABS(OBJ2x - OBJ1x) \leq OBJ1r$ ist oder nicht. OBJ1 ist ein Objekt, welches zu bestimmen ist, ob es Treffer hat und bezieht sich in dieser Ausführungsform auf das Spielerobjekt **60**. OBJ2 ist ein Objekt, welches in Richtung auf OBJ1 zukommt, und bezieht sich in dieser Ausführungsform auf ein angreifendes Objekt, wie ein Mitgliedsobjekt, ein Feindobjekt, ein stationäres Objekt, ein Laserobjekt, welches durch ein Feindobjekt bedient wird. OBJ1x ist ein X-Koordinaten-Wert des OBJ1, während OBJ2x ein X-Koordinaten-Wert des OBJ2 ist. OBJ1x und OBJ2x können auf einer Spielraumkoordinate oder einer Spielerkoordinate sein, vorausgesetzt, daß sie X-Koordinaten-Werte in einem gemeinsamen Koordinaten-System sind. Abs () bezeichnet einen absoluten Wert einer Zahl, die innerhalb () gegeben ist. OBJ1r ist ein Wert, der eine halbe Länge von einer Seite eines Kubus darstellt, worin OBJ1 als ein Kubus gesehen wird. In anderen Worten ist OBJ1r ein Wert, der einen Trefferbestimmungsbereich von OBJ1 darstellt. Wenn $ABS(OBJ2x - OBJ1x) \leq OBJ1r$ stillsteht, so schreitet der Prozeß zu einem Schritt 302 voran.

[0183] In dem Schritt 352 bestimmt die CPU **11**, ob $ABS(OBJ2y - OBJ1y) \leq OBJ1r$ ist oder nicht. OBJ1y ist ein Y-Koordinaten-Wert von OBJ1, während OBJ2y ein Y-Koordinaten-Wert von OBJ2 ist. OBJ1y und OBJ2y können eine Spielraumkoordinate oder einer Spielerkoordinate sein, vorausgesetzt, daß sie Y-Koordinaten-Werte auf einem gemeinsamen Koordinaten-System haben. Wenn $ABS(OBJ2y - OBJ1y) \leq OBJ1r$ stillsteht, so schreitet der Prozeß zu einem Schritt 353 voran. Bei dem Schritt 353 bestimmt die CPU **11**, ob oder ob nicht $ABS(OBJ2z - OBJ1z) \leq OBJ1r$ stillsteht. OBJ1z ist ein Z-Koordinaten-Wert des OBJ1, und OBJ2z ist ein Z-Koordina-

ten-Wert des OBJ2. OBJ1z und OBJ2z können auf einer gemeinsamen Spielraumkoordinate oder auf einer Spielerkoordinate sein, vorausgesetzt, daß sie Z-Achsen-Koordinaten-Werte auf einem gemeinsamen Koordinaten-System haben. Wenn ABS(OBJ2z – OBJ1z) ≤ OBJ1r stillsteht, so schreitet der Prozeß zu einem Schritt 354 voran.

[0184] Bei dem Schritt 354 bestimmt die CPU **11** die Treffer von OBJ2 und OBJ1 und setzt eine Markierung der Trefferbestimmung in den Markierungsbereich **159F** des RAM **15** fest. Das Spielerobjekt **60** in dieser Ausführungsform besteht aus drei Objekten, einem Hauptkörperobjekt **61** (hierin als Hauptkörper **61** bezeichnet), einem linken Flügelobjekt **60L** und einem rechten Flügelobjekt **60R**. Das jeweilige Objekt kann auf der Anzeige **30** durch Eingabe in eine Anzeigeliste angezeigt werden. Ebenso hat das Spielerobjekt **60**, wie es in Fig. 35 gezeigt wird, insgesamt vier Bestimmungspunkte, die an einem oberen Anteil des Hauptkörpers **61**, an einem unteren Anteil des Hauptkörpers **61**, an einem Anteil eines linken Flügels **62** nahe dem Hauptkörper und an einem Anteil des rechten Flügels **63** nahe dem Hauptkörper angeordnet sind, so daß sie jeweils 1 Punkt haben. Die jeweiligen Punkte werden der Trefferbestimmung in den Schritten 351–354 unterworfen, um zu bestimmen, ob eine Markierung der Trefferbestimmung für jeden Punkt gesetzt werden sollte oder nicht. Der Prozeß geht dann zurück zu der vorherigen Routine.

[0185] Währenddessen, wenn die Bestimmung in dem Schritt 351 nicht ABS(OBJ2x – OBJ1x) ≤ OBJ1r ist, so geht der Prozeß zu der früheren Routine zurück.

[0186] Auf der anderen Seite, wenn die Bestimmung in dem Schritt 352 nicht ABS(OBJ2y – OBJ1y) ≤ OBJ1r ist, so geht der Prozeß zurück zu der vorherigen Routine.

[0187] Auf der anderen Seite, wenn die Bestimmung in dem Schritt 353 nicht ABS(OBJ2z – OBJ1r) ≤ OBJ1r ist, so geht der Prozeß zurück zu der vorherigen Routine.

[0188] Auf der anderen Seite, wenn der Schritt 350 geendet hat, so schreitet der Prozeß zu einem Schritt 360 voran.

[0189] In dem Schritt 360 bestimmt die CPU **11**, ob das Spielerobjekt **60** in dem unteren Bereich des Hauptkörpers **61** durch ein anderes Objekt getroffen wurde oder nicht. Insbesondere detektiert die CPU **11**, ob oder ob nicht eine Markierung der Trefferbestimmung in dem Markierungsbereich **159F** des RAM **15** gesetzt wird. Wenn der untere Anteil des Hauptkörpers **61** durch ein anderes Objekt getroffen wurde, so schreitet der Prozeß zu einem Schritt 361 voran. Die CPU **11** funktioniert in dem Schritt 361 nach der folgenden Gleichung:

R1 = R1 – 40.

[0190] R1 stellt eine Größe des Schadens dar, gegen den der Hauptkörpe **61** wiederstanden hat, und hat einen Anfangswert von 255 in dieser Ausführungsform. Wenn der untere Anteil des Hauptkörpers **61** einen Schaden erleidet, so wird R1 um 40 subtrahiert. Z. B. ergibt das Erleiden von sieben Schäden R1 ≤ 0.

[0191] In einem Schritt 362 funktioniert die CPU **11** nach der folgenden Gleichung:

R2 (oder R3) = R2 (oder R3) – 15.

[0192] R2 stellt eine Größe des Schadens dar, dem der linke Flügel **62** widerstehen konnte, und hat einen Anfangswert von 60 in dieser Ausführungsform. Ähnlich bezeichnet R3 eine Größe des Schadens, dem der rechte Flügel **63** widerstehen konnte und hat einen Anfangswert von 60. In diesem Schritt wird die linke und die rechte Seite per Zufall ausgewählt, um so einen größeren Schaden für einen ausgewählten Flügel zu liefern. Wenn der linke Flügel **62** oder der rechte Flügel **63** einen Schaden erleidet, so wird R2 oder R3 um 15 subtrahiert. Z. B. ergibt ein viermaliger Schaden R2 (oder R3) = 0.

[0193] In einem Schritt 363 bestimmt die CPU **11**, ob oder oh nicht der obere Anteil des Hauptkörpers **61** des Spielerobjektes **60** von einem anderen Objekt geschlagen wird. Wenn der obere Anteil der Hauptkörpers **61** von einem anderen Objekt geschlagen wird, schreitet der Prozeß weiter zu einem Schritt 364.

[0194] Die CPU **11** bestimmt im Schritt 364, ob R1 ≤ 0 ist oder nicht. In anderen Worten, sie bestimmt, ob oder ob nicht das Spielerobjekt **60** in einem Ausmaß ist, daß das Spiel fortgesetzt werden kann. Wenn nicht R1 ≤ 0 ist, schreitet der Prozeß zu einem Schritt 365 voran.

[0195] In dem Schritt 365 fügt die CPU **11** den Hauptkörper **61** in die Anzeigenliste ein. Der Hauptkörper **61** ist ein Objekt, das den Mittelanteil des Spielerobjekts **60** bildet.

[0196] In einem Schritt 366 bestimmt die CPU **11**, ob oder oh nicht R2 und R3 ≤ 0 ist. In anderen Worten gesagt, sie bestimmt, ob oder ob nicht der Schaden am linken Flügel **62** ein Ausmaß hat, daß der linke Flügel tragend ist, und der Schaden an dem rechten Flügel **63** ein Ausmaß hat, daß der rechte Flügel tragend ist. Wenn nicht R2 und R3 ≤ 0 ist, schreitet der Prozeß weiter zum Schritt 367.

[0197] Im Schritt 367 bestimmt die CPU **11**, ob oder ob nicht R2 ≤ 0 ist. In anderen Worten, sie bestimmt, ob oder ob nicht der Schaden an dem linken Flügel **62** ein Ausmaß hat, daß der linke Flügel tragend ist. Wenn nicht R2 ≤ 0 ist, schreitet der Prozeß weiter zum Schritt 368.

[0198] Im Schritt 368 bestimmt die CPU **11**, ob oder ob nicht R3 ≤ 0 ist. In anderen Worten gesagt, sie bestimmt, ob der Schaden an dem rechten Flügel **62** ein Ausmaß hat, daß der rechte Flügel tragend ist. Wenn nicht R3 ≤ 0 ist, schreitet der Prozeß weiter zum Schritt 369.

[0199] Im Schritt 369 fügt die CPU **11** einen komplettes linkes Flügelobjekt **60L** und ein komplettes linkes Flügelobjekt **60L** in die Anzeigenliste ein. Mit dieser Eintragung kann die Anzeige **30** ein Spielerobjekt anzeigen, wie in Fig. 35 gezeigt. Der Prozeß kehrt dann zurück zu der früheren Routine, um mit Schritt 8 fortzufahren.

[0200] Auf der anderen Seite, wenn R2 und R3 ≤ 0 beim Schritt 366 sind, schreitet der Prozeß voran zu einem Schritt 370. Im Schritt 370 trägt die CPU **11** ein linkes Flügelobjekt **60L**, welches keinen linken Flügel **62** hat, und ein rechtes Flügelobjekt, welches keinen rechten Flügel **63** hat, in die Anzeigenliste ein. Mit dieser Eintragung zeigt die Anzeige **30** ein Spielerobjekt **60** an, welches den linken Flügel **6** und den rechten Flügel **63** verliert als Teile des linken Flügels und des rechten Flügels, wie in Fig. 40 gezeigt wird. Der Prozeß kehrt dann zurück zu der früheren Routine, um mit Schritt 8 fortzufahren.

[0201] Währenddessen, wenn R2 ≤ 0 beim Schritt 371 ist, schreitet der Prozeß voran zum Schritt 371. Im Schritt 371 trägt die CPU **11** ein linkes Flügelobjekt **60L**, welches den linken Flügel **62** verliert, und ein komplettes rechtes Flügelobjekt **60R** auf der Anzeigenliste ein. Diese Eintragung veranlaßt die Anzeige **30**, ein Spielerobjekt anzuzeigen, welches den linken Flügel **62** verliert, als einen Teil des linken Flügels, jedoch mit dem normalen rechten Flügel, wie in Fig. 41 gezeigt ist. Der Prozeß kehrt dann zu der früheren Routine zurück, um zum Schritt 8 fortzuschreiten.

[0202] Auf der anderen Seite, wenn R3 ≤ 0 im Schritt 368 ist, schreitet der Schritt fort zum Schritt 372. Beim Schritt 372 trägt die CPU **11** ein rechtes Flügelobjekt **60R**, welches einen rechten Flügel **63** verliert, und einen komplettes linkes Flügelobjekt **60L** in die Anzeigenliste ein. Diese Eintragung veranlaßt die Anzeige **30**, das Spielerobjekt **60** anzuzeigen, welches den rechten Flügel **63** verliert als einen Teil des rechten Flügels, jedoch mit dem normalen linken Flügel, umgekehrt wie bei dem Spielerobjekt **60** in Fig. 41. Der Prozeß kehrt dann zu der früheren Routine zurück, um mit Schritt 8 fortzufahren.

[0203] Auf der anderen Seite, wenn R1 ≤ 0 beim Schritt 364 ist, schreitet der Prozeß fort zu einem Schritt 373.

[0204] Beim Schritt 373 führt die CPU **11** einen Detonationsprazeß für das Spielerobjekt **60** aus. Beim Ausführen des Detonationsprozesses, werden andere Prozesse über eine Vielzahl von Rahmen gesprungen, um eine Szene anzuzeigen, bei der das Spielerobjekt **60** auf der Anzeige **30** explodiert. Danach wird von dem Wert des Registrierungsbereiches **159** in der RAM **15** 1 subtrahiert, und der Prozeß schreitet voran zu einem Schritt **17**. Wenn das Spiel nicht beendet wird, wird das Spiel von einer vorbestimmten Position in dem Spielfeld fortgesetzt.

[0205] Währenddessen, wenn der obere Anteil des Hauptkörpers **61** nicht von einem anderen Objekt im Schritt 363 geschlagen wird, schreitet der Prozeß fort zu einem Schritt 374.

[0206] In dem Schritt 374 führt die CPU **11** einen Prozeß durch, um das Spielerobjekt **60** aufwärts aber eine Vielzahl von Rahmen zu bewegen. Diese Prozeß ermöglicht eine Darstellung, bei der der untere Anteil des Spielerobjekts **60** mit einem anderen Objekt kollidiert und umgelegt wird. Der Prozeß schreitet dann fort zum Schritt 364.

[0207] Auf der anderen Seite, wenn bei dem Schritt 360 der untere Anteil des Hauptkörpers **61** nicht von einem anderen Objekt geschlagen wird, fährt der Prozeß fort zu einem Schritt 370.

[0208] In dem Schritt 375 bestimmt die CPU **11**, ob oder ob nicht der obere Anteil des Hauptkörpers **61** des Spielerobjektes **60** von einem anderen Objekt geschlagen wird oder nicht. Wenn der obere Anteil des Hauptkörpers **61** von einem anderen Objekt geschlagen wird, schreitet der Prozeß fort zu einem Schritt 376. In dem Schritt 376, funktioniert die CPU **11** nach der folgenden Gleichung:

R1 = R1 – 40.

[0209] In einem Schritt 377, funktioniert die CPU **11** nach der folgenden Gleichung:

R2 (oder R3) = R2 (oder R3) – 15.

[0210] In einem Schritt 378 bewirkt die CPU **11**, daß das Spielerobjekt **60** über eine Vielzahl von Rahmen nach unten bewegt wird. Dieser Prozeß ermöglicht eine Darstellung, als ob der obere Anteil des Spielerobjektes **60** mit einem anderen Objekt kollidieren und umgelegt werden würde. Dann schreitet der Prozeß fort zum Schritt 364.

[0211] Auf der anderen Seite, wenn ein anderes Objekt nicht gegen den oberen Anteil des Hauptkörpers

61 in dem Schritt 375 schlägt, schreitet der Prozeß voran zu einem Schritt 380. Bei dem Schritt 380 bestimmt die CPU **11**, ob oder ob nicht der Hauptkörper **61** und das rechte Flügelobjekt **60R** des Spielerobjektes **60** nicht von einem anderen Objekt geschlagen wird, sondern das linke Flügelobjekt **60L** nur von einem anderen Objekt geschlagen wird. Wenn das linke Flügelobjekt **60L** nur von einem anderen Objekt geschlagen wird, schreitet der Prozeß voran zu einem Schritt 381. Beim dem Schritt 381 funktioniert die CPU **11** nach der folgenden Gleichung:

R2 = R2 – 20.

[0212] Dann schreitet der Prozeß voran zu einem Schritt 365.

[0213] Auf der anderen Seite, wenn nur das linke Flügelobjekt **60L** nicht von einem anderen Objekt bei dem Schritt 380 geschlagen wird, schreitet der Prozeß voran zu einem Schritt 382. In dem Schritt 382 bestimmt die CPU **11**, ob oder ob nicht der Hauptkörper **61** und das linke Flügelobjekt **60L** des Spielerobjektes **60** von einem anderen Objekt geschlagen wird, sondern nur das rechte Flügelobjekt **60L** von einem anderen Objekt geschlagen wird. Wenn nur das rechte Flügelobjekt **60R** von einem anderen Objekt geschlagen wird, schreitet der Prozeß voran zu einem Schritt 383.

[0214] In dem Schritt 383 funktioniert die CPU **11** nach der folgenden Gleichung:

R3 = R3 – 20.

[0215] Der Prozeß schreitet dann voran zu einem Schritt 365.

[0216] Währenddessen, wenn in dem Schritt 382 nur das rechte Flügelobjekt **60R** nicht von einem anderen Objekt geschlagen wird, schreitet der Prozeß voran zu dem Schritt 365.

[0217] Nun werden Erklärungen zu einem anderen Ausführungsbeispiel für die Behandlung des Spielerobjekts gegeben. Dieses Ausführungsbeispiel ist ein Beispiel, das ein linkes Flügelobjekt **60L** und ein rechtes Fogelobjekt **60R** besitzt, wobei jedes aus drei Anteilen anstatt aus zwei Anteilen besteht. Insbesondere sind der linke Flügel und der rechte Flügel jeweils aus drei Anteilen gebildet, so daß sie auf der Anzeige **30** dargestellt werden können, als seien sie zerbrochen oder durch einen Schaden von außen verlorengegangen. Wie in Fig. 35 gezeigt ist, beinhalten das linke Flügelobjekt **60L** und das rechte Flügelobjekt **60R** einen linken Flügel **62** und einen linken Flügel **64**, einen rechten Flügel **63** und einen rechten Flügel **65**. Z. B. wird eine Bildanzeige auf der Anzeige **30** dargestellt, bei der das linke Flügelobjekt **60L**, wenn es einen kleinen Schaden erleidet, z. B. 0 < R2 ≤ 30, den linken Flügel **62** verliert, wie in Fig. 41 gezeigt ist, und den linken Flügel **64** verliert, wie in Fig. 43 gezeigt wird, wenn es einen ernsteren Schaden erleidet (z. B. R2 ≤ 0). Der Prozeß wird ähnlich ausgeführt für den rechten Flügel **63** und den rechten Flügel **65**.

[0218] Bei der Ausführung der Schritte 321–325 in Fig. 33 und der Schritte 332–336 in Fig. 36, kann die Bewegung des Spielerobjektes **60** in diversifizierter Weise gesteuert werden, indem zwei Stufen der Bewegungssteuerung jeweils rechts und links vom Spielerobjekt **60** bereitgestellt werden, abhängig vom Zustand des linken Flügelobjekts **60L** und des rechten Flügelobjekts **60R**. Dementsprechend wird dem Spieler ein noch größeres Realitätsgefühl oder Leistungsgefühl beim Spiel vermittelt.

[0219] Nun werden detaillierte Erklärungen zu den Fig. 44 und Fig. 45 geliefert, die Subroutine-Flußdiagramme des Schritts 8 der Fig. 15 darstellen. Beim Schritt 500 bestimmt die CPU **11**, ob oder ob nicht die All-Bereich-Modusmarkierung des Markierungsbereiches von dem Markierungsbereich **159F** der RAM **15** gesetzt ist. Wenn die All-Bereich-Modusmarkierung des Markierungsberiches von dem Markierungsbereich **159F** nicht gesetzt ist, schreitet der Prozeß voran zu einem Schritt 501. In dem Schritt 501, funktioniert die CPU **11** nach der folgenden Gleichung:

Xc = (Xa – X0)·0.8.

[0220] Xc ist eine X-Koordinaten-Position einer Kamera, die das Spielerobjekt **60** fotografiert, ähnlich zu der Fig. 34. Diese Kamera ist eine hypothetische Kamera, die bestimmt, in welchem Winkel ein Objekt in einem dreidimensionalen Raum auf der Anzeige **30** angezeigt wird. X0 ein ein Abweichwert, wenn es eine Abweichung zwischen dem Mittelpunkt der X-Koordinate gibt, auf der das Spielerobjekt **60** sich bewegt, und ein Ursprung der X-Koordinate des dreidimensionalen Raumes. Dieser Wert wird hauptsächlich verwendet, wenn der Ablauf sich verzweigt. In diesem Ausführungsbeispiel wird X0 als 0 genommen.

[0221] In einem Schritt 502 funktioniert die CPU **11** nach der folgenden Gleichung:

Yc = (Ya – Y0)·0.8.

[0222] Yc ist eine Y-Koordinaten-Position der Kamera, die das Spielerobjekt **60** fotografiert. Y0 ist ein Abweichungswert, wenn es eine Abweichung zwischen dem Mittelpunkt der Y-Koordinate gibt, auf der das Spielerobjekt **60** sich bewegt, und einen Ursprung der Y-Koordinate des dreidimensionalen Raumes. Dieser Wert wird hauptsächlich verwendet, wenn der Ablauf sich verzweigt. In diesem Ausführungsbeispiel wird Y0 als 0 genommen.

[0223] In einem Schritt 503 bestimmt die CPU **11**, ob der Spieler den Schalterknopf C1 drückt oder nicht. Wenn der Spieler den Schalterknopf C1 nicht drückt, schreitet der Prozeß weiter zu einem Schritt 504.

[0224] In dem Schritt 504 bestimmt die CPU **11**, ob der Spieler den Schalterknop **47Cd** drückt oder nicht. Wenn der Spieler den Schalterknopf **47Cd** nicht drückt, schreitet der Prozeß weiter zu einem Schritt 505. In dem Schritt 505 funktioniert die CPU **11** nach der folgenden Gleichung:

Zc = Za + 400.

[0225] Zc- ist ein Z-Achsen-Wert der Kamera, die das Spielerobjekt **60** fotografiert. Dementsprechend stellt Zc eine Z-Achsen-Koordinaten-Position dar, die um 400 in die Z-Achsen-Richtung von der Position des Spielerobjekts aus bewegt wird. In einem Schritt 506, wird die Fotografierrichtung der Kamera in eine Ablaufrichtung festgesetzt. Sie wird in eine Z-Achsen-Richtung festgesetzt, wie es in Fig. 34 gezeigt wird.

[0226] Auf der anderen Seite, wenn der Spieler den Knopfsschalter C1 in dem Schritt 503 drückt, so schreitet der Prozeß zu einem Schritt 507 voran. In dem Schritt 507 funktioniert die CPU **11** nach der folgenden Gleichung:

Zc = (Za + abs(Zα)) + 400.

[0227] abs () stellt einen absoluten Wert einer Zahl dar, die innerhalb () gegeben wird. Zα ist ein Wert, mit welchem das Spielerobjekt **60** von der Kamera weggeht. Die Addition von abs(Zα) für Zc liefert auf diese Art eine Anzeige, ob das Spielerobjekt, welches auf der Anzeige **30** angezeigt wird, schnell in die Tiefenrichtung des Bildschirmes weggeflogen ist. Der Prozeß schreitet dann zu dem Schritt 506 voran.

[0228] Währenddessen, wenn der Spieler den Knopfschalter **47Cd** in dem Schritt 504 drückt, schreitet der Prozeß zu einem Schritt 508 voran.

[0229] In dem Schritt 508 funktioniert die CPU **11** nach der folgenden Gleichung:

Zc = (Za – abs(Zβ)) + 400.

[0230] Zβ ist ein Wert, mit welchem die Kamera sich dem Spielerobjekt **60** nähert, wenn das Spielerobjekt **60** eine Bremse verwendet. Die Subtrahierung von abs(Zβ) von Zc liefert auf diese Weise eine Anzeige, ob das Spielerobjekt **60**, welches auf der Anzeige **30** angezeigt wird, schnell in Richtung auf den Bildschirm zurückgezogen wurde. Der Prozeß schreitet dann zu dem Schritt 506 voran. Auf der anderen Seite schreitet der Prozeß in dem Schritt 500, wenn die Markierung des All-Bereich-Modus in dem Markierungsbereich **159F** festgesetzt ist, zu einem Schritt 510 in **Fig.** 55 voran.

[0231] In dem Schritt 510 bestimmt die CPU **11**, ob der Spieler den Knopfschalter C1 drückt oder nicht. Wenn der Spieler nicht den Knopfschalter C1 drückt, so schreitet der Prozeß zu einem Schritt 511 voran.

[0232] In dem Schritt 511 bestimmt die CPU **11**, ob der Spieler den Knopfschalter **47Cd** drückt oder nicht. Wenn der Spieler nicht den Knopfschalter **47Cd** drückt, so schreitet der Prozeß zu einem Schritt 512 voran. In dem Schritt 512 funktioniert die CPU **11** nach der folgenden Gleichung:

Xr = 2.0·θx1·cosθx2

Zr = 2.0·θx1·sinθx2

Xc = Xa + 400·sinθx2 + Xr

Zc = Za + 400·cosθx2 + Zr

Yc = 0.8·Ya

[0233] Ähnlich zu Fig. 37 ist 2.0·θx1 ein Wert zum Abweichen der Kamera in die (–X)-Achsen-Richtung von dem hinteren Bereich des Spielerobjektes **60** in dem Spielerraum. Xr ist ein Wert eines X-Koordinaten-Wertes des Spielraumes, welcher von einem X-Koordinaten-Wert des Spielerraumes umgewandelt wird. Zr ist ein Wert eines Z-Koordinaten-Wertes eines Spielraumes, welcher von einem Z-Koordinaten-Wert eines Spielerraumes umgewandelt wird. Auf diese Art kann die Kamera in ihre Position in eine Richtung in Richtung, in welcher das Spielerobjekt **60** ausgelenkt ist, bewegt werden, im Verhältnis zu der Auslenkung des Spielerobjektes **60**. Durch Demonstrieren eines Beispieles, welches auf der Anzeige **30** angezeigt wird, wird das Spielerobjekt **60**, wenn es geradeaus fliegt, in einem Zentrum der linken und rechten Richtungen des Bildschirmes angezeigt, wie es in Fig. 46 gezeigt wird. Jedoch wird, wenn das Spielerobjekt **60** linksseitig bewegt wird, wie es in Fig. 47 gezeigt wird, es auf der rechten Seite in der Richtung der linken und rechten Seiten des Bildschirmes angezeigt. Ebenso wird das Spielerobjekt **60** auf der linken Seite in der linken und rechten Richtung des Bildschirmes angezeigt, wenn es in die rechte Richtung bewegt wird. Eine solche Anzeige erleichtert es, die Sicht eines Feindes wahrzunehmen. Dementsprechend ist es für den Spieler leicht, sich bei einem Feind zu helfen, welches ihn an einem Spiel mit mehr Komfort erfreuen läßt. In einem Schritt 513 setzt die CPU **11** die Fotografierrichtung der Kamera auf eine –Z2-Achsen-Richtung fest, wie es in Fig. 32 gezeigt wird.

[0234] In dem Schritt 510 wird auf der anderen Seite, wenn der Spieler den Knopfschalter C1 drückt, der

Prozeß zu einem Schritt 514 voranschreiten. In dem Schritt 514 funktioniert die CPU **11** nach den folgenden Gleichungen:

$$Xr = 2.0 \cdot \theta x1 \cdot \cos\theta x2$$

$$Zr = 2.0 \cdot \theta x1 \cdot \sin\theta x2$$

$$Xc = Xa + (400 + abs(Z\alpha)) \cdot \sin\theta x2 + Xr$$

$$Zc = Za + (400 + abs(Z\alpha)) \cdot \cos\theta x2 + Zr$$

$$Yc = 0.8 \cdot Ya.$$

[0235] Der Prozeß schreitet dann zu dem Schritt 513 voran.

[0236] Währenddessen, wenn der Spieler den Knopfschalter **47Cd** in dem Schritt 511 drückt, schreitet der Prozeß zu einem Schritt 515 voran. In dem Schritt 515 funktioniert die CPU **11** nach den folgenden Gleichungen:

$$Xr = 2.0 \cdot \theta x1 \cdot \cos\theta x2$$

$$Zr = 2.0 \cdot \theta x1 \cdot \sin\theta x2$$

$$Xc = Xa + (400 - abs(Z\beta)) \cdot \sin\theta x2 + Xr$$

$$Zc = Za + (400 - abs(Z\beta)) \cdot \cos\theta x2 + Zr$$

$$Yc = 0.8 \cdot Ya$$

[0237] Dann schreitet der Prozeß zu dem Schritt 513 voran.

[0238] Mit Bezug auf die Fig. 24–Fig. 26 werden Erklärungen für die Funktion einer Subroutine für den Mitgliedsobjekte-Prozeß durchgeführt. Es wird in einem Schritt 201 bestimmt, ob ein erstes Mitglied anwesend ist oder nicht. Wenn das Mitglied anwesend ist, so wird ein Prozeß in einem Schritt 202 für das erste Mitgliedsobjekt durchgeführt. Danach wird in einem Schritt 203 bestimmt, ob ein zweites Mitglied existiert. Wenn das Mitglied existiert, so wird ein Prozeß in einem Schritt 204 für das zweite Mitgliedsobjekt durchgeführt. Ähnlich wird eine Bestimmung bei Anwesenheit oder Abwesenheit eines dritten Mitglieds durchgeführt und der Prozeß für das dritte Mitgliedsobjekt wird in den Schritten 205 bis 206 ausgeführt. Hierin sind die Prozesse für das erste bis dritte Mitgliedsobjekt, die in den Schritten 202, 204, 206 gezeigt werden, dieselben, mit der Ausnahme des Unterschieds der Art des Mitglieds und sie werden im konkreten durch einen Subroutinen-Prozeß (Schritte 211–230) der Fig. 25 und Fig. 26 realisiert.

[0239] Das heißt, es wird in einem Schritt 211 bestimmt, daß der Prozeß nicht in einem Suspensionsprozeß ist, und ein Bewegungsprozeß wird in einem Schritt 212 zum Bewegen irgendeines der ersten bis dritten Mitglieder. Die Bestimmung wird in einem Schritt 213 gemacht, ob der Abstand innerhalb eines Bereichs, der für den Feind angreifbar ist, ist. Wenn der Abstand innerhalb eines angreifbaren Bereichs ist, so wird ein Prozeß des Durchführens eines Angriffs auf das Feinobjekt in einem Schritt 214 (Prozeß des Berechnens und Anzeigens zum Abschießen einer Strahlenbombe) durchgeführt. Es wird bestimmt, ob irgendein Kamerad durch einen Feind in einem Schritt 215 verfolgt wird. Wenn die Verfolgung bestimmt wird, so wird eine Bestimmung durchgeführt, ob er innerhalb einer angreifbaren Entfernung von dem Feind entfernt ist oder nicht. Wenn der Feind innerhalb der angreifbaren Entfernung ist, so wird ein Prozeß des Anzeigens der Phrase 5 (z. B. „Hilfe") in einem Schritt 217 durchgeführt, und ein Klangausgabeprozeß für die Phrase 5 wird in einem Schritt 218 durchgeführt. Nebenbei gesagt, wenn die Bestimmung in dem Schritt 215 und/oder 216 eine unterschiedliche ist (nein), so wird dann in einem Schritt 219 bestimmt, ob irgendeinem der Mitgliedsobjekte von dem Spielerobjekt geholfen wurde oder nicht. Wenn das Geholfenwerden bestimmt wird, so wird ein Anzeigeprozeß für die Phrase 6 (z. B. „Wir haben geholfen!") in einem Schritt 220 durchgeführt. Ein Klangausgabegrozeß für die Phase 6 wird in einem Schritt 221 durchgeführt.

[0240] In einem Schritt 222 wird eine Schlagbestimmung (z. B. die Bestimmung, ob ein Mitglied durch einen Angriff in einem Schritt 254, auf welchen später zurückgekommen wird, geschlagen wird oder nicht) durchgeführt, wenn das Mitglied von dem Feind angegriffen wird. Es wird in einem Schritt 223 bestimmt ob er durch eine Kugel des Feindes getroffen worden ist oder nicht. Wenn er getroffen worden ist, so wird ein Prozeß zum Reduzieren der Schadenssubtraktion des Mitglieds an dem Wert von dem Register R4 in einem Schritt 224 durchgeführt. Es wird in einem Schritt 225 bestimmt, ob der Schadenszuführungsbetrag des Registers R4 100 oder niedriger ist oder nicht. Wenn er höher ist, dann fährt der Prozeß zu einem Schritt 229 fort, während, wenn er nicht höher ist, der Prozeß zu einem Schritt 226 fortfährt. In dem Schritt 226 wird ein Anzeigeprozeß für eine Nachricht durchgeführt, daß der Kamerad, der mit einem bestimmten Betrag oder mehr beschädigt den Kampf einstellt, um zu einer Basis zurückzukehren, und ein Klangausgabeprozeß wird deshalb in einem Schritt 227 durchgeführt. Es wird in einem Schritt 228 bestimmt, ob ein Feuereinstellungsprozeß beendet ist oder nicht. Wenn er nicht beendet ist, so wird in einem Schritt 229 ein Prozeß zum Einfügen des Kameradenobjekts, der verarbeitet wird, in die Anzeigeliste durchgeführt. Wenn er geendet hat, so wird die Mitgliedermarkierung F1 ausgeschaltet, und dann geht der Prozeß zurück zu der Hauptroutine.

[0241] Mit Bezug auf Fig. 27 werden Erklärungen für die Funktion einer Subroutine für den Feindobjekt-Prozeß (Schritt 10) durchgeführt werden. Bei einem Schritt 241 wird das Register R5 auf 1 festgesetzt, um zeitlich die Zahl der Feinobjekte zu speichern. Es wird in einem Schritt 242 bestimmt, daß es ein Feindobjekt basierend auf dem Wert des Registers R5 gibt. In einem Schritt 243 wird eine Subroutine (Fig. 28) zum Verarbeiten bezüglich dem durchgeführt, was die Anzahl der Feindobjekte ist. Dann wird 1 zu dem Register R5 in einem Schritt 244 hinzugefügt. Es wird in einem Schritt 245 bestimmt, ob ein Prozeß des Anzeigens aller Feindobjekte in der Zahl, welche durch das Programm gesetzt wird, beendet ist oder nicht. Wenn all die Prozesse nicht geendet haben, so geht der Prozeß zurück zu dem Schritt 242, um die Prozesse der Schritte 242–245 zu wiederholen.

[0242] Mit Bezug auf Fig. 28 werden Erklärungen im Detail über einen Prozeß für eines der Feindobjekte gemacht. In einem Schritt 251 wird bestimmt, daß das Feindobjekt nicht in einem Explosionsprozeß ist. In einem Schritt 252 wird ein Bewegungsprozeß für das Feindobjekt in der Anzahl, wie es in dem Register (E) gespeichert ist, durchgeführt. Es wird in einem Schritt 253 bestimmt, ob das Spielerobjekt oder das Mitgliedsobjekt innerhalb eines Bereichs des Erschießens ist oder nicht. Wenn sie innerhalb eines Schießbereichs sind, so wird ein Prozeß in einem Schritt 254 zum Durchführen eines Angriffs auf das Spielerobjekt oder das Mitgliedsobjekt, die innerhalb des Schieß-Abstandsbereichs anwesend sind, durchgeführt.

[0243] Währenddessen wird in einem Schritt 255 eine Trefferbestimmung für einen Fall durchgeführt, indem das Spielerobjekt oder das Kameradenobjekt einen Angriff auf das Feindobjekt unternimmt. Es wird in einem Schritt 256 bestimmt, ob die Strahlenbombe, die durch das Spielerobjekt oder das Mitgliedsobjekt abgefeuert wird, das Feindobjekt trifft oder nicht. Wenn der Treffer detektiert wird, so wird ein Prozeß zum Reduzieren eines Schadensbetrags (Subtrahieren von 1 von dem Register F5) des Feindobjekts durchgeführt, wodurch getroffen wird und Punkte an den Spieler geliefert werden (Prozeß des Addieren der Punkte, die abhängig von dem besiegten Feind bestimmt werden, zu dem Register R10-Wert). Es wird in einem Schritt 258 bestimmt, ob der Schadensbetrag 0 oder niedriger wird oder nicht (R5 ≤ 0). Wenn er nicht niedriger ist (R5 ≥ 0), so tritt das Feindobjekt, das zu verarbeiten ist, in die Anzeigeliste in einem Schritt 261 ein. Demgegenüber, wenn es niedriger ist (R5 ≤ 0), so wird ein Prozeß in einem Schritt 259f durchgeführt oder das Explodieren und Verschwinden des Feindobjekts findet statt. Wenn in einem Schritt 260 bestimmt wird, daß der Explosionsablauf geendet hat, so wird in einem Schritt 262 die Markierung des Feinobjekts, das von dem Spielerobjekt angegriffen worden ist, ausgeschaltet, und dann geht der Prozeß zurück zu der Hauptroutine.

[0244] Fig. 48 zeigt ein Flußdiagramm, welches im Detail den Schritt 255 zum Verarbeiten des Feindobjektes von Fig. 28 darstellt.

[0245] In dem herkömmlichen Spielgerät wird der Bereich, in dem das Treffen des Feindes zu bestimmen ist, von vornherein festgesetzt. Dementsprechend ist, wenn der Spieler einen Feind angreift, die momentane Entfernung des Feindes zu gering, um anzugreifen und zu treffen. Dies macht das Spiel ungeeignet schwierig, welches darin resultiert, daß der Spieler seine Lust daran verliert. Im Gegensatz dazu hat die Bestimmung des Feindtreffers gemäß der vorliegenden Erfindung einen Trefferbestimmungsbereich des Feindobjektes, welcher im Verhältnis zu einer Entfernung zwischen dem Feindobjekt und dem Spielerobjekt **60** erweitert ist. Dies ermöglicht einem Spieler, ein Feindobjekt, welches auf gleicher Entfernung vorhanden ist, anzugreifen und zu treffen. Die vorliegende Erfindung hält ein Spiel deshalb davon ab, ausgesprochen schwierig zu werden, und ermöglicht dem Spieler, mit ziemlich relativen Schwierigkeiten zu spielen. Der Spieler kann ein Spiel mit einer hohen Bereitschaft spielen.

[0246] In einem Schritt 600 führt die CPU **11** die Berechnung wie folgt durch:

RAD = (sqrt((OBJ2x – OBJIx)^2 + (OBJ2y – OBJ1y) ^2 + (OBJ2z – OBJIz)^2)/50.

[0247] RAD ist eine Zahl, die sich im Verhältnis zu einer Entfernung zwischen zwei Objekten, die zum Treffen zu bestimmen ist, erhöht. sqrt() ist eine Funktion, welche eine Wurzel von demjenigen darstellt, welches innerhalb () gegeben ist. Das Symbol „^„ ist ein Symbol, um eine erstbeschriebene Zahl mit einer zweitbeschriebenen Zahl hochzusetzen. Dementsprechend ist „^2” ein Quadratzeichen. Das Symbol „” zeigt die Division an. OBJ1 ist ein Objekt, das für das Treffen zu bestimmen ist und bezieht sich hierin auf ein Feindobjekt. OBJ2 ist ein Objekt, welches in Richtung OBJ1 zukommt und bezieht sich hierin auf ein angreifendes Objekt, wie ein Laserobjekt, welches durch das Spielerobjekt **60** gestartet wird. Wenn OBJ2 ein Spielerobjekt **60**, ein Mitgliedsobjekt, ein stationäres Objekt, oder ein angreifendes Objekt, welches durch ein Mitgliedsobjekt abgeschossen wird, ist, so ist die Trefferbestimmung ähnlich durchzuführen zu der von Fig. 37.

[0248] In einem Schritt 601 bestimmt die CPU **11**, ob RAD > 200 ist oder nicht. Wenn RAD > 200 nicht ist, so fährt der Prozeß zu einem Schritt 602 fort.

[0249] In dem Schritt 602 bestimmt die CPU **11**, ob oder ob nicht ABA(OBJ2x – OBJ1x) ≤ OBJ1r + RAD

ist. Wenn ABS(OBJ2x – OBJ1x) ≤ OBJ1r + RAD ist, so schreitet der Prozeß zu einem Schritt 603 voran.

[0250] In dem Schritt 603 bestimmt die CPU **11**, ob oder ob nicht ABS(OBJ2y – OBJ1y) ≤ OBJ1r + RAD ist oder nicht. Wenn ABS(OBJ2y – OBJ1y) ≤ OBJ1r + RAD stillsteht, so schreitet der Prozeß zu einem Schritt 604 voran.

[0251] In dem Schritt 604 bestimmt die CPU **11**, ob oder ob nicht ABS(OBJ2z – OBJ1z) ≤ OBJ1r + RAD ist. Wenn ABS(OBJ2z – OBJ1z) ≤ DBJ1r + RAD ist, so fährt der Prozeß zu einem Schritt 605 fort.

[0252] In dem Schritt 605 bestimmt die CPU **11**, ob ein Treffer zwischen OBJ2 und OBJ1 stattgefunden hat und setzt eine Markierung der Trefferbestimmung in dem Markierungsbereich des RAM **15**.

[0253] Der Prozeß geht dann zurück zu der vorherigen Routine.

[0254] Auf der anderen Seite, wenn ABS(OBJ2x – OBJ1x) ≤ OBJ1r + RAD nicht in dem Schritt 602 stillsteht, so geht der Prozeß zurück zu der vorherigen Routine.

[0255] Auf der anderen Seite, wenn ABS(OBJ2y – OBJ1y) ≤ OBJ1r + RAD nicht in dem Schritt 603 stillsteht, so geht der Prozeß zurück zu der vorherigen Routine.

[0256] Auf der anderen Seite, wenn ABS(OBJ2z – OBJ1z) ≤ OBJ1r + RAD nicht stillsteht in dem Schritt 604, so geht der Prozeß zurück zu der vorherigen Routine.

[0257] Währenddessen, wenn RAD > 200 in dem Schritt 601 ist, so schreitet der Prozeß zu einem Schritt 606 voran. In dem Schritt 606 berechnet die CPU **11** das folgende:

RAD = 200.

[0258] Der Grund, die RAD davon abzuhalten, nicht 200 zu überschreiten, ist, die Bestimmung davon abzuhalten, ungenau und leicht zu werden, trotzdem ein Feind objekt in einer weiten Entfernung von dem Spielerobjekt **60** existiert. Wenn es kein Festsetzen einer oberen Begrenzung des RAD gibt, so könnte ein angreifendes Objekt, welches durch das Spielerobjekt losgelassen wird, auf ein Feindobjekt treffen, das als zu klein auftritt, um auf dem Bildschirm unterschieden zu werden, wodurch das Realitätsgefühl verloren geht.

[0259] Mit Bezug auf Fig. 29 werden Erklärungen für die Funktion einer Subroutine für den stationären Objektprozeß (Schritt 11) gegeben werden. In einem Schritt 271 wird ein stationäres Objektregister (R9) auf 1 festgesetzt. In einem Schritt 272 wird das stationäre Objekt, welches von dem Register (R9) spezifiziert wird, in die Anzeigeliste eintreten. In einem Schritt 273 wird das Register (R9) um 1 aufaddiert. Es wird in einem Schritt 274 bestimmt, ob ein Prozeß zum Anzeigen aller stationärer Objekte in der Zahl, die durch das Programm festgesetzt wird, geendet hat oder nicht. Wenn alle Prozesse nicht geendet haben, so geht der Prozeß zurück zu dem Schritt 272, um die Prozesse der Schritte 272–274 zu wiederholen. Wenn alle Prozesse geendet haben, so geht der Prozeß zurück zu der Hauptroutine.

[0260] Mit Bezug auf die Fig. 30 werden Erklärungen für die Funktion einer Subroutine für den Wiedergabeprozeß (Schritt 12) gegeben werden. In einem Schritt 281 wird ein Koordinateninformations-Prozeß durchgeführt. Der Koordinateninformations-Prozeß wird während des Steuerns des RCP12 durch Transformieren der Koordinaten-Daten von einer Vielzahl von Polygone für das bewegbare Objekt, wie die Feinde, die Spieler und die Mitglieder und die stationären Objekte, wie die Hintergründe, die in den Speicherdatenbereichen **154** des RAM **15** gespeichert sind, in Augenkoordinaten des Beobachters von dem Kamera durchgeführt. Insbesondere werden Berechnungen zum Transformieren der Polygondaten, die die Vielzahl von beweglichen Objekten und stationären Objekten ausmachen, aus den absoluten Koordinaten zu den Kamerakoordinaten-Daten durchgeführt, so daß sie in Bilder umgewandelt werden, wie sie von der Kamera gesehen werden. In einem Schritt 282 wird ein Wiedergabeprozeß für den Rahmenspeicher durchgeführt. Dieser Prozess wird durch Schreiben der Chrominanzdaten, die basierend auf den Textdaten bestimmt werden, in den Bildpufferbereich **152** auf eine Punkt-bei-Punkt- Basis durchgeführt, d. h. in einer Ebene von einem Dreieck, welches das Objekt, das durch die Polygonkoordinaten umrundet wird, nachdem sie in die Kamerakoordinaten transformiert wurden, ausmacht. Zu dieser Zeit werden die Chrominanzdaten für ein näheres Objekt so geschrieben, dass das Objekt, welches in Richtung auf diese (in der Nähe von diesen) steht, vorzugsweise basierend auf den Tiefendaten für jedes Polygon angezeigt wird. Entsprechend werden die Tiefendaten korrespondierend zu den Punkten, die durch die Chrominanzdaten geschrieben werden, zu einer korrespondierenden Adresse in dem Z-Puffer-Bereich **153** geschrieben. Danach geht der Prozess zurück zu der Hauptroutine.

[0261] Die Funktionen der Schritte 281 und 282 werden innerhalb einer konstanten Zeitperiode für jeden Rahmen durchgeführt. Sie werden jedoch in Reihenfolge verarbeitet für die Polygone, die die Vielzahl der Objekte, die auf einer Szene anzuzeigen sind, aufbauen, und werden wiederholt, bis die Prozesse für alle Objekte, die in einer Szene anzuzeigen sind, vollständig sind.

[0262] Mit Bezug auf Fig. 31 werden Erklärungen bezüglich der Funktion einer Subroutine für den Klangprozess (Schritt 13) gemacht. Bei einem Schritt 291 wird bestimmt, ob eine Klangmarkierung eingeschaltet ist oder nicht. Wenn die Bestimmung des Eingeschaltetseins vorhanden ist, so werden die Klangdaten, die auszugeben sind, in einem Schritt 292 ausgewählt. Die ausgewählten Klangdaten werden in einem Schritt 293 leseverarbeitet, und dann geht der Prozess zurück zu der Hauptroutine. Nebenbei bemerkt werden die Klangdaten einer Nachricht, die so gelesen wird, digital analog umgewandelt durch die Klangerzeugungsschaltung **16**, die als Klänge auszugeben sind.

Patentansprüche

1. Videospielvorrichtung zum Spielen eines Videospiels mit mehreren verschiedenen Ablaufen, durch die ein Spieler voranschreiten kann, indem dieser erfolgreich einen der Ablaufe löscht, wobei die Videospielvorrichtung umfasst,
einen Bildschirm, um ein Spielerobjekt (**60**) und ein anderes Objekt (**72a**–**72n**) anzuzeigen und um einen zu spielenden Ablauf aus mehreren Ablaufen auszuwählen, wenn das Videospiel gestartet wird oder nachdem ein Ablauf gelöscht worden ist;
eine erste Zähleinrichtung (**11**, **22**, R10), die ausgebildet ist, eine Punktezahl in dem zu spielenden Ablauf zu zahlen abhängig von einer Beziehung des Spielerobjektes (**60**) zu dem anderen Objekt,
gekennzeichnet durch,
eine Steuereinrichtung (**11**, S19, **159**), die ausgebildet ist, um eine Punktezahl eines gelöschten Ablaufs in eine Speichereinrichtung (**11**, R11–R1n) zu schreiben,
wobei die Steuereinrichtung ausgebildet ist, die Punktezahl für den gelöschten Ablauf auf eine anfängliche Punktezahl zurückzusetzen, wenn der Benutzer mit Hilfe des Bildschirms auswählt, den bereits gelöschten Ablauf erneut zu spielen,
eine zweite Zähleinrichtung (**11**, **159**, R20) zum Bestimmen einer Summe der Punktezahlen der mehreren Ablaufe.

2. Verfahren zum Betrieb einer Videospielvorrichtung zum Spielen eines Videospiels mit mehreren verschiedenen Abläufen, durch die ein Spieler voranschreiten kann, indem dieser erfolgreich einen der Ablaufe löscht, wobei das Verfahren folgende Schritte umfasst,
– Anzeigen eines Spielerobjektes (**60**) und eines anderen Objektes (**72a**–**72n**),
– Auswählen eines zu spielenden Ablaufs aus mehreren Abläufen, wenn das Videospiel gestartet wird oder nachdem ein Ablauf gelöscht worden ist,
– Zählen einer Punktezahl in dem zu spielenden Ablauf abhängig von einer Beziehung des Spielerobjektes (**60**) zu dem anderen Objekt;
gekennzeichnet durch
– Schreiben einer Punktezahl eines gelöschten Ablaufs in eine Speichereinrichtung (**11**, R11–R1n),
– Zurücksetzen der Punktezahl für den gelöschten Ablauf auf eine anfängliche Punktezahl, wenn der Benutzer mit Hilfe des Bildschirms auswählt, den bereits gelöschten Ablauf erneut zu spielen;
– Bestimmen einer Summe der Punktezahlen der mehreren Abläufe

Es folgen 48 Blatt Zeichnungen

Anhängende Zeichnungen

F I G. 1

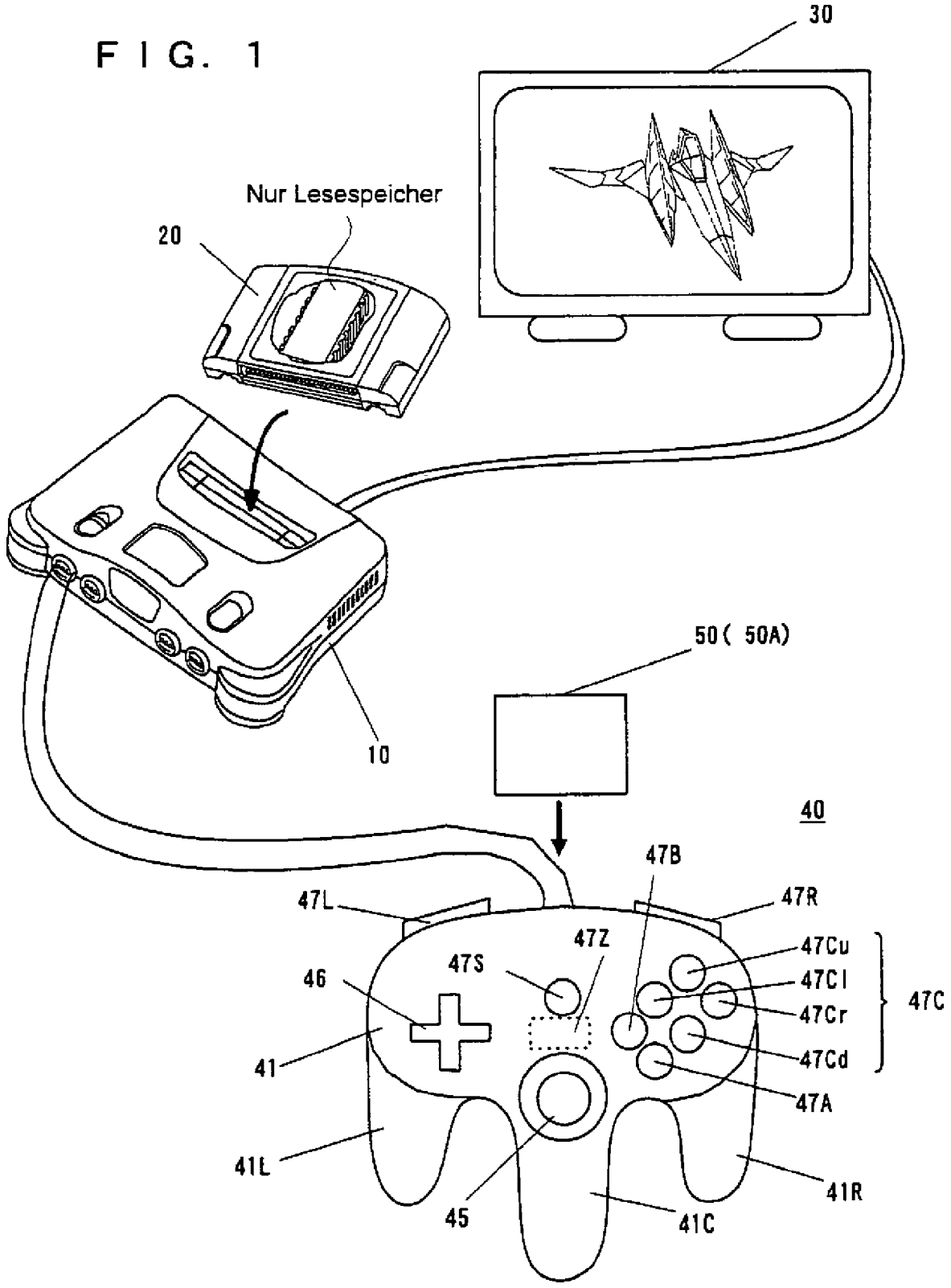


FIG. 2

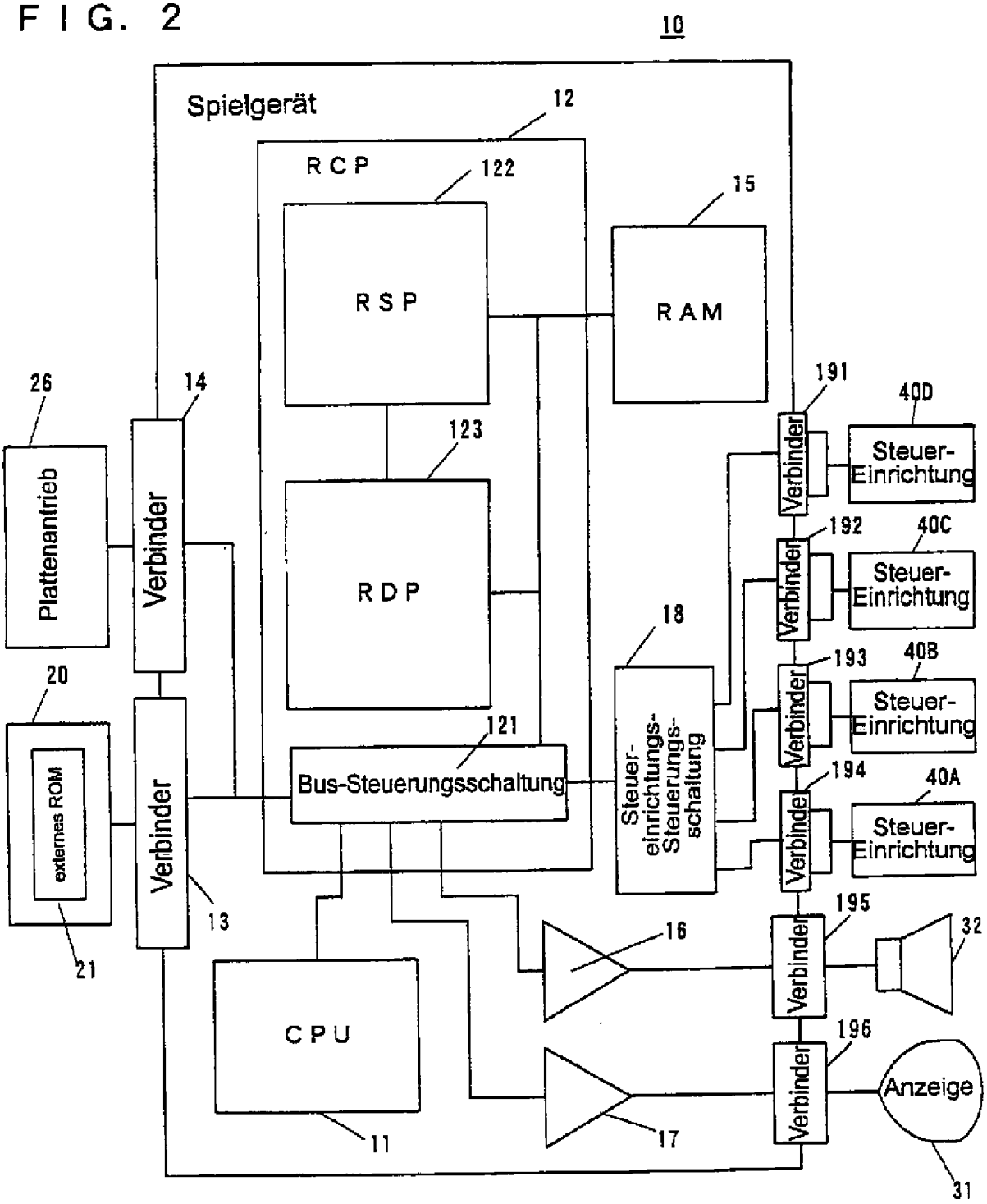

10
Spielgerät
12
RCP
122
RSP
15
RAM
123
RDP
121
Bus-Steuerungsschaltung
18
Steuer-einrichtungs-Steuerungs-schaltung
26
Plattenantrieb
14
Verbinder
20
externes ROM
21
13
CPU
11
16
17
191
192
193
194
195
196
Verbinder
40D
40C
40B
40A
Steuer-Einrichtung
32
Anzeige
31

FIG. 3

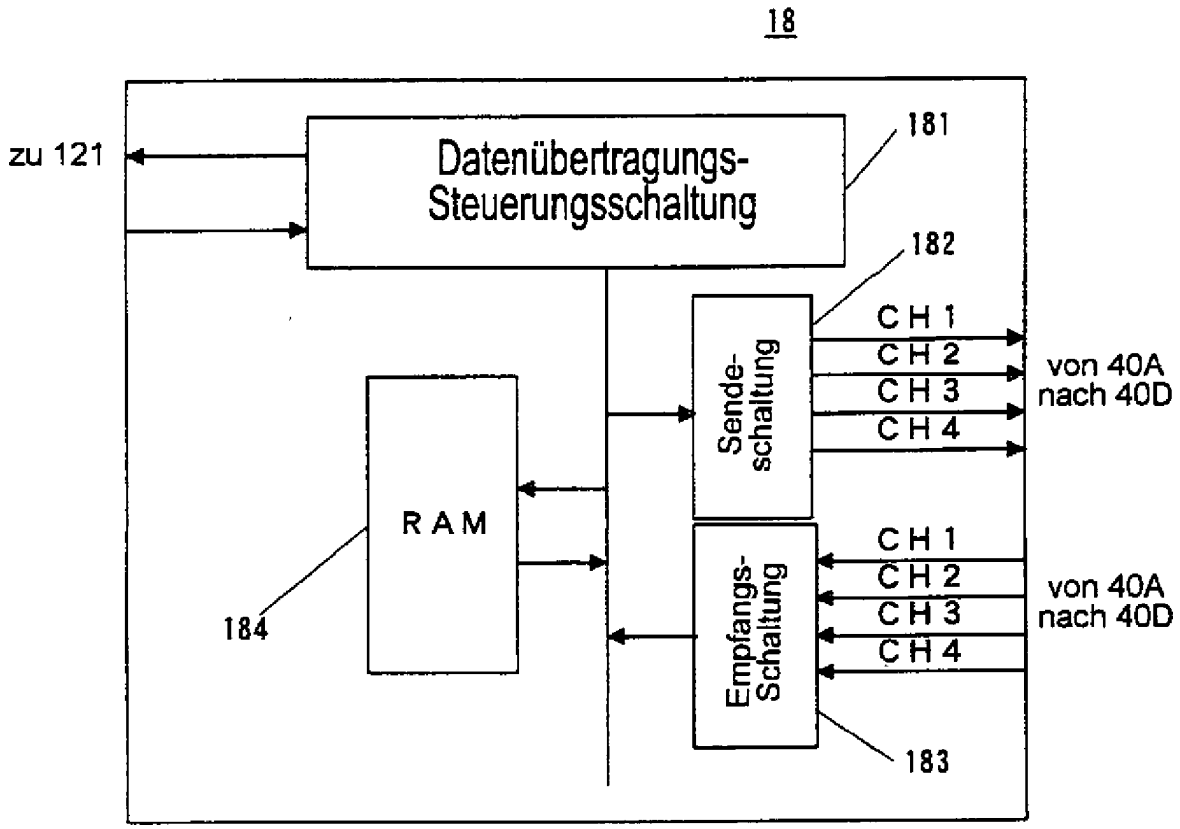

18
zu 121
Datenübertragungs-
Steuerungsschaltung
181
182
Sende-
schaltung
CH 1
CH 2
CH 3
CH 4
von 40A
nach 40D
RAM
184
Empfangs-
Schaltung
CH 1
CH 2
CH 3
CH 4
von 40A
nach 40D
183

FIG. 4

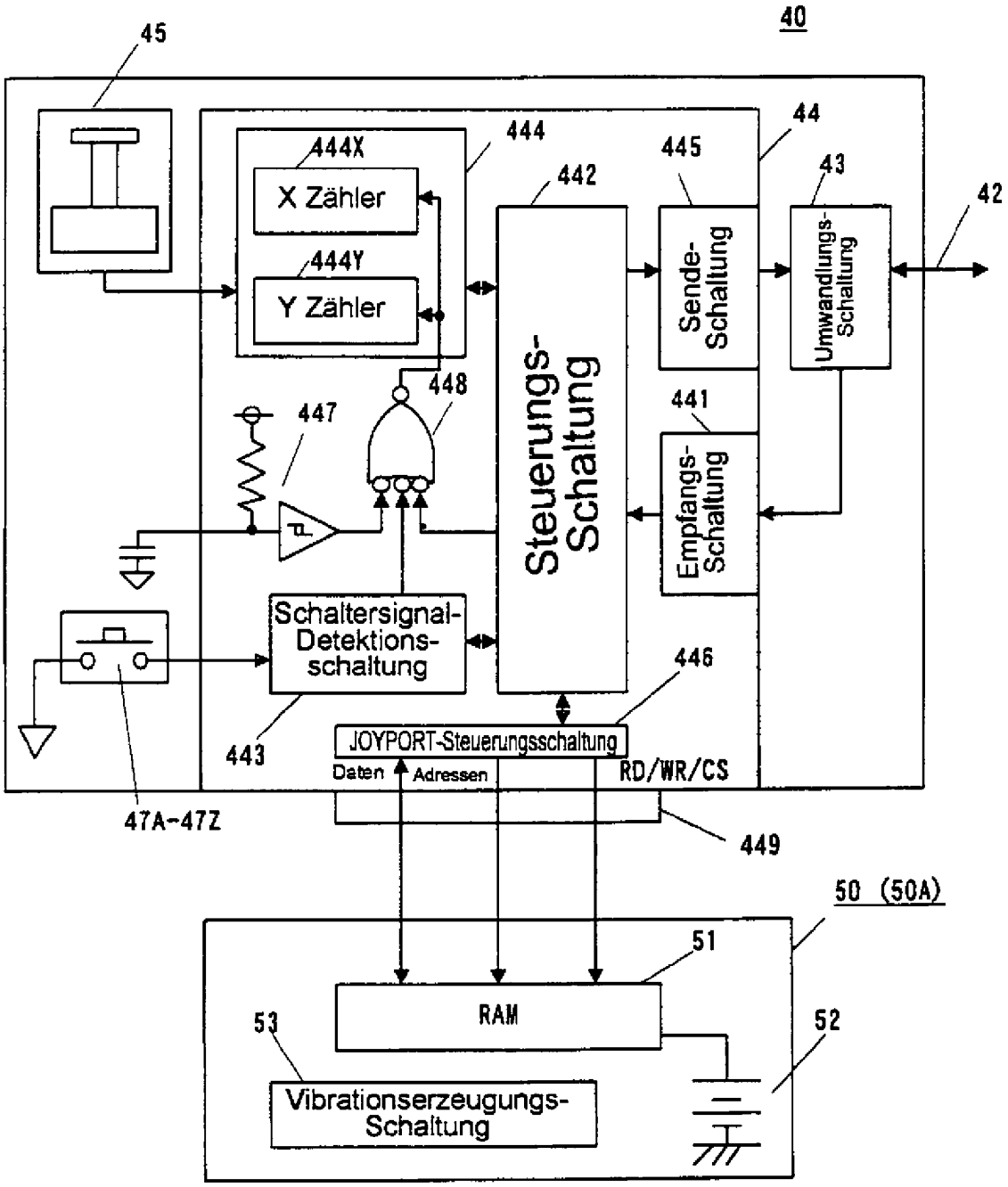

40
45
44
444
444X
X Zähler
444Y
Y Zähler
442
445
43
42
Sende-Schaltung
Umwandlungs-Schaltung
Steuerungs-Schaltung
448
447
441
Empfangs-Schaltung
Schaltersignal-Detektions-schaltung
446
443
JOYPORT-Steuerungsschaltung
Daten
Adressen
RD/WR/CS
47A-47Z
449
50 (50A)
51
RAM
53
52
Vibrationserzeugungs-Schaltung

FIG. 5

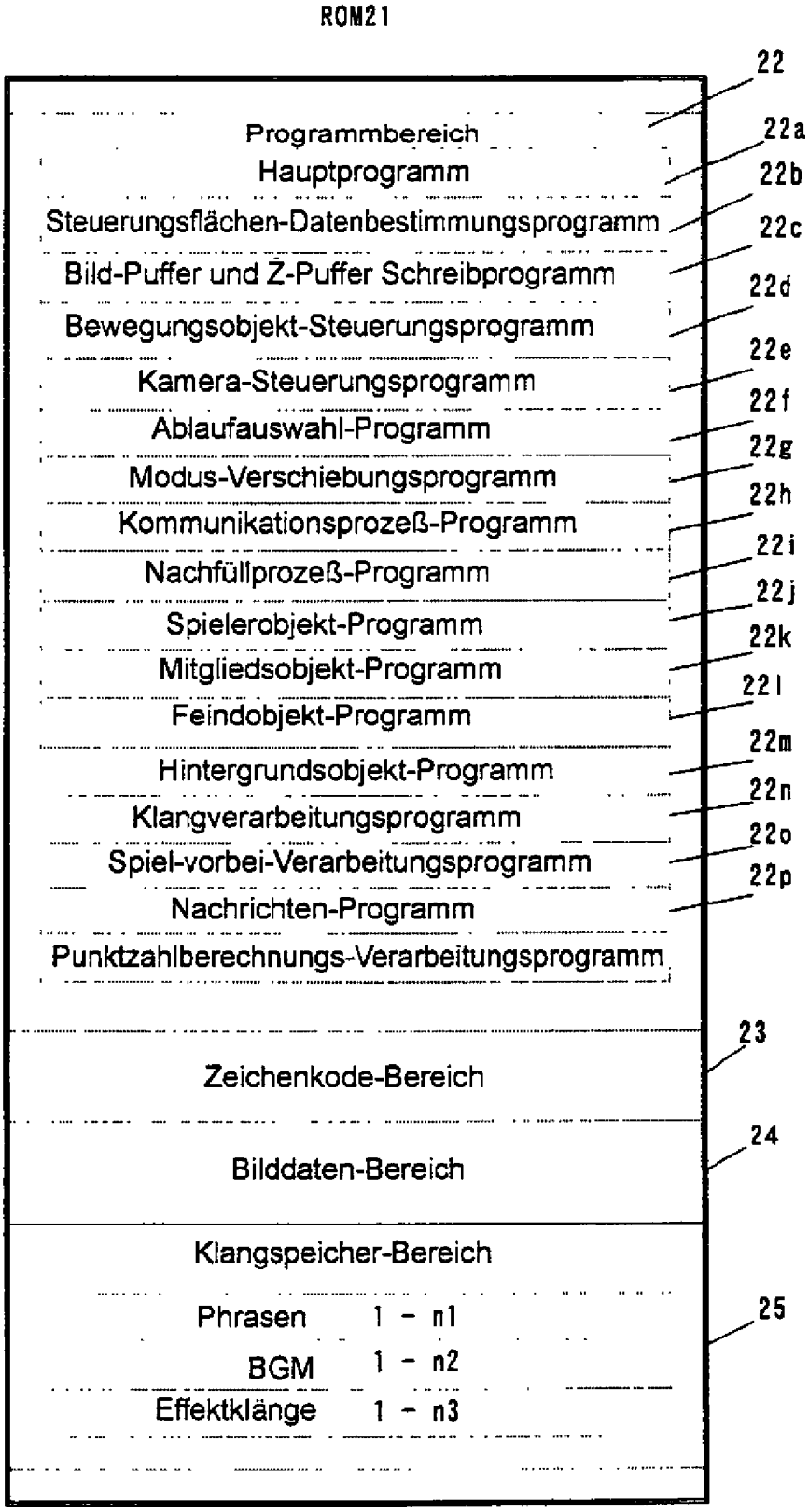

ROM21
22
Programmbereich
22a
Hauptprogramm
22b
Steuerungsflächen-Datenbestimmungsprogramm
22c
Bild-Puffer und Z-Puffer Schreibprogramm
22d
Bewegungsobjekt-Steuerungsprogramm
22e
Kamera-Steuerungsprogramm
22f
Ablaufauswahl-Programm
22g
Modus-Verschiebungsprogramm
22h
Kommunikationsprozeß-Programm
22i
Nachfüllprozeß-Programm
22j
Spielerobjekt-Programm
22k
Mitgliedsobjekt-Programm
22l
Feindobjekt-Programm
22m
Hintergrundsobjekt-Programm
22n
Klangverarbeitungsprogramm
22o
Spiel-vorbei-Verarbeitungsprogramm
22p
Nachrichten-Programm
Punktzahlberechnungs-Verarbeitungsprogramm
23
Zeichenkode-Bereich
24
Bilddaten-Bereich
Klangspeicher-Bereich
25
Phrasen 1 - n1
BGM 1 - n2
Effektklänge 1 - n3

FIG. 6

Details des Bilddaten-Bereichs 24

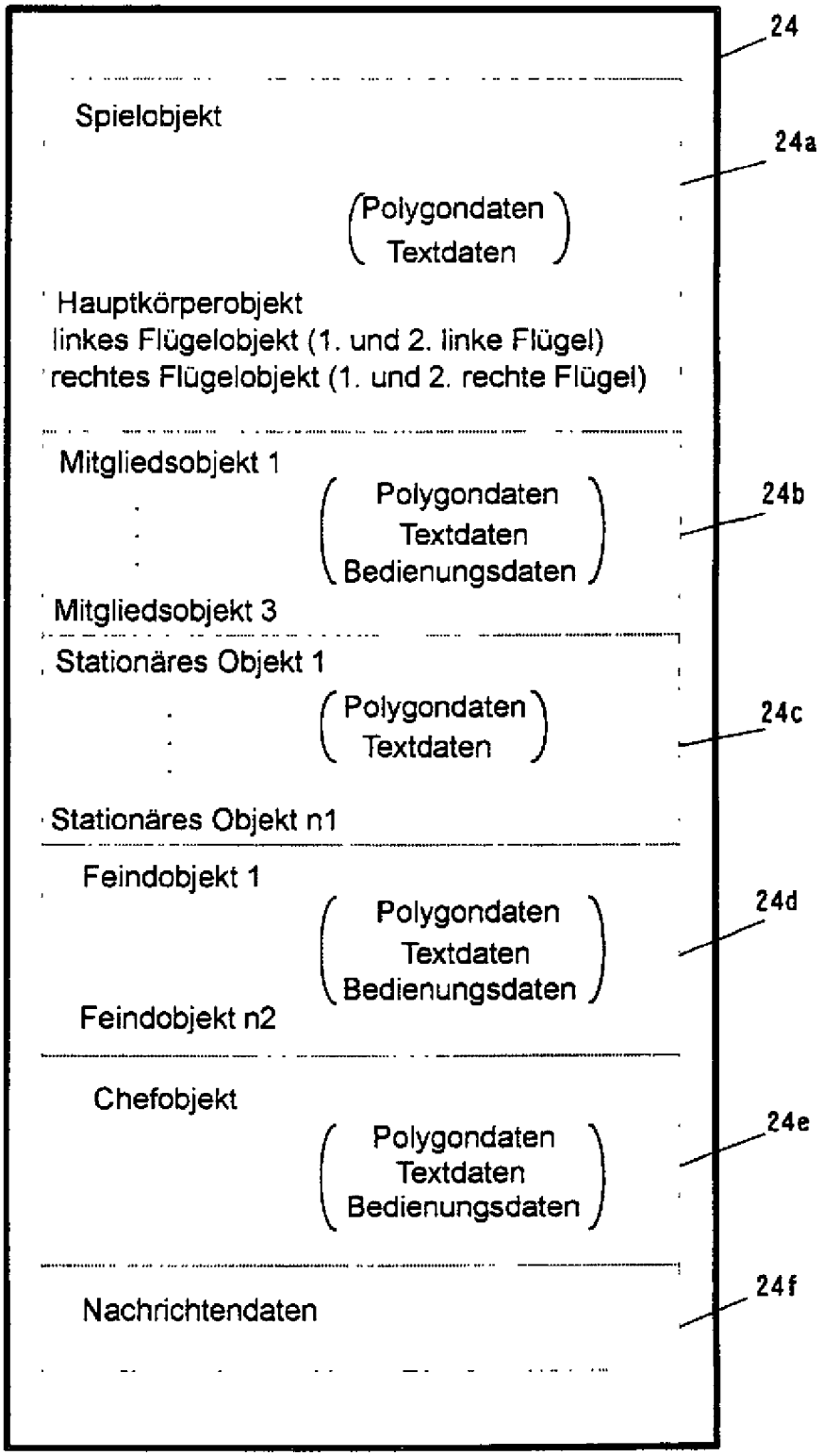


FIG. 7

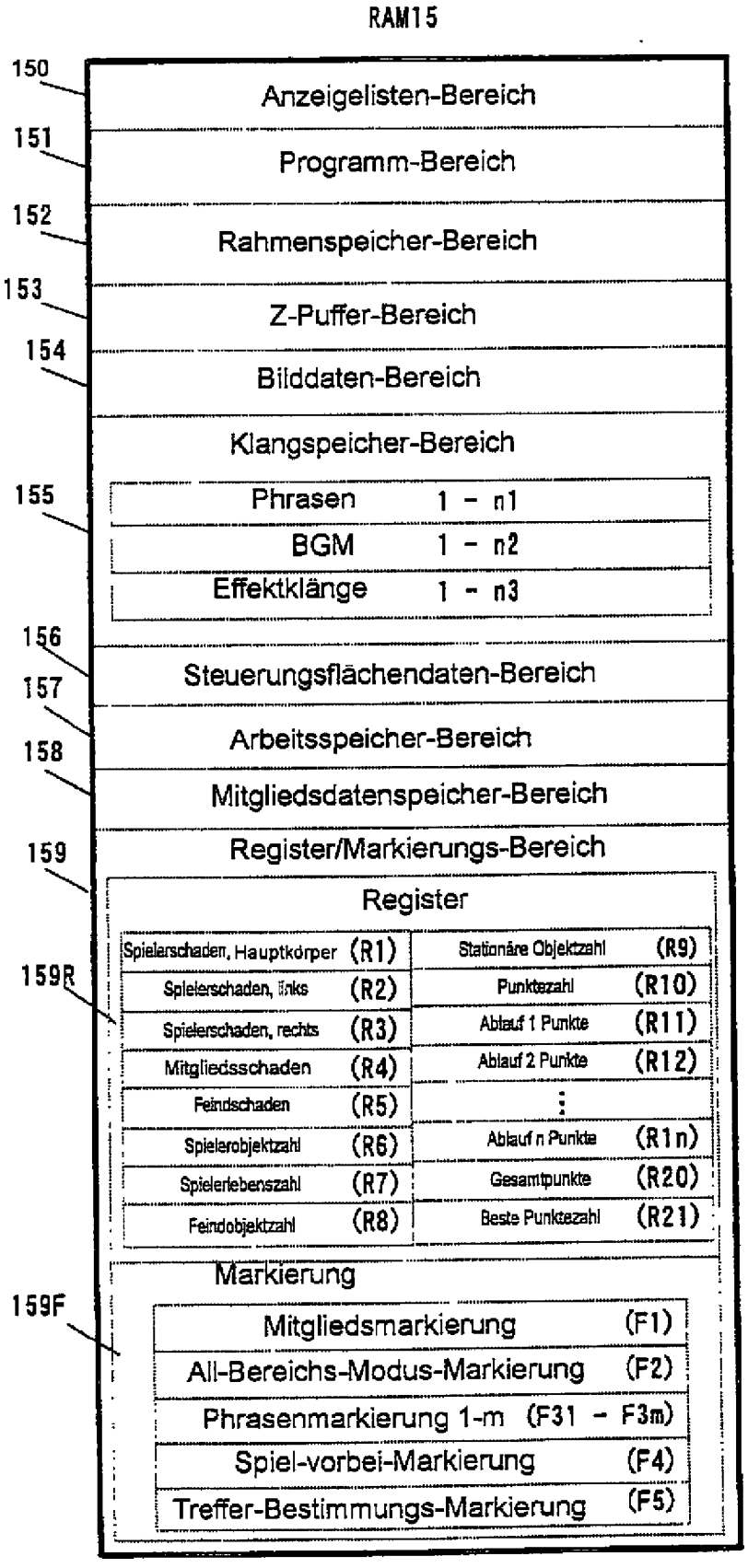

RAM15
150
Anzeigelisten-Bereich
151
Programm-Bereich
152
Rahmenspeicher-Bereich
153
Z-Puffer-Bereich
154
Bilddaten-Bereich
Klangspeicher-Bereich
155
Phrasen 1 - n1
BGM 1 - n2
Effektklänge 1 - n3
156
Steuerungsflächendaten-Bereich
157
Arbeitsspeicher-Bereich
158
Mitgliedsdatenspeicher-Bereich
159
Register/Markierungs-Bereich
Register
159R
Spielerschaden, Hauptkörper (R1)
Spielerschaden, links (R2)
Spielerschaden, rechts (R3)
Mitgliedsschaden (R4)
Feindschaden (R5)
Spielerobjektzahl (R6)
Spielerlebenszahl (R7)
Feindobjektzahl (R8)
Stationäre Objektzahl (R9)
Punktezahl (R10)
Ablauf 1 Punkte (R11)
Ablauf 2 Punkte (R12)
Ablauf n Punkte (R1n)
Gesamtpunkte (R20)
Beste Punktezahl (R21)
Markierung
159F
Mitgliedsmarkierung (F1)
All-Bereichs-Modus-Markierung (F2)
Phrasenmarkierung 1-m (F31 - F3m)
Spiel-vorbei-Markierung (F4)
Treffer-Bestimmungs-Markierung (F5)

FIG. 8 Details des Bilddaten-Bereichs 154

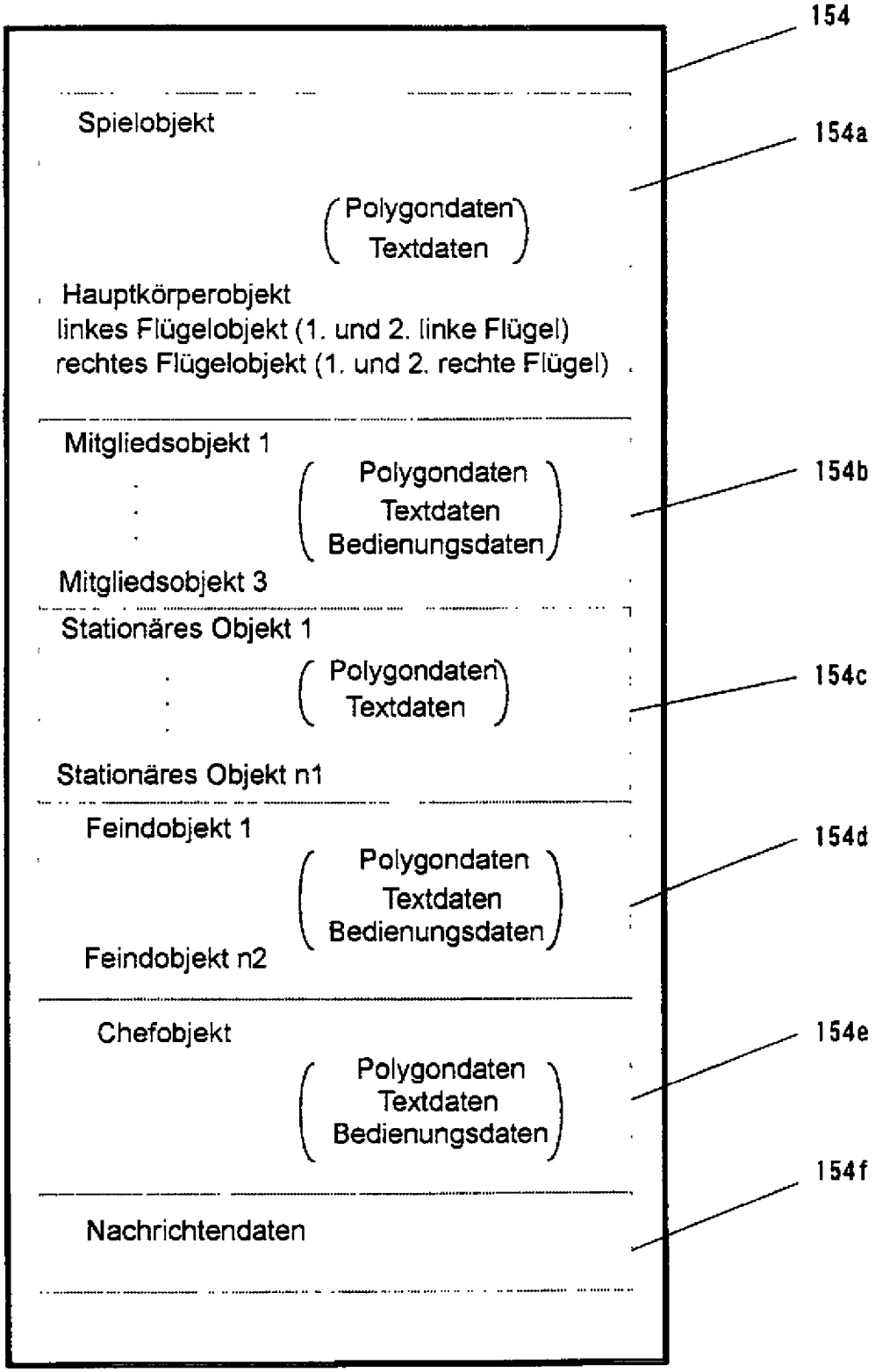


FIG. 9

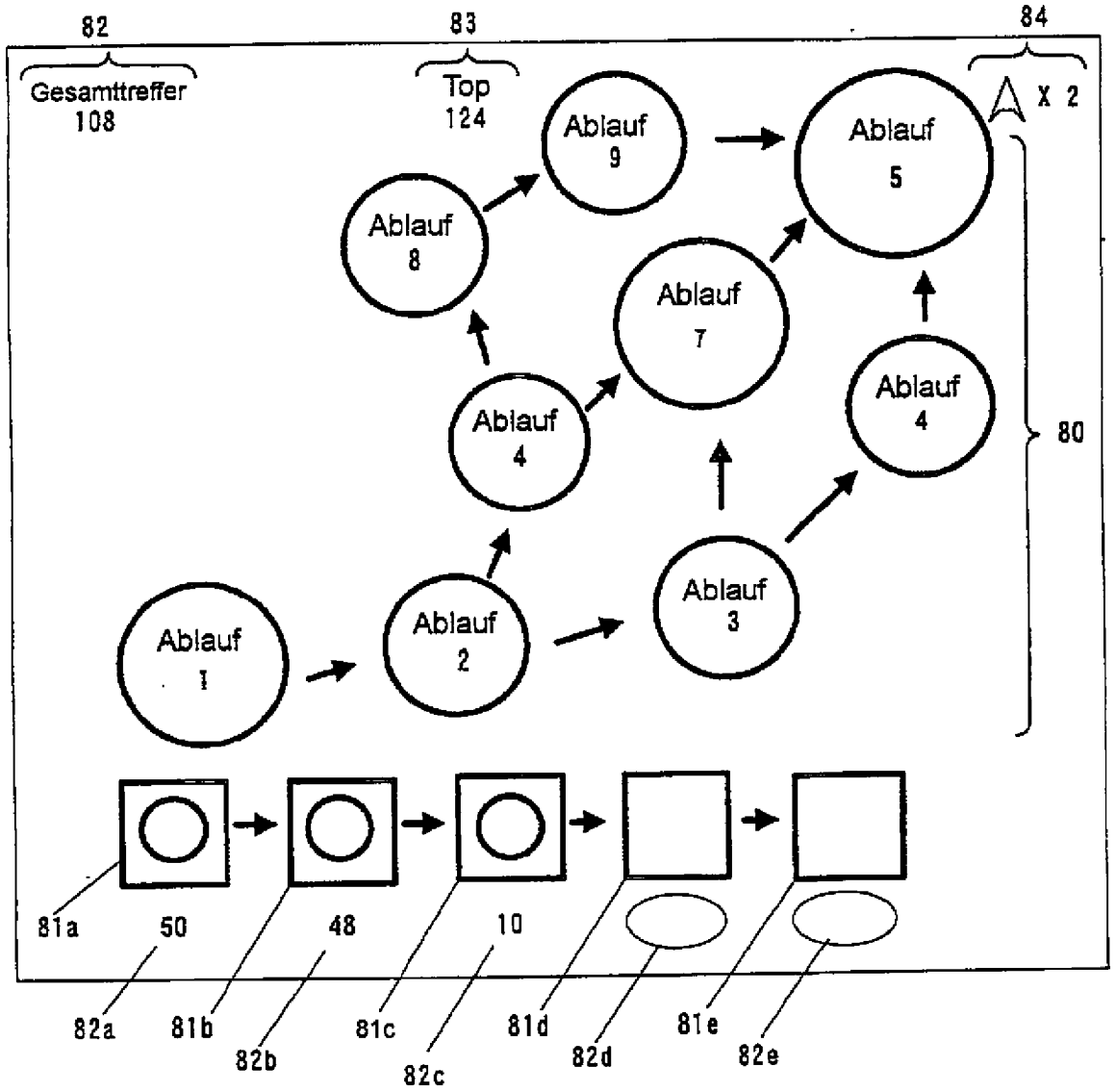

82
Gesamttreffer
108
83
Top
124
84
X 2
Ablauf
1
Ablauf
2
Ablauf
3
Ablauf
4
Ablauf
4
Ablauf
5
Ablauf
7
Ablauf
8
Ablauf
9
80
81a
50
48
10
82a
81b
82b
81c
82c
81d
82d
81e
82e

FIG. 10

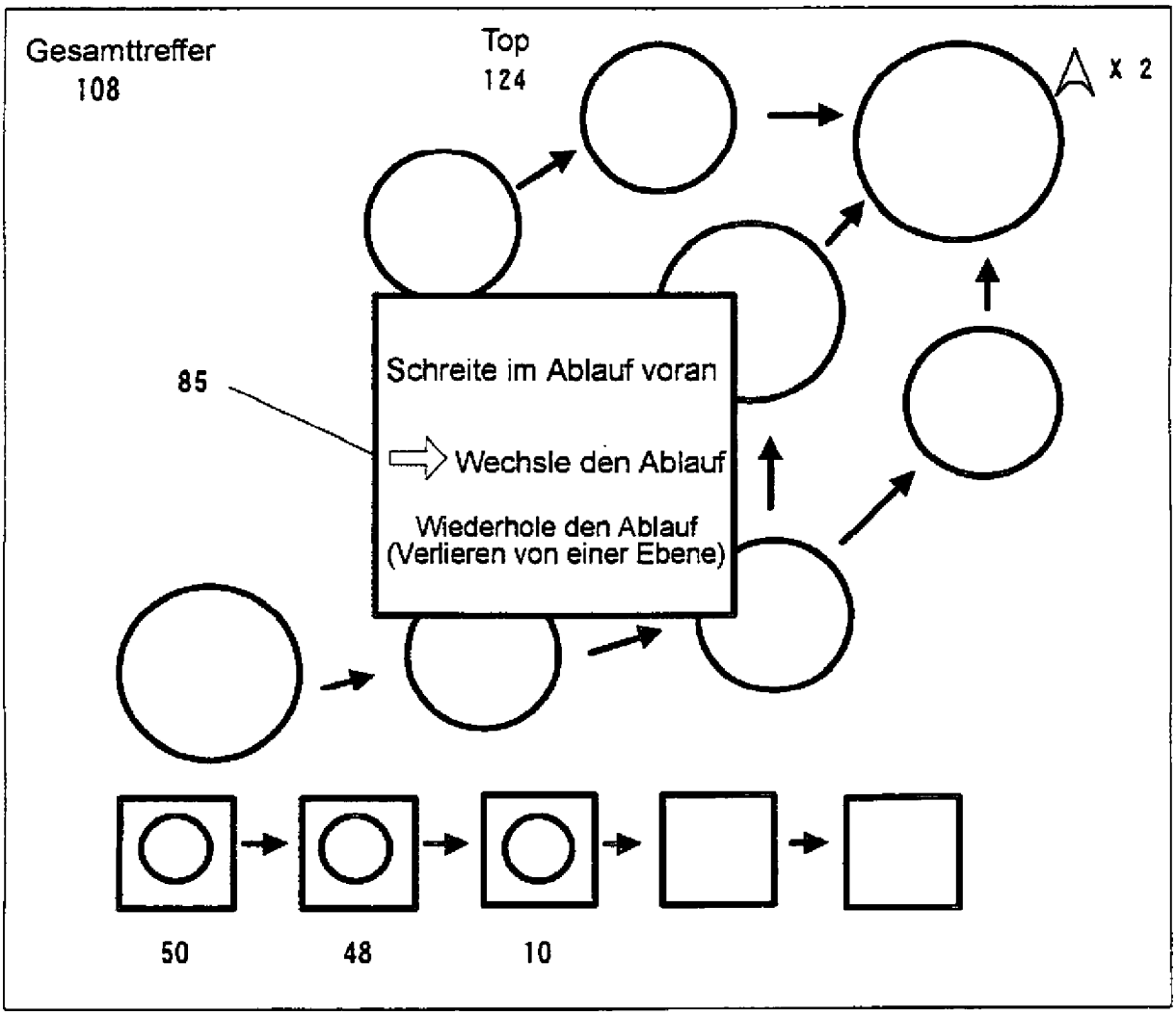

Gesamttreffer
108
Top
124
X 2
85
Schreite im Ablauf voran
Wechsle den Ablauf
Wiederhole den Ablauf
(Verlieren von einer Ebene)
50
48
10

FIG. 11

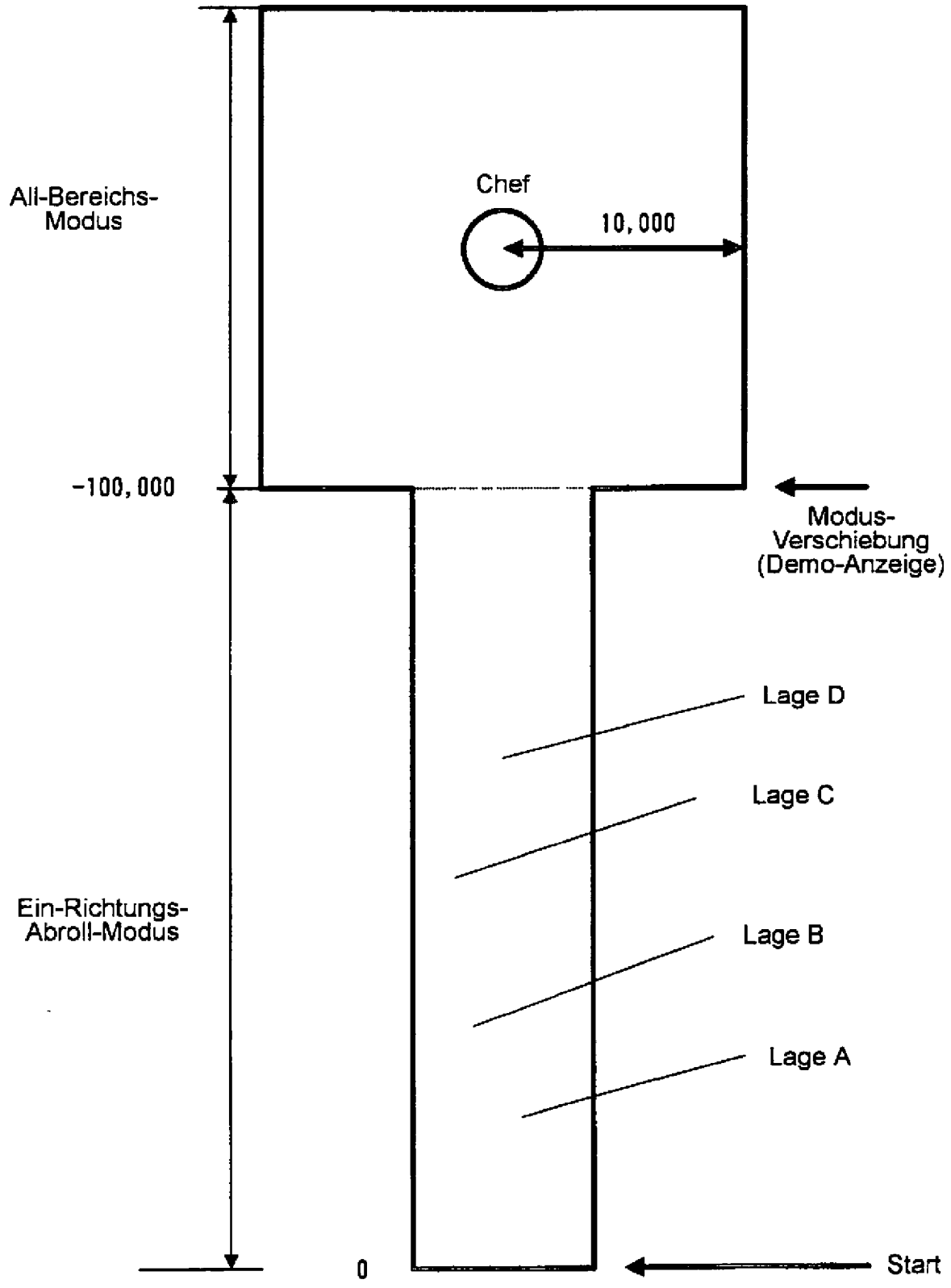

All-Bereichs-
Modus
Chef
10,000
-100,000
Modus-
Verschiebung
(Demo-Anzeige)
Lage D
Lage C
Ein-Richtungs-
Abroll-Modus
Lage B
Lage A
0
Start

FIG. 12

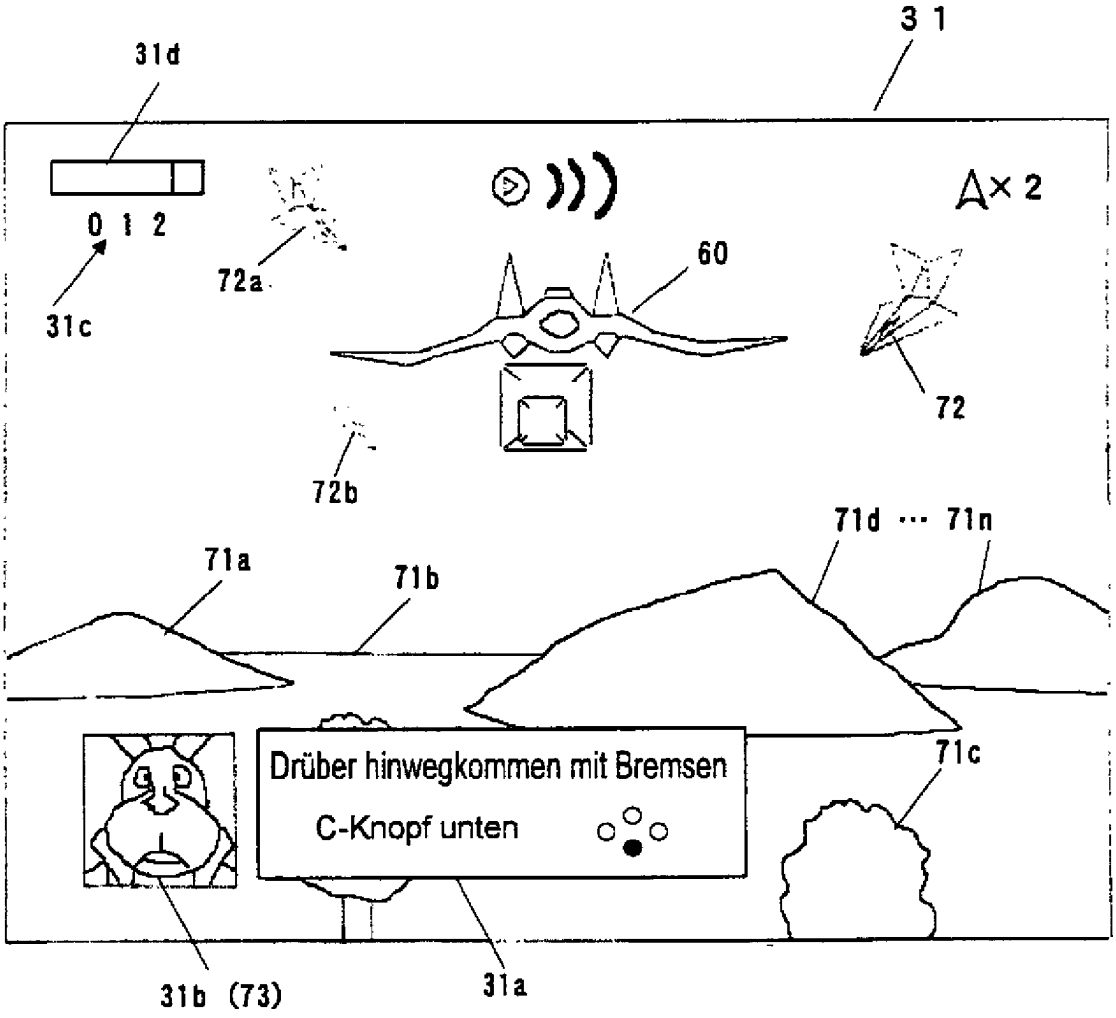
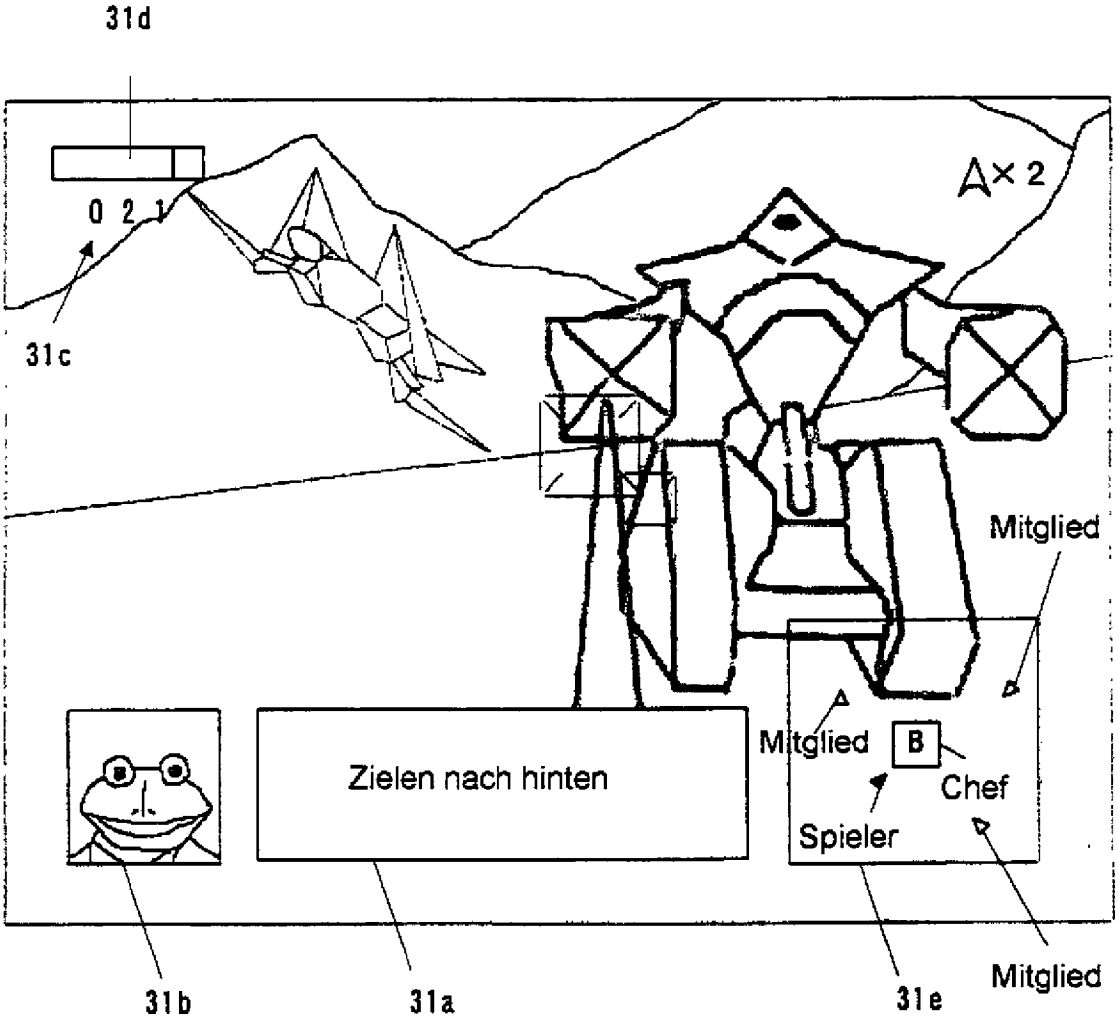

31
31d
0 1 2
31c
72a
60
A× 2
72
72b
71d ··· 71n
71a
71b
71c
Drüber hinwegkommen mit Bremsen
C-Knopf unten
31b (73)
31a

FIG. 13

	Person	Phrase	Zustand	Priorität
Phrase 1	Mitglied 1	Drüber hinwegkommen mit Bremsen, C-Knopf unten ↓	Lage A	1
Phrase 2	Mitglied 2	Drüber hinwegkommen durch Verstärkung, C-Knopf links ←	Lage B	2
Phrase 3	Mitglied 3	Nach hinten zielen	Zeit A von Finden des Chefs ab	2
Phrase 4	Mitglied 4	Abwehren durch zweimal Rollen, Drücke Z oder R	Lage C	1
Phrase 5	Mitglied 5	Hilfe	Angezielt durch Feind	1
Phrase 6	Mitglied 6	Überlebt	Retten	1
Phrase 7	Mitglied 3	Taktvoll Bomben benutzen	Lage D	1
Phrase 8	Held	UWAHHH.....	Bd ≦ 0	3
Phrase 9	Chef	GWAHHH....	Des Feindes Ed ≦ 0	0

FIG. 14


31d
0 2 1
31c
A×2
Mitglied
Mitglied
B
Chef
Spieler
Zielen nach hinten
Mitglied
31b
31a
31e

Gesamtes Flußdiagramm

FIG. 15

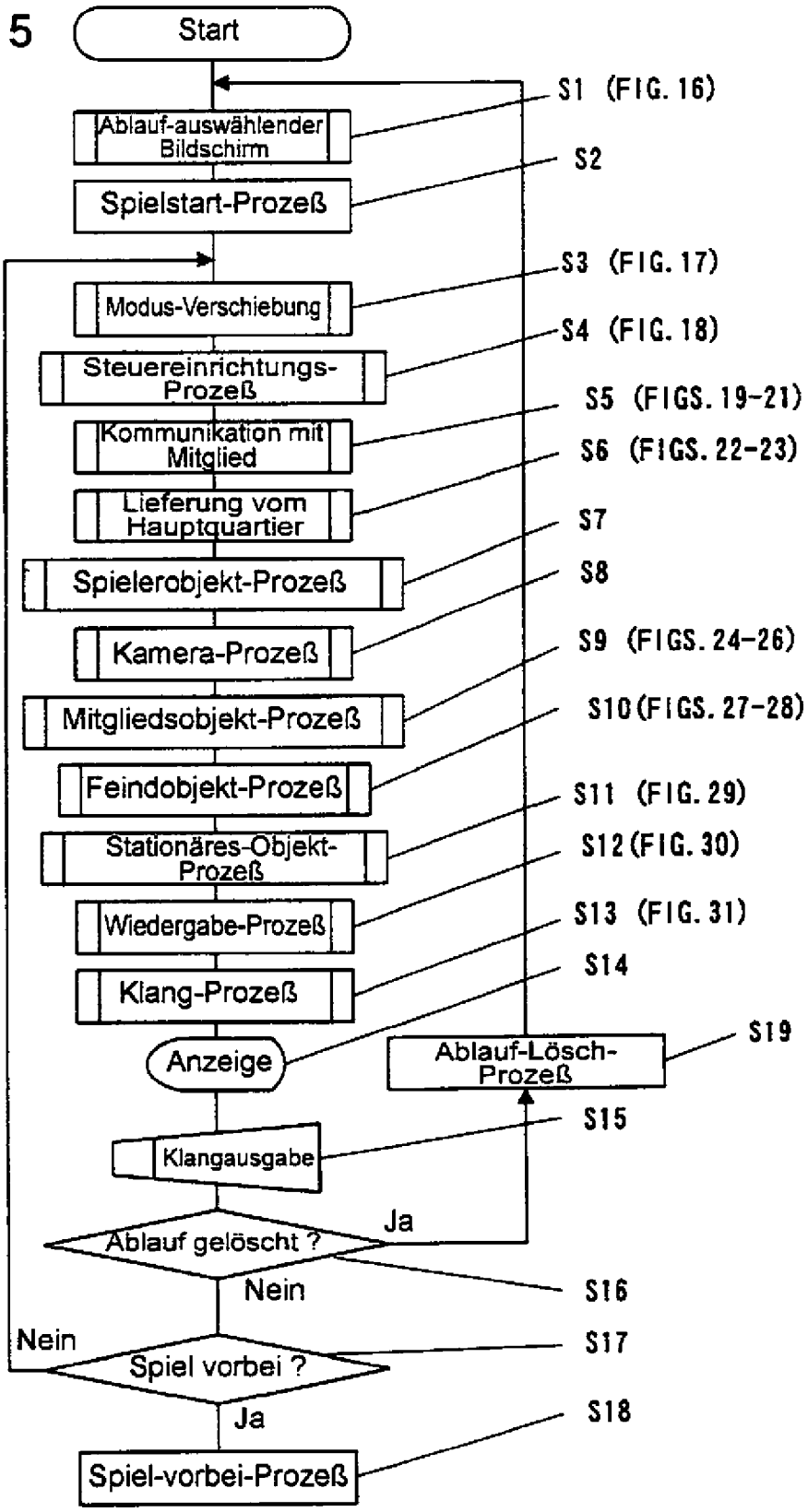


FIG. 16

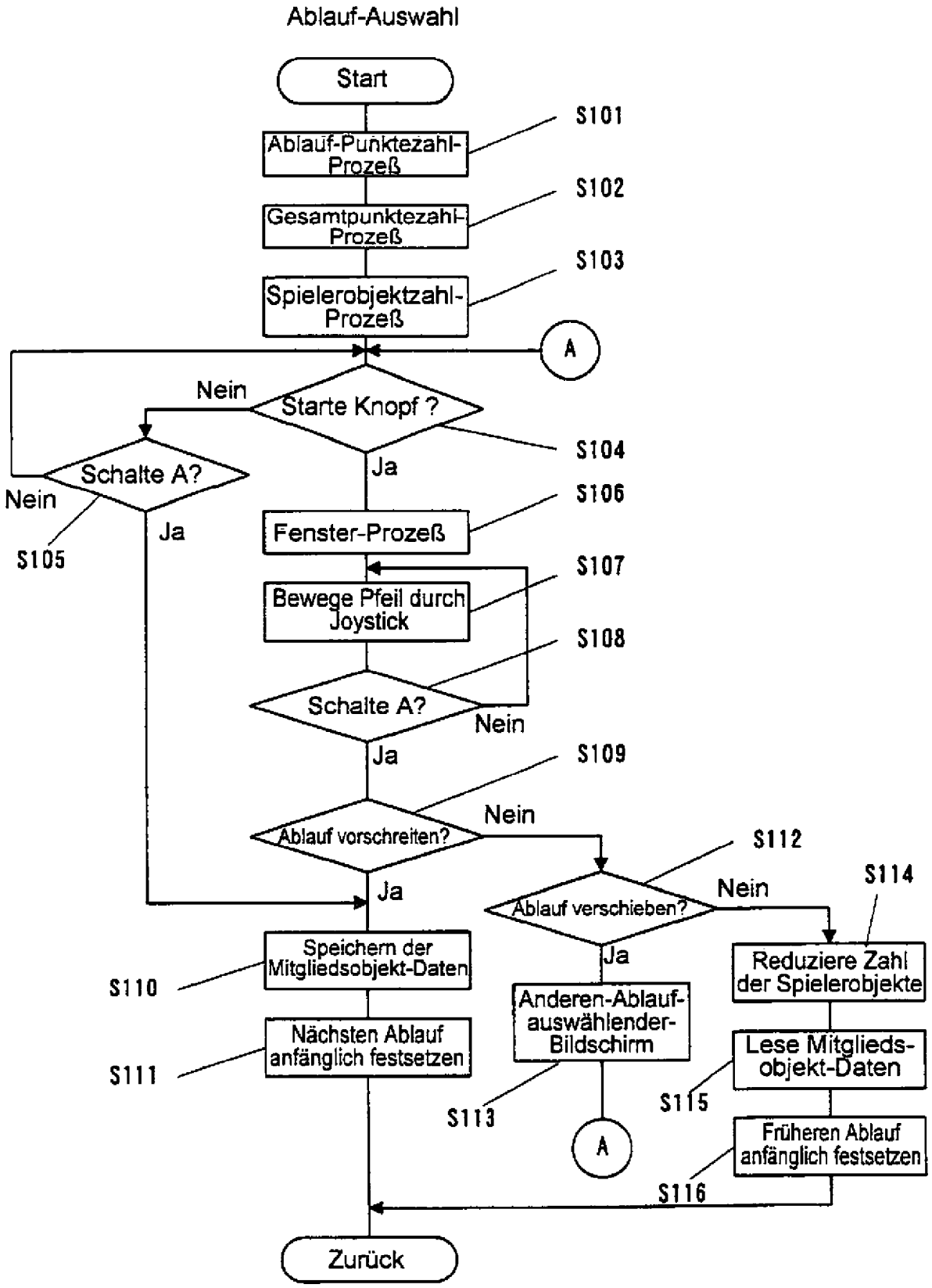

Ablauf-Auswahl
Start
S101
Ablauf-Punktezahl-Prozeß
S102
Gesamtpunktezahl-Prozeß
S103
Spielerobjektzahl-Prozeß
A
Nein
Starte Knopf ?
S104
Ja
Schalte A?
Nein
S105
Ja
S106
Fenster-Prozeß
S107
Bewege Pfeil durch Joystick
S108
Schalte A?
Nein
Ja
S109
Ablauf vorschreiten?
Nein
Ja
S112
Ablauf verschieben?
Nein
S114
Ja
Speichern der Mitgliedsobjekt-Daten
S110
Nächsten Ablauf anfänglich festsetzen
S111
Anderen-Ablauf-auswählender-Bildschirm
S113
A
Reduziere Zahl der Spielerobjekte
Lese Mitglieds-objekt-Daten
S115
Früheren Ablauf anfänglich festsetzen
S116
Zurück

FIG. 17

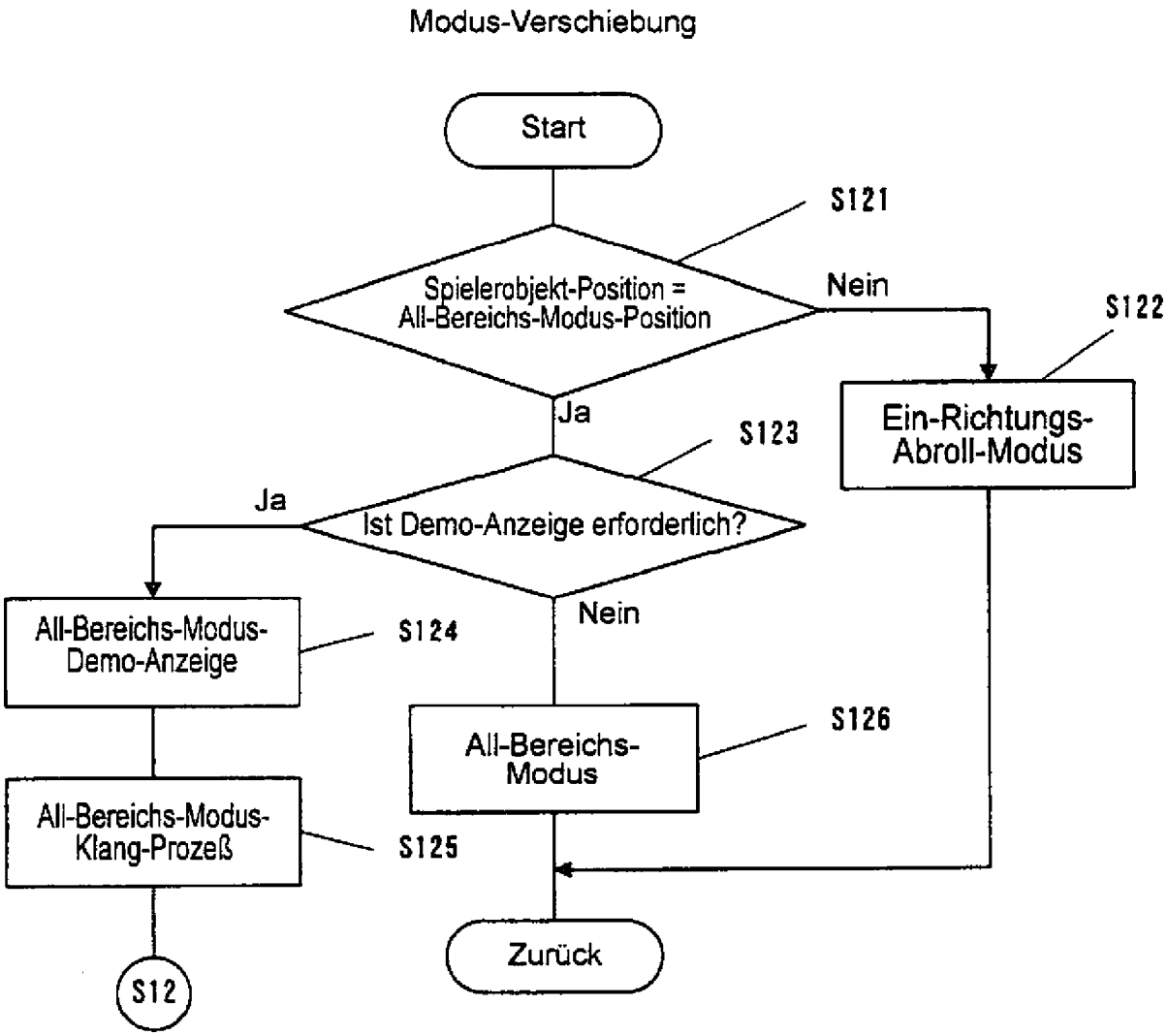

Modus-Verschiebung
Start
S121
Spielerobjekt-Position = All-Bereichs-Modus-Position
Nein
S122
Ein-Richtungs-Abroll-Modus
Ja
S123
Ist Demo-Anzeige erforderlich?
Ja
Nein
All-Bereichs-Modus-Demo-Anzeige
S124
All-Bereichs-Modus
S126
All-Bereichs-Modus-Klang-Prozeß
S125
Zurück
S12

FIG. 18

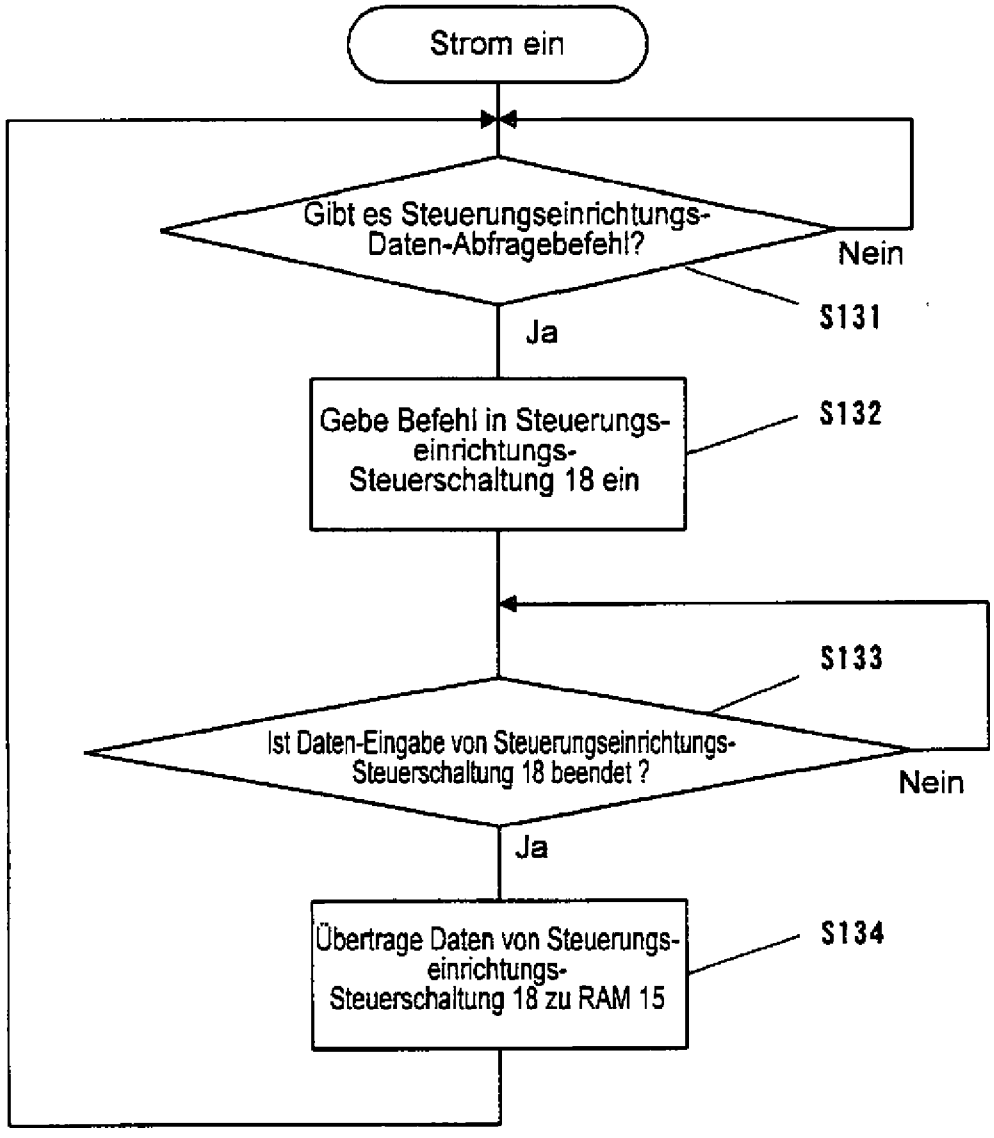

Strom ein
Gibt es Steuerungseinrichtungs-Daten-Abfragebefehl?
Nein
S131
Ja
Gebe Befehl in Steuerungseinrichtungs-Steuerschaltung 18 ein
S132
S133
Ist Daten-Eingabe von Steuerungseinrichtungs-Steuerschaltung 18 beendet ?
Nein
Ja
Übertrage Daten von Steuerungseinrichtungs-Steuerschaltung 18 zu RAM 15
S134

FIG. 19

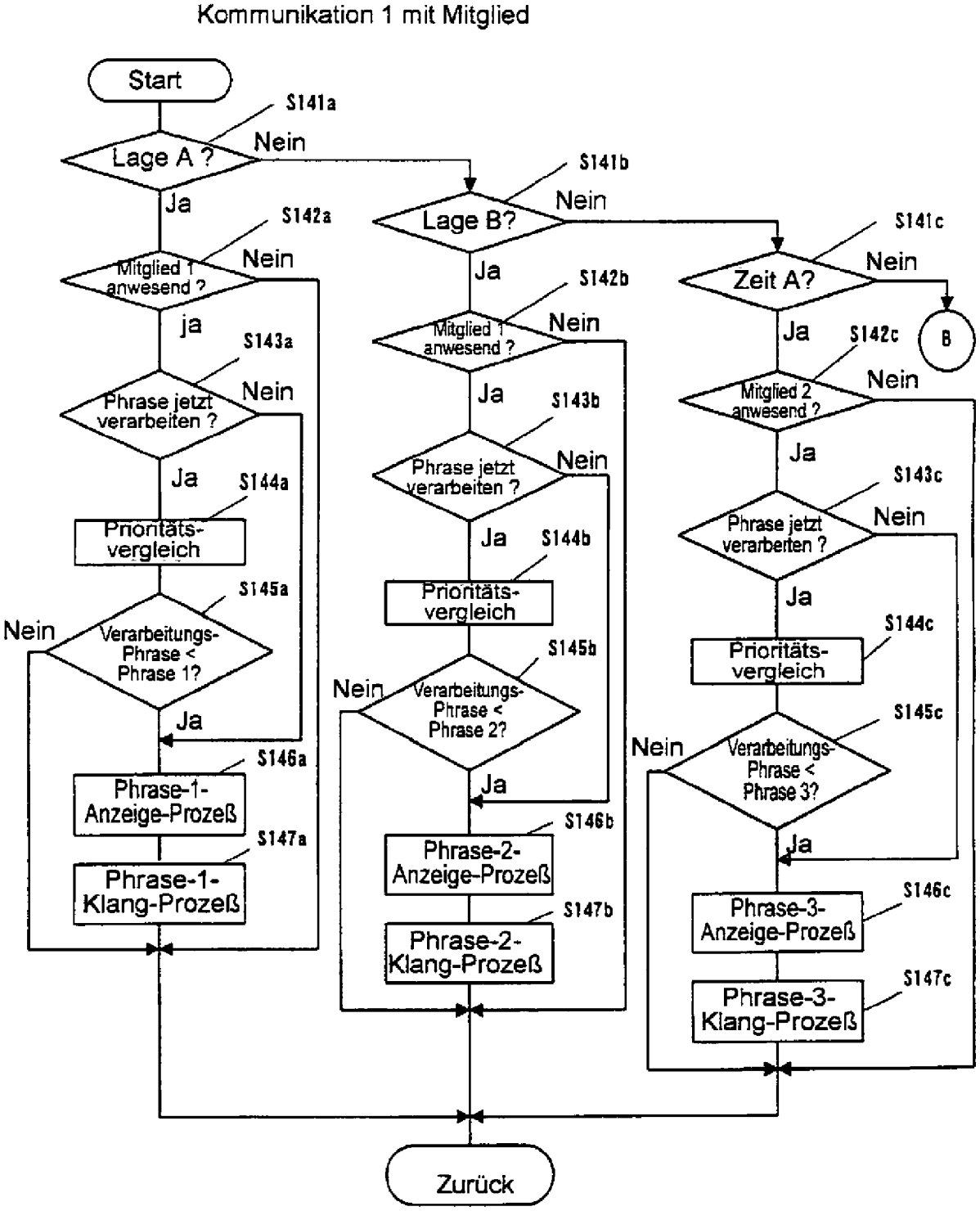

Kommunikation 1 mit Mitglied
Start
S141a
Lage A ?
Nein
Ja
S142a
Mitglied 1 anwesend ?
Nein
ja
S143a
Phrase jetzt verarbeiten ?
Nein
Ja
S144a
Prioritäts-vergleich
S145a
Nein
Verarbeitungs-Phrase < Phrase 1?
Ja
S146a
Phrase-1-Anzeige-Prozeß
S147a
Phrase-1-Klang-Prozeß
S141b
Lage B?
Nein
Ja
S142b
Mitglied 1 anwesend ?
Nein
Ja
S143b
Phrase jetzt verarbeiten ?
Nein
Ja
S144b
Prioritäts-vergleich
S145b
Nein
Verarbeitungs-Phrase < Phrase 2?
Ja
S146b
Phrase-2-Anzeige-Prozeß
S147b
Phrase-2-Klang-Prozeß
S141c
Zeit A?
Nein
Ja
B
S142c
Mitglied 2 anwesend ?
Nein
Ja
S143c
Phrase jetzt verarbeiten ?
Nein
Ja
S144c
Prioritäts-vergleich
S145c
Nein
Verarbeitungs-Phrase < Phrase 3?
Ja
S146c
Phrase-3-Anzeige-Prozeß
S147c
Phrase-3-Klang-Prozeß
Zurück

FIG. 20

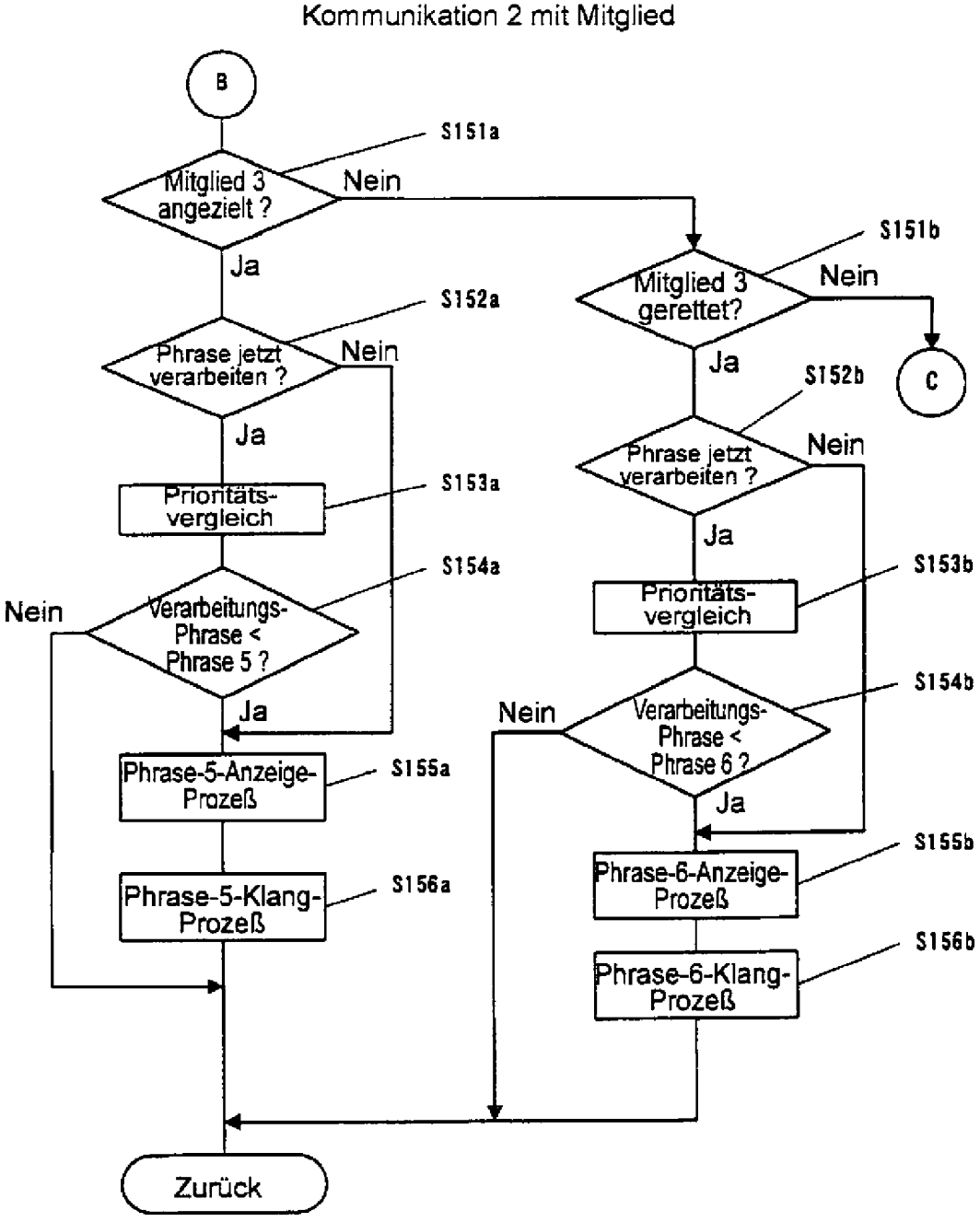

Kommunikation 2 mit Mitglied
B
S151a
Mitglied 3 angezielt ?
Nein
Ja
S152a
Phrase jetzt verarbeiten ?
Nein
Ja
Prioritäts-vergleich
S153a
S154a
Nein
Verarbeitungs-Phrase < Phrase 5 ?
Ja
Phrase-5-Anzeige-Prozeß
S155a
Phrase-5-Klang-Prozeß
S156a
S151b
Mitglied 3 gerettet?
Nein
C
Ja
S152b
Phrase jetzt verarbeiten ?
Nein
Ja
Prioritäts-vergleich
S153b
S154b
Nein
Verarbeitungs-Phrase < Phrase 6 ?
Ja
S155b
Phrase-6-Anzeige-Prozeß
S156b
Phrase-6-Klang-Prozeß
Zurück

FIG. 21

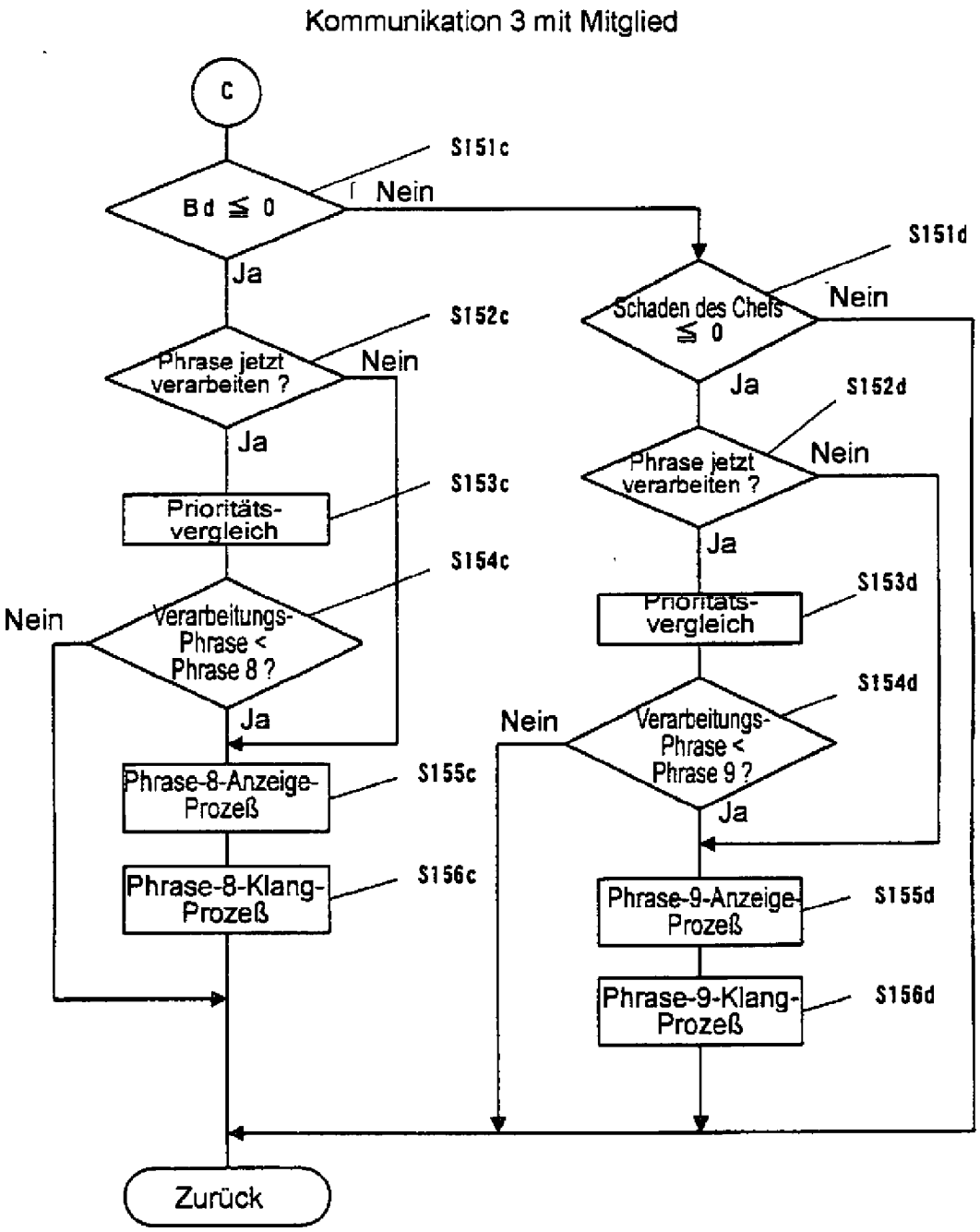

Kommunikation 3 mit Mitglied
C
S151c
Bd ≦ 0
Nein
Ja
S151d
Schaden des Chefs ≦ 0
Nein
Ja
S152c
Phrase jetzt verarbeiten ?
Nein
Ja
S152d
Phrase jetzt verarbeiten ?
Nein
Ja
S153c
Prioritäts-vergleich
S154c
Nein
Verarbeitungs-Phrase < Phrase 8 ?
Ja
S153d
Prioritäts-vergleich
S154d
Nein
Verarbeitungs-Phrase < Phrase 9 ?
Ja
S155c
Phrase-8-Anzeige-Prozeß
S156c
Phrase-8-Klang-Prozeß
S155d
Phrase-9-Anzeige-Prozeß
S156d
Phrase-9-Klang-Prozeß
Zurück

FIG. 22

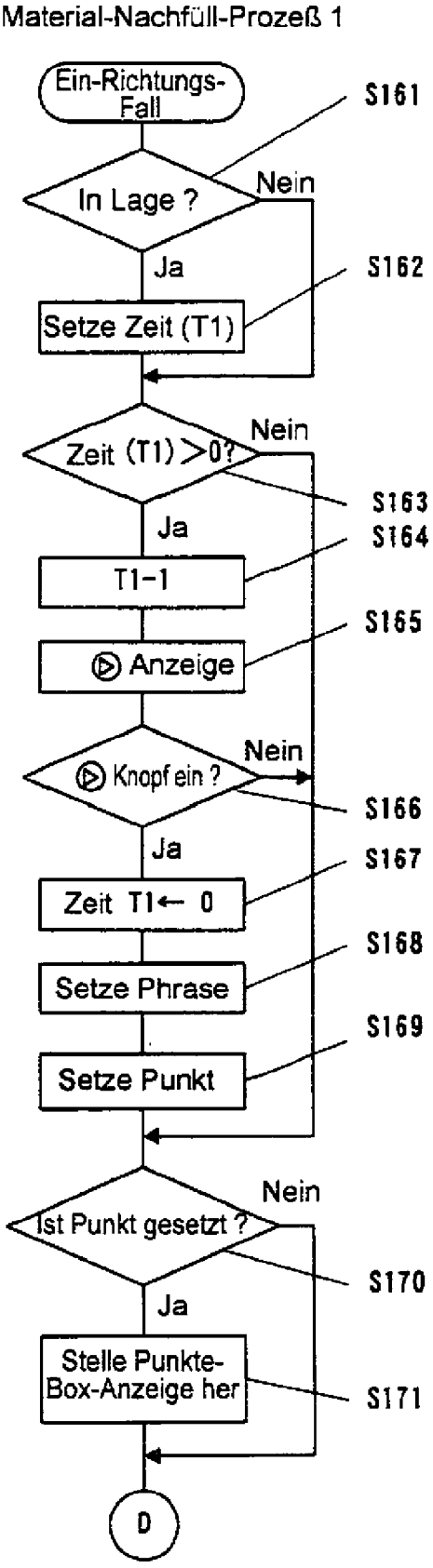

Material-Nachfüll-Prozeß 1
Ein-Richtungs-Fall
S161
In Lage ?
Nein
Ja
S162
Setze Zeit (T1)
Zeit (T1) > 0?
Nein
S163
Ja
S164
T1-1
S165
Anzeige
Knopf ein ?
Nein
S166
Ja
S167
Zeit T1← 0
S168
Setze Phrase
S169
Setze Punkt
Ist Punkt gesetzt ?
Nein
S170
Ja
Stelle Punkte-Box-Anzeige her
S171
D

FIG. 23

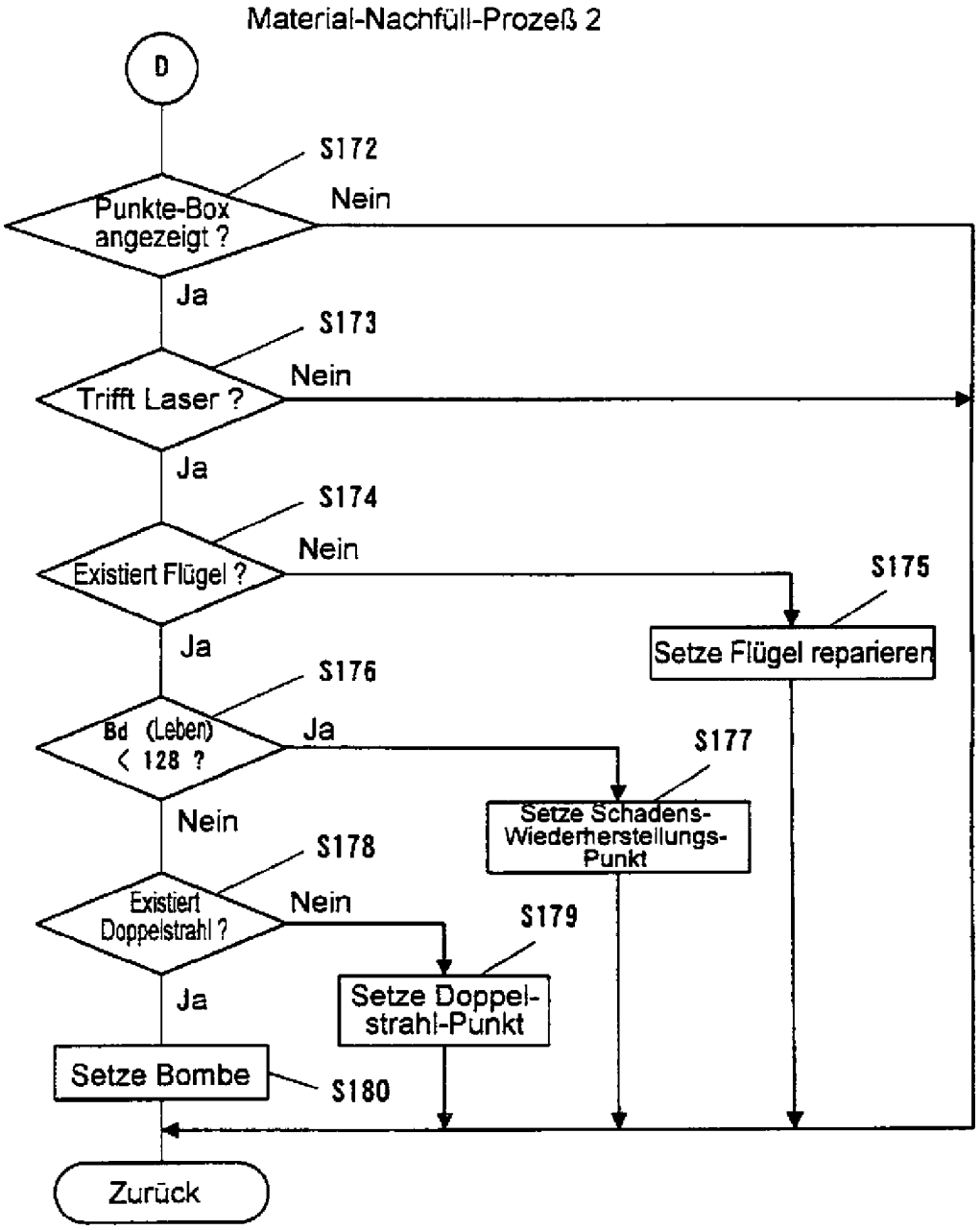

Material-Nachfüll-Prozeß 2
D
S172
Punkte-Box angezeigt ?
Nein
Ja
S173
Trifft Laser ?
Nein
Ja
S174
Existiert Flügel ?
Nein
S175
Setze Flügel reparieren
Ja
S176
Bd (Leben) < 128 ?
Ja
S177
Setze Schadens-Wiederherstellungs-Punkt
Nein
S178
Existiert Doppelstrahl ?
Nein
S179
Setze Doppel-strahl-Punkt
Ja
Setze Bombe
S180
Zurück

FIG. 24

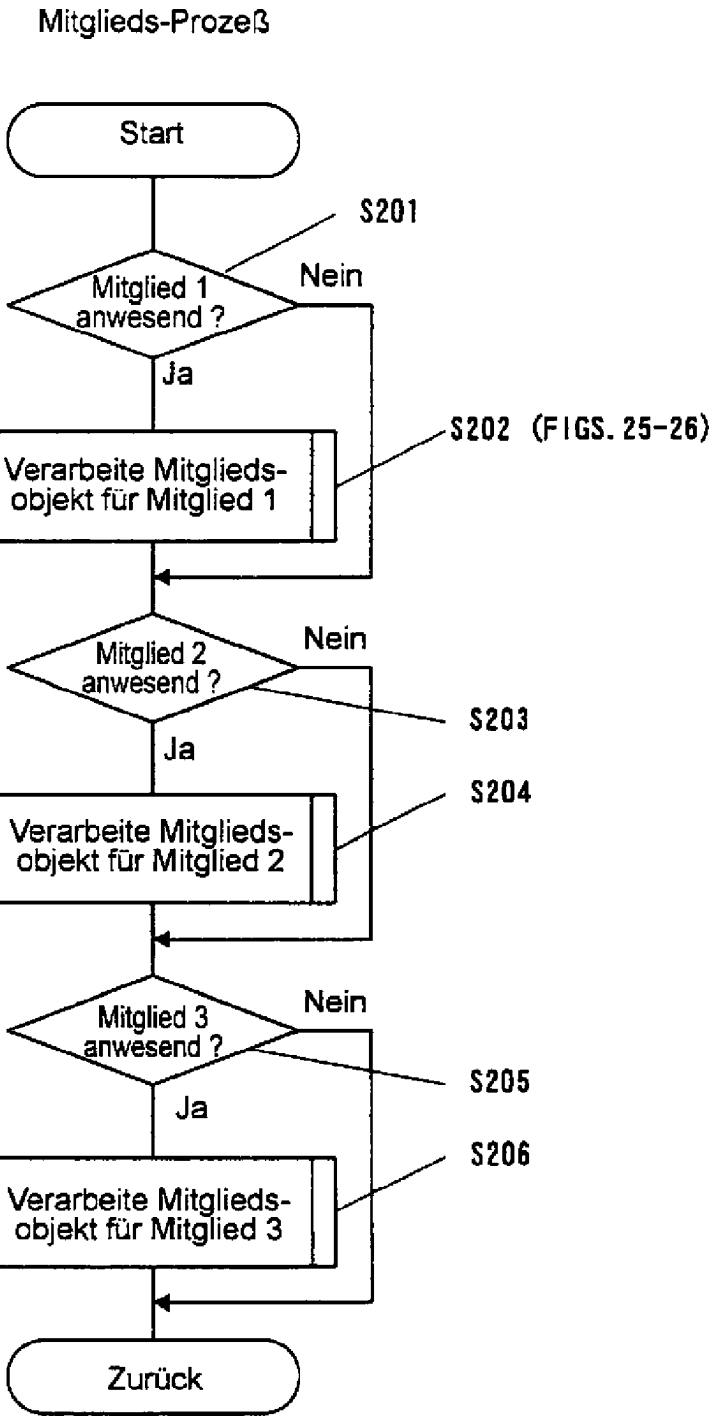

Mitglieds-Prozeß
Start
S201
Mitglied 1 anwesend ?
Nein
Ja
S202 (FIGS. 25-26)
Verarbeite Mitglieds-objekt für Mitglied 1
Mitglied 2 anwesend ?
Nein
S203
Ja
S204
Verarbeite Mitglieds-objekt für Mitglied 2
Mitglied 3 anwesend ?
Nein
S205
Ja
S206
Verarbeite Mitglieds-objekt für Mitglied 3
Zurück

FIG. 25

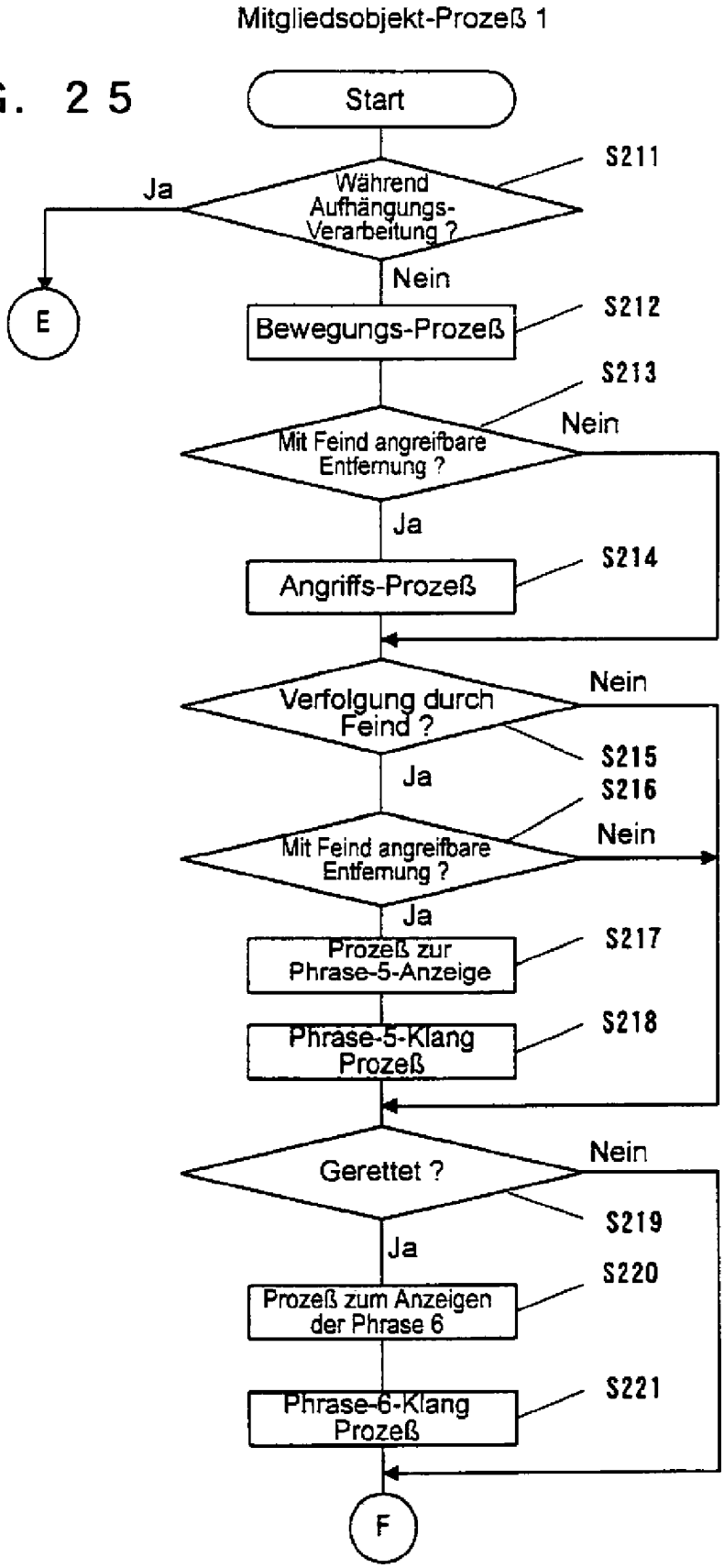

Mitgliedsobjekt-Prozeß 1
Start
S211
Während Aufhängungs-Verarbeitung ?
Ja
Nein
E
S212
Bewegungs-Prozeß
S213
Mit Feind angreifbare Entfernung ?
Nein
Ja
S214
Angriffs-Prozeß
Verfolgung durch Feind ?
Nein
S215
Ja
S216
Mit Feind angreifbare Entfernung ?
Nein
Ja
S217
Prozeß zur Phrase-5-Anzeige
S218
Phrase-5-Klang Prozeß
Gerettet ?
Nein
S219
Ja
S220
Prozeß zum Anzeigen der Phrase 6
S221
Phrase-6-Klang Prozeß
F

FIG. 26

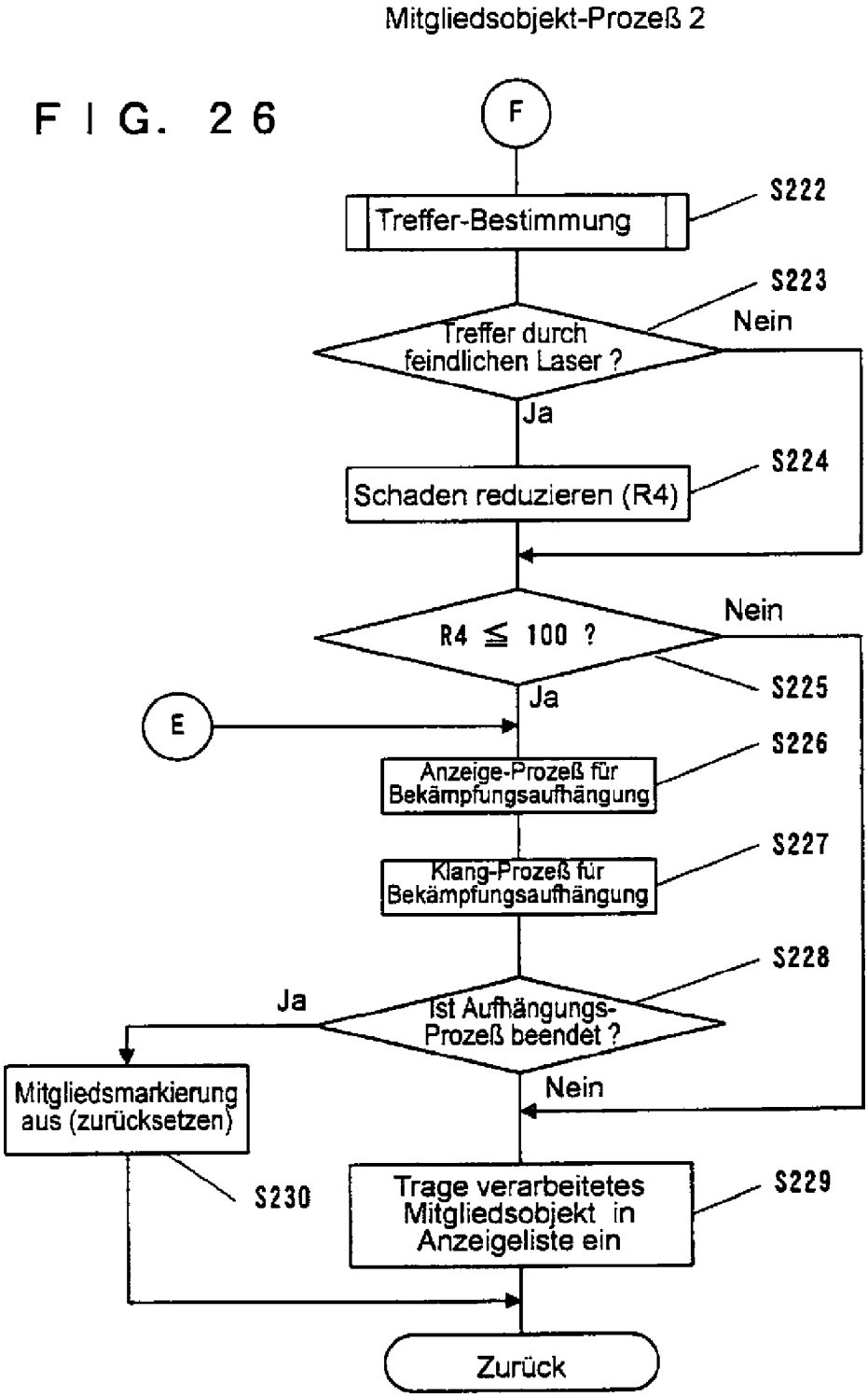

Mitgliedsobjekt-Prozeß 2
F
Treffer-Bestimmung
S222
S223
Treffer durch feindlichen Laser ?
Nein
Ja
Schaden reduzieren (R4)
S224
R4 ≦ 100 ?
Nein
Ja
S225
E
Anzeige-Prozeß für Bekämpfungsaufhängung
S226
Klang-Prozeß für Bekämpfungsaufhängung
S227
Ist Aufhängungs-Prozeß beendet ?
S228
Ja
Nein
Mitgliedsmarkierung aus (zurücksetzen)
S230
Trage verarbeitetes Mitgliedsobjekt in Anzeigeliste ein
S229
Zurück

FIG. 27

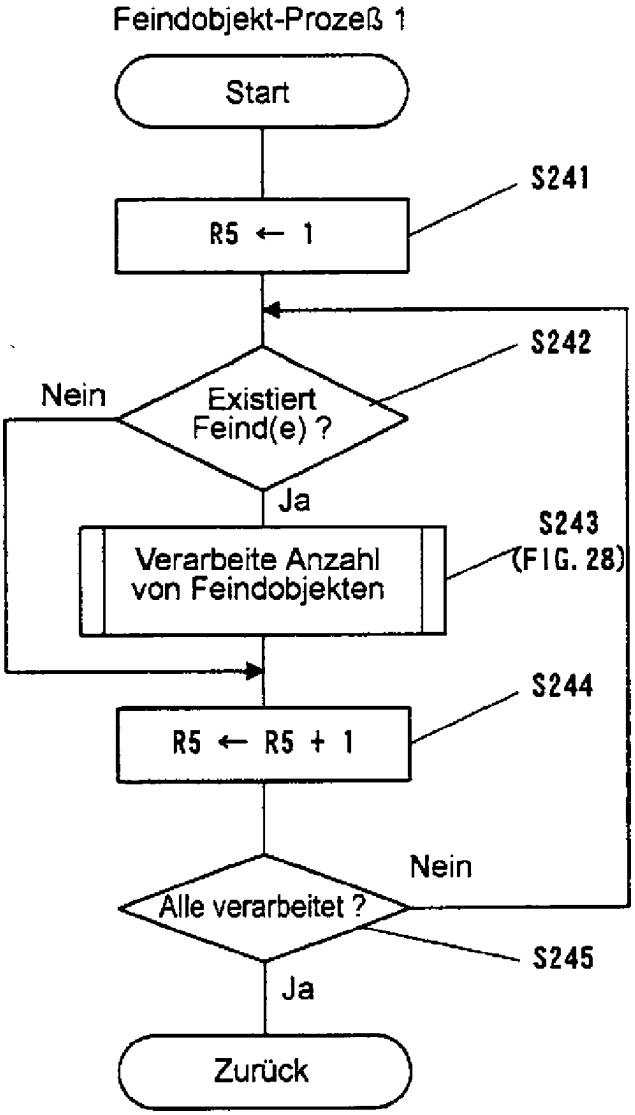

Feindobjekt-Prozeß 1
Start
S241
R5 ← 1
S242
Existiert Feind(e) ?
Nein
Ja
S243
(FIG. 28)
Verarbeite Anzahl von Feindobjekten
S244
R5 ← R5 + 1
Nein
Alle verarbeitet ?
S245
Ja
Zurück

FIG. 28

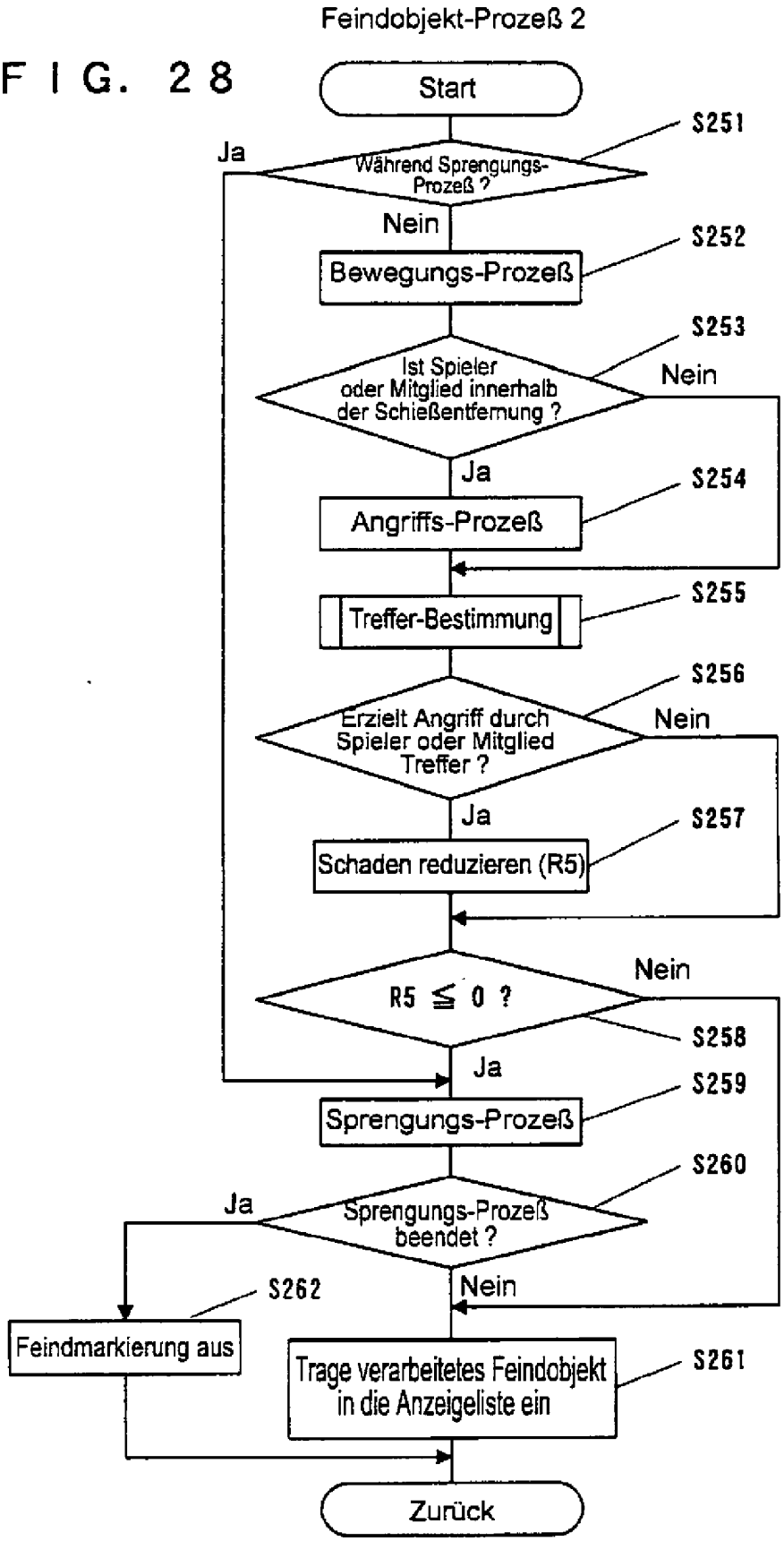

Feindobjekt-Prozeß 2
Start
S251
Während Sprengungs-Prozeß ?
Ja
Nein
S252
Bewegungs-Prozeß
S253
Ist Spieler oder Mitglied innerhalb der Schießentfernung ?
Nein
Ja
S254
Angriffs-Prozeß
S255
Treffer-Bestimmung
S256
Erzielt Angriff durch Spieler oder Mitglied Treffer ?
Nein
Ja
S257
Schaden reduzieren (R5)
Nein
R5 ≦ 0 ?
S258
Ja
S259
Sprengungs-Prozeß
S260
Sprengungs-Prozeß beendet ?
Ja
Nein
S262
Feindmarkierung aus
Trage verarbeitetes Feindobjekt in die Anzeigeliste ein
S261
Zurück

FIG. 29

Stationäres-Objekt-Prozeß

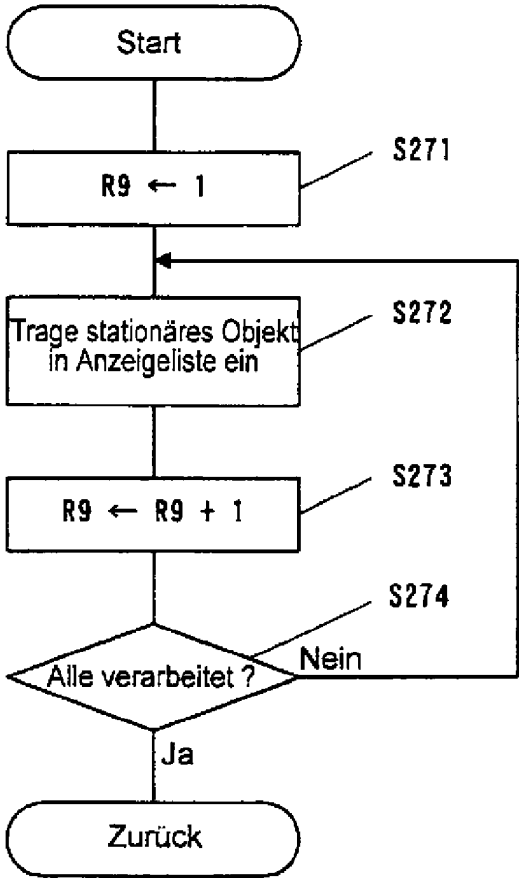


FIG. 30

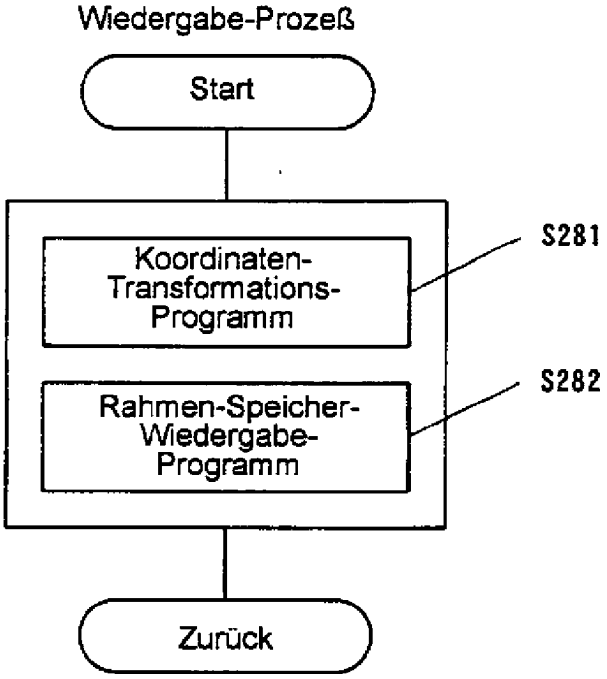
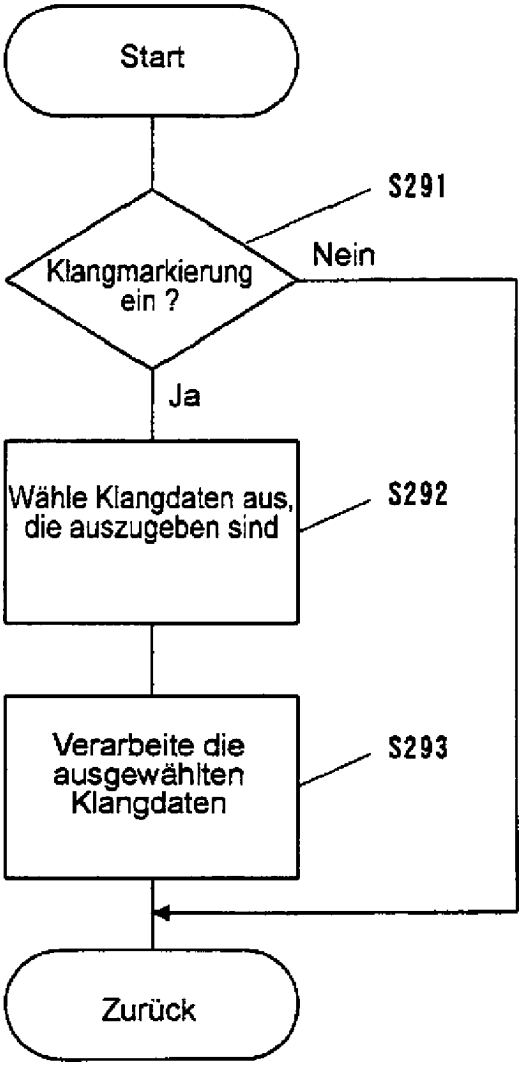

Wiedergabe-Prozeß
Start
Koordinaten-Transformations-Programm
S281
Rahmen-Speicher-Wiedergabe-Programm
S282
Zurück

FIG. 31

Klangausgabe-Prozeß



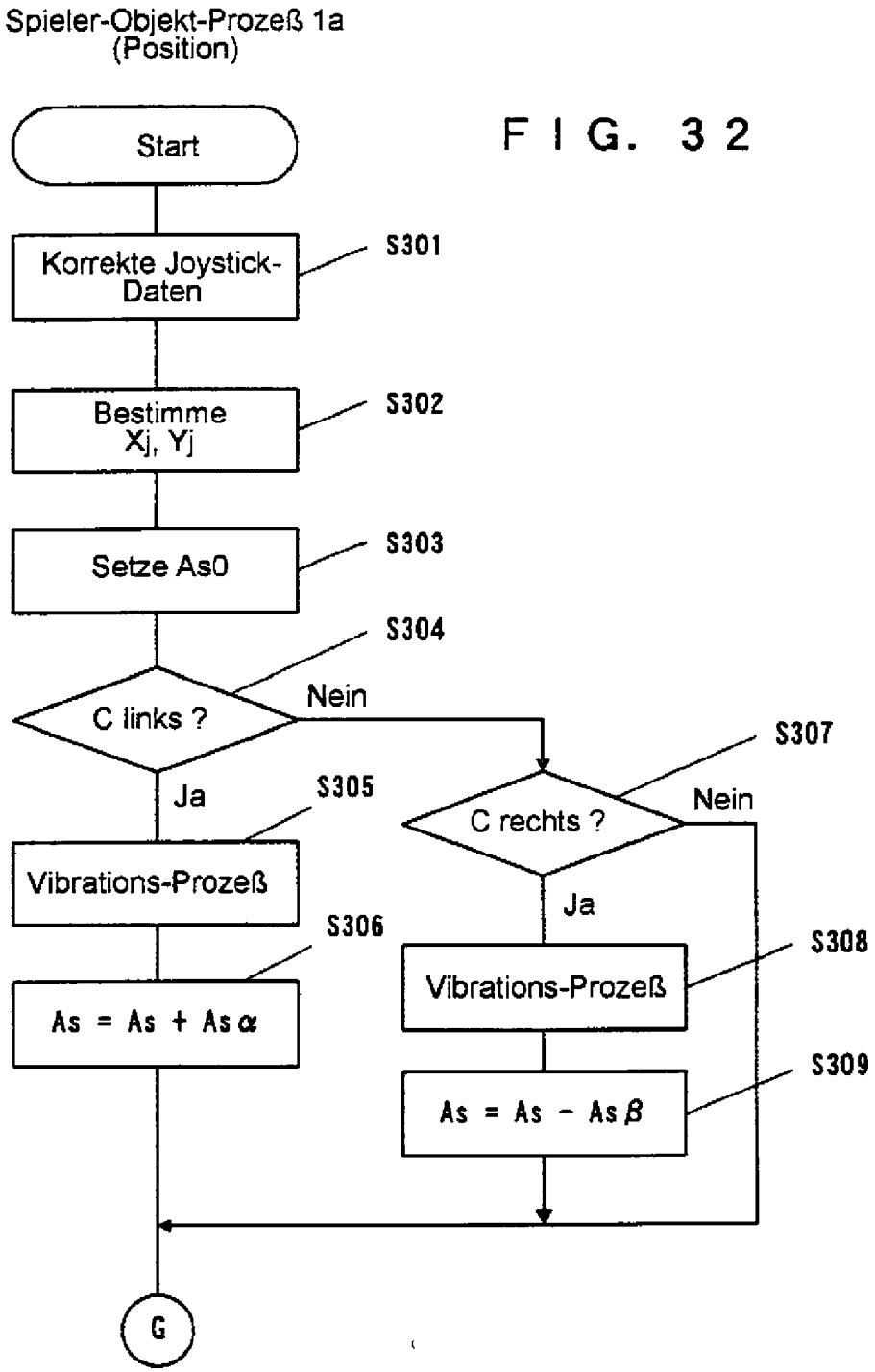
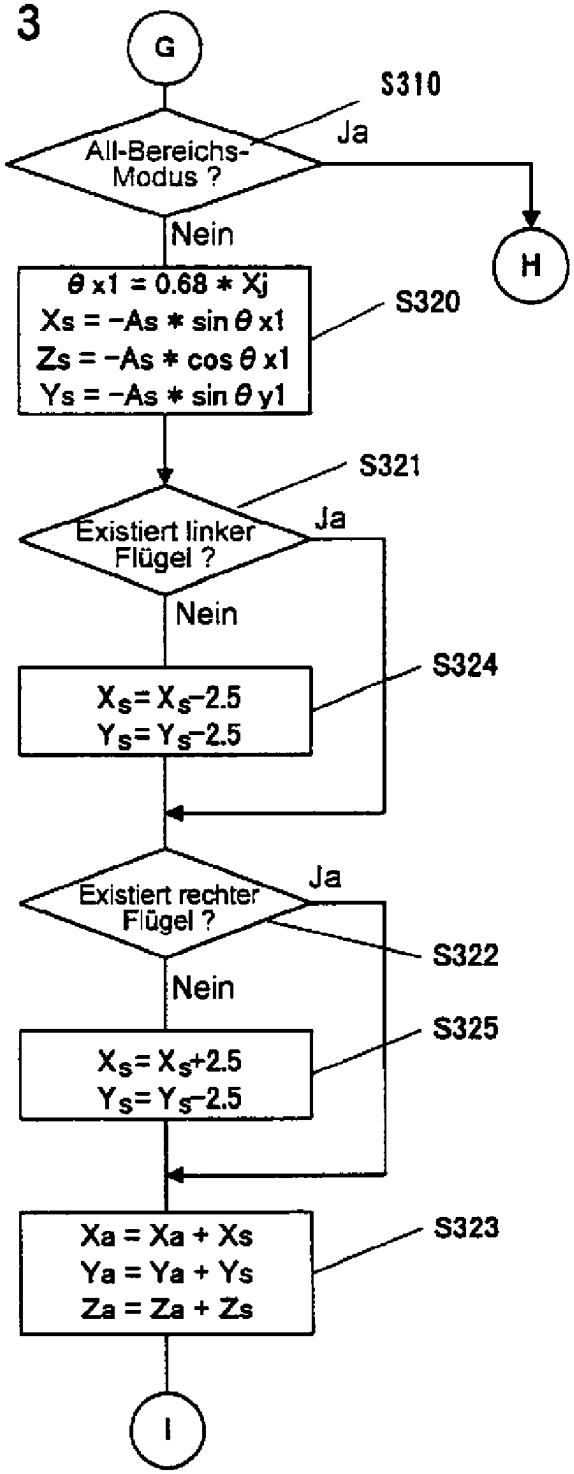

Spieler-Objekt-Prozeß 1a
(Position)
Start
Korrekte Joystick-Daten
S301
Bestimme Xj, Yj
S302
Setze As0
S303
S304
C links ?
Nein
Ja
S305
Vibrations-Prozeß
S306
As = As + As α
S307
C rechts ?
Nein
Ja
S308
Vibrations-Prozeß
S309
As = As − As β
G

FIG. 32

Spieler-Objekt-Prozeß 1b
(Position)

FIG. 33



Ein-Richtungs-Abroll-Modus

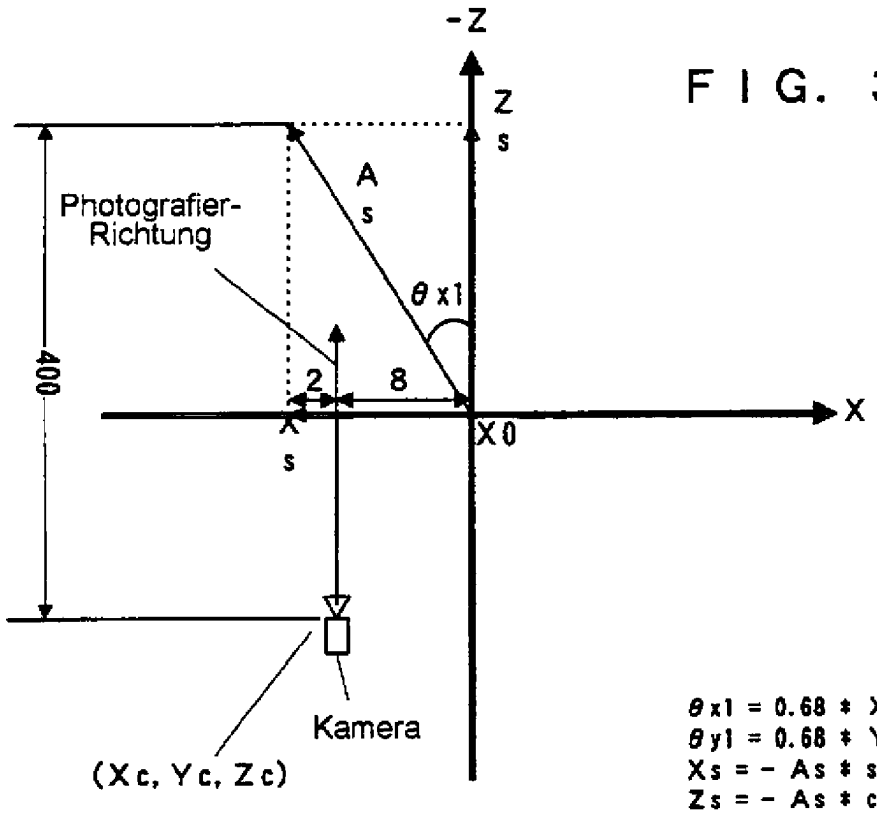


FIG. 34

$$\theta x1 = 0.68 * Xj$$
$$\theta y1 = 0.68 * Yj$$
$$Xs = -As * \sin\theta x1$$
$$Zs = -As * \cos\theta x1$$
$$Ys = -As * \sin\theta y1$$

$$\begin{cases} Xc = (Xa - X0) * 0.8 \\ Yc = (Ya - Y0) * 0.8 \\ Zc = (Za + abs(Z\alpha)) + 400 \\ Zc = (Za - abs(Z\beta)) + 400 \\ Zc = Za + 400 \end{cases}$$

FIG. 35

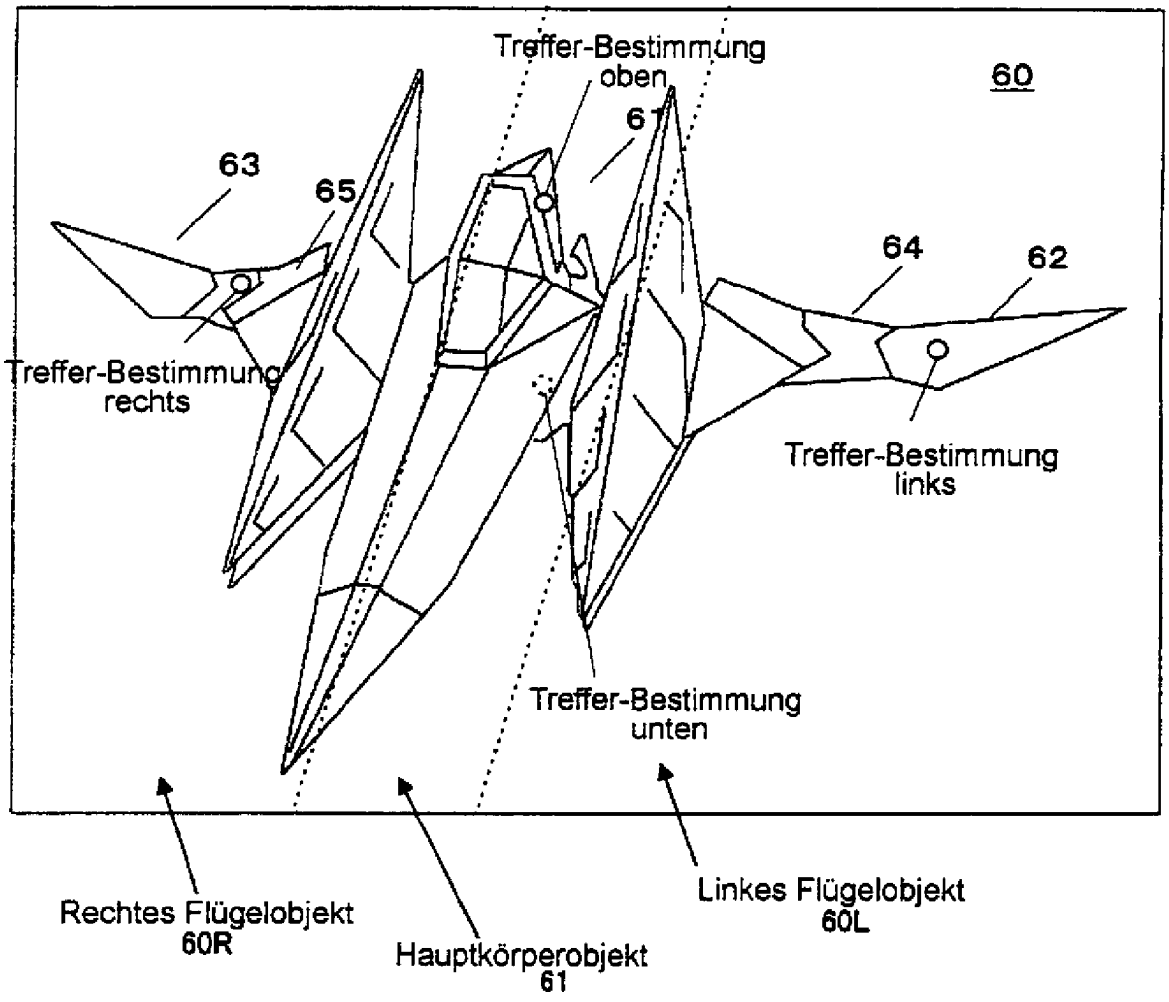

Treffer-Bestimmung
oben
60
61
63
65
64
62
Treffer-Bestimmung
rechts
Treffer-Bestimmung
links
Treffer-Bestimmung
unten
Rechtes Flügelobjekt
60R
Hauptkörperobjekt
61
Linkes Flügelobjekt
60L

FIG. 36

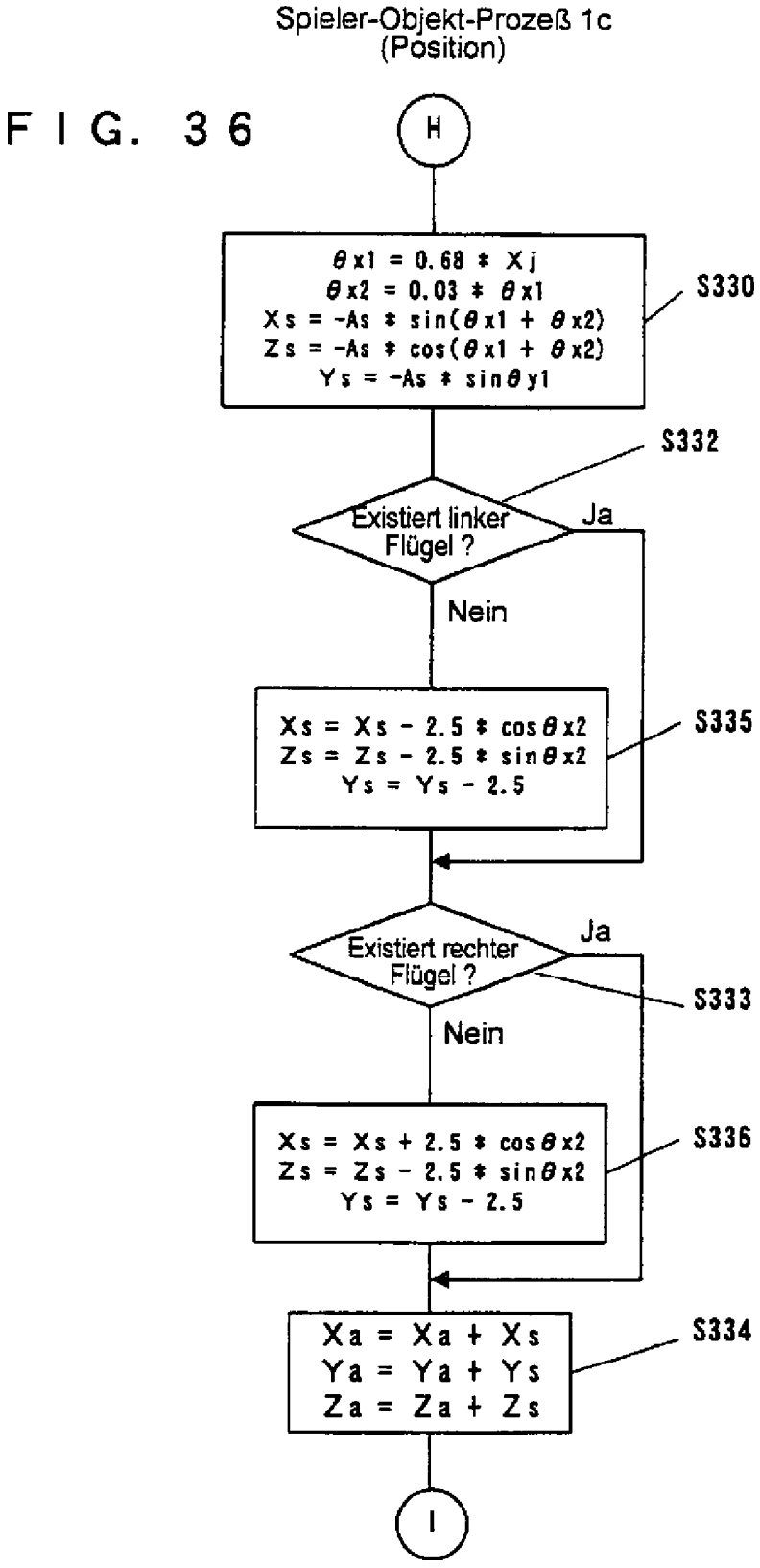
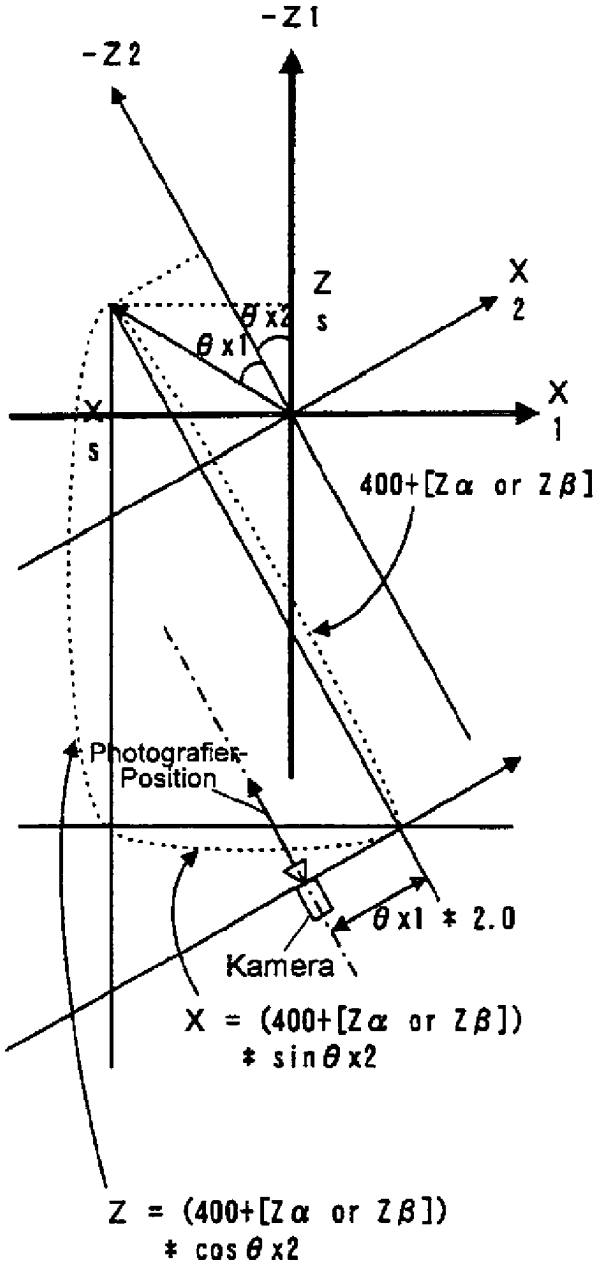

Spieler-Objekt-Prozeß 1c
(Position)
H
θx1 = 0.68 * Xj
θx2 = 0.03 * θx1
Xs = -As * sin(θx1 + θx2)
Zs = -As * cos(θx1 + θx2)
Ys = -As * sinθy1
S330
S332
Existiert linker Flügel ?
Ja
Nein
Xs = Xs - 2.5 * cosθx2
Zs = Zs - 2.5 * sinθx2
Ys = Ys - 2.5
S335
Existiert rechter Flügel ?
Ja
S333
Nein
Xs = Xs + 2.5 * cosθx2
Zs = Zs - 2.5 * sinθx2
Ys = Ys - 2.5
S336
Xa = Xa + Xs
Ya = Ya + Ys
Za = Za + Zs
S334
I

FIG. 37

All-Bereichs-Modus



Winkel

$\theta x1 = 0.68 * Xj$

$\theta y1 = 0.68 * Yj$

$\theta x2 = 0.03 * \theta x1 + \theta x2$

(Vorheriger Rahmenwert)

Geschwindigkeit

$Xs = - As * \sin(\theta x1 + \theta x2)$

$Zs = - As * \cos(\theta x1 + \theta x2)$

$Ys = - As * \sin\theta y1$

Position

$Xa = Xa + Xs$

$Zs = Za + Zs$

$Ys = Ya + Ys$

Xa, Ya, Za :

Xs, Ys, Zs :

Kamera-Position

$Xc = Xa + (400+[0 \text{ or } abs(Z\alpha) \text{ or } - abs(Z\beta)]) * \sin\theta x2 + Xr$

$Zc = Za + (400+[0 \text{ or } abs(Z\alpha) \text{ or } - abs(Z\beta)]) * \cos\theta x2 + Zr$

$Yc = Ya * 0.8$

$Xr = \theta x1 * 2.0 * \cos\theta x2$

$Zr = \theta x1 * 2.0 * \sin\theta x2$

$Z\alpha = As\alpha * \cos\theta x1$

$Z\beta = As\beta * \cos\theta x1$

FIG. 38

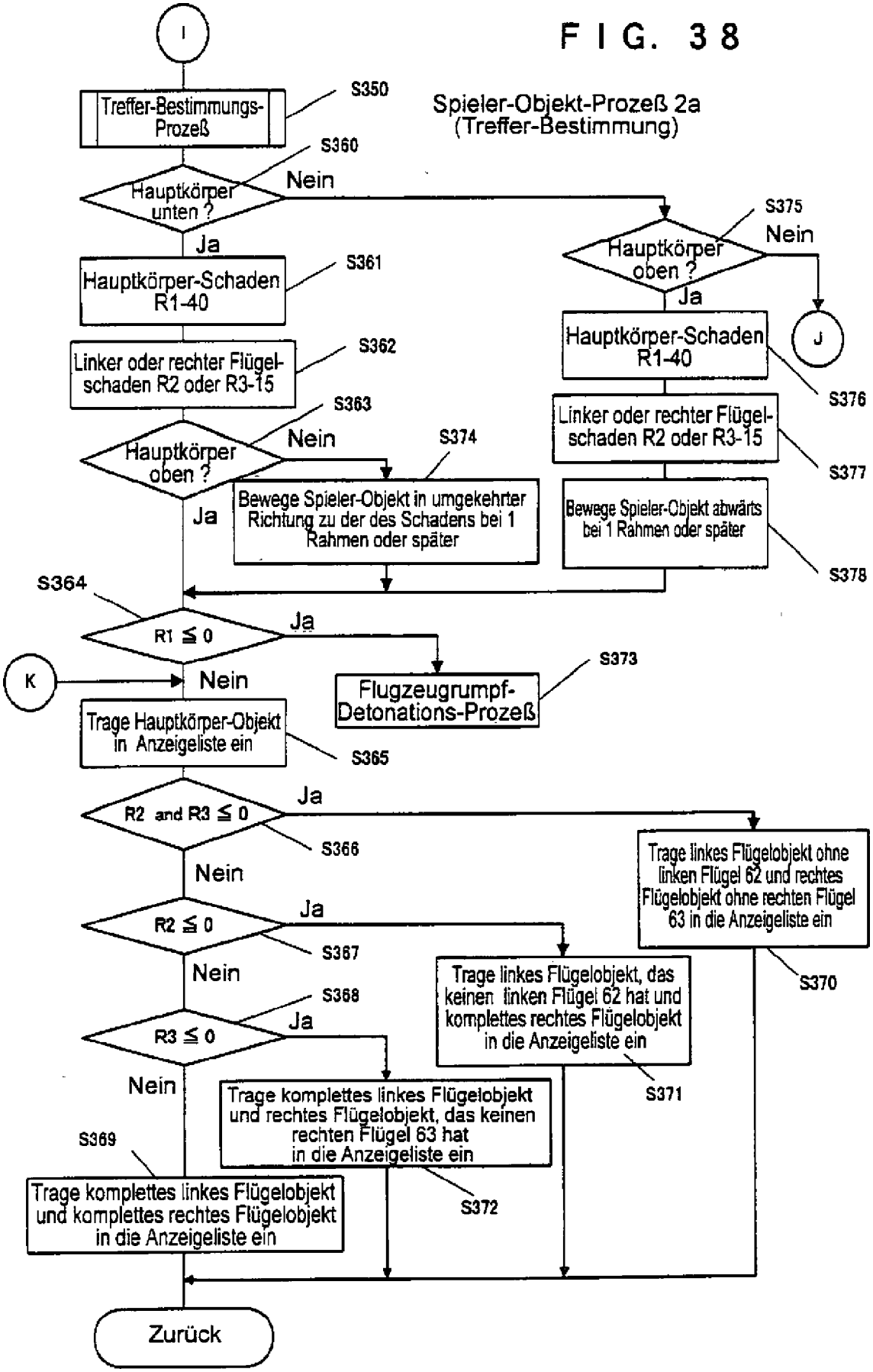

I
Treffer-Bestimmungs-Prozeß
S350
Spieler-Objekt-Prozeß 2a
(Treffer-Bestimmung)
S360
Hauptkörper unten ?
Nein
Ja
S375
Hauptkörper oben ?
Nein
Ja
Hauptkörper-Schaden R1-40
S361
Hauptkörper-Schaden R1-40
J
S376
Linker oder rechter Flügel-schaden R2 oder R3-15
S362
Linker oder rechter Flügel-schaden R2 oder R3-15
S377
S363
Hauptkörper oben ?
Nein
S374
Ja
Bewege Spieler-Objekt in umgekehrter Richtung zu der des Schadens bei 1 Rahmen oder später
Bewege Spieler-Objekt abwärts bei 1 Rahmen oder später
S378
S364
R1 ≦ 0
Ja
K
Nein
Flugzeugrumpf-Detonations-Prozeß
S373
Trage Hauptkörper-Objekt in Anzeigeliste ein
S365
R2 and R3 ≦ 0
Ja
S366
Nein
Trage linkes Flügelobjekt ohne linken Flügel 62 und rechtes Flügelobjekt ohne rechten Flügel 63 in die Anzeigeliste ein
S370
R2 ≦ 0
Ja
S367
Nein
Trage linkes Flügelobjekt, das keinen linken Flügel 62 hat und komplettes rechtes Flügelobjekt in die Anzeigeliste ein
S371
S368
R3 ≦ 0
Ja
Nein
Trage komplettes linkes Flügelobjekt und rechtes Flügelobjekt, das keinen rechten Flügel 63 hat in die Anzeigeliste ein
S372
S369
Trage komplettes linkes Flügelobjekt und komplettes rechtes Flügelobjekt in die Anzeigeliste ein
Zurück

FIG. 39

Treffer-Bestimmung (Standard)

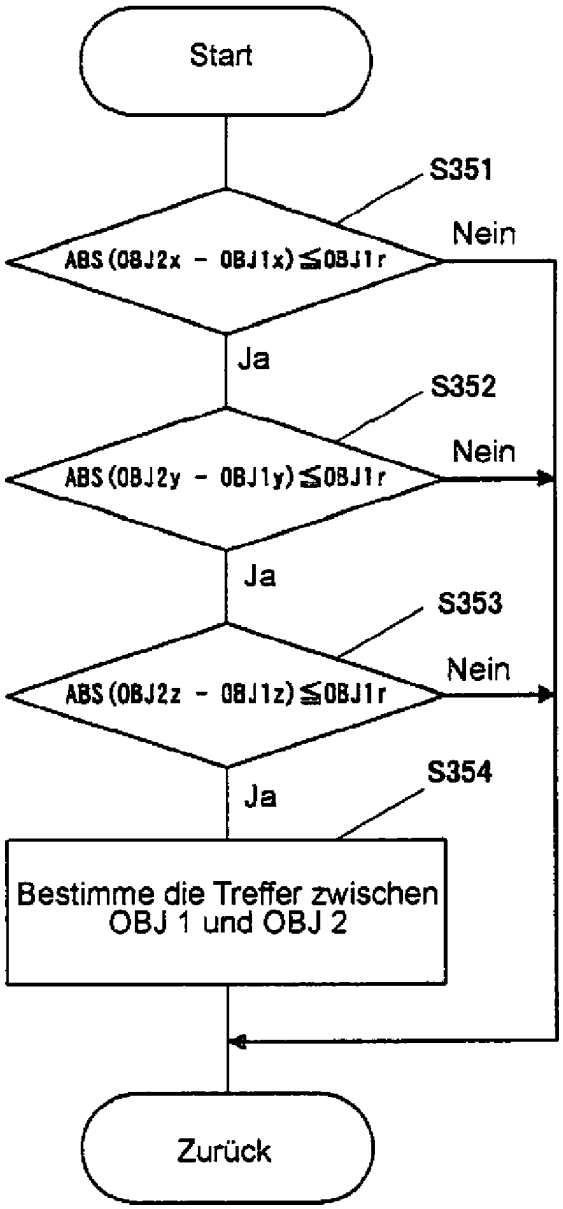


FIG. 40

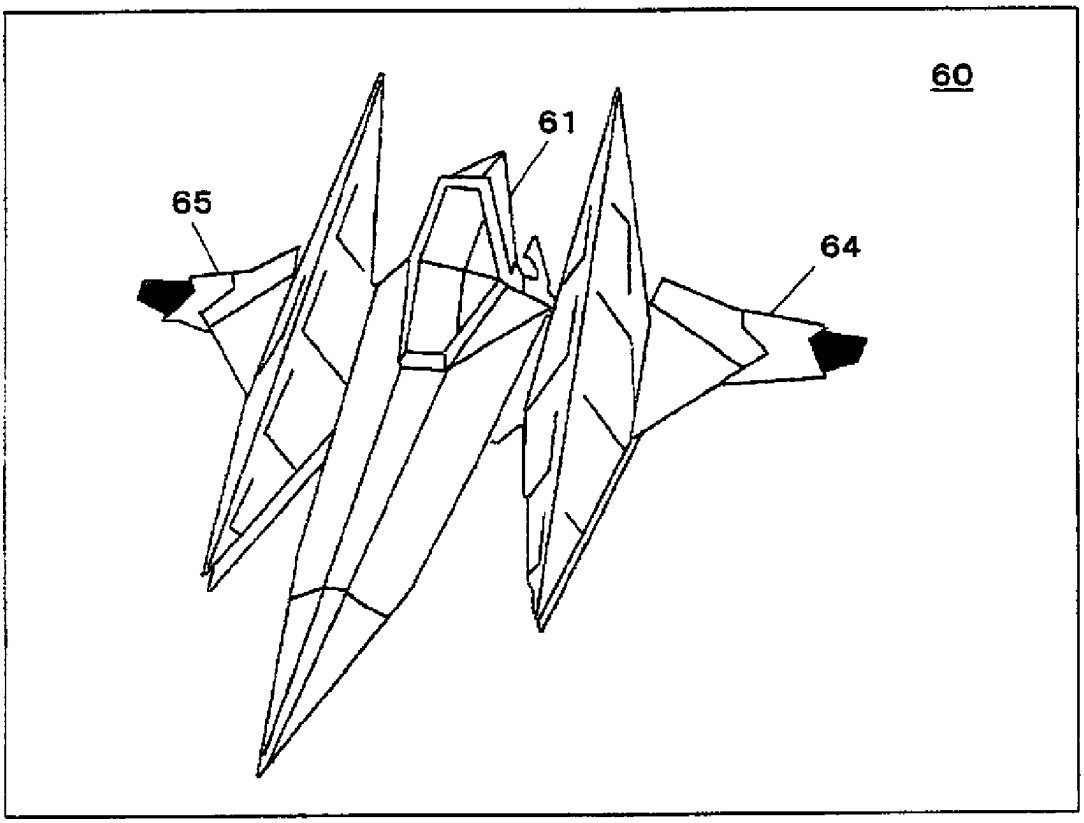


FIG. 41

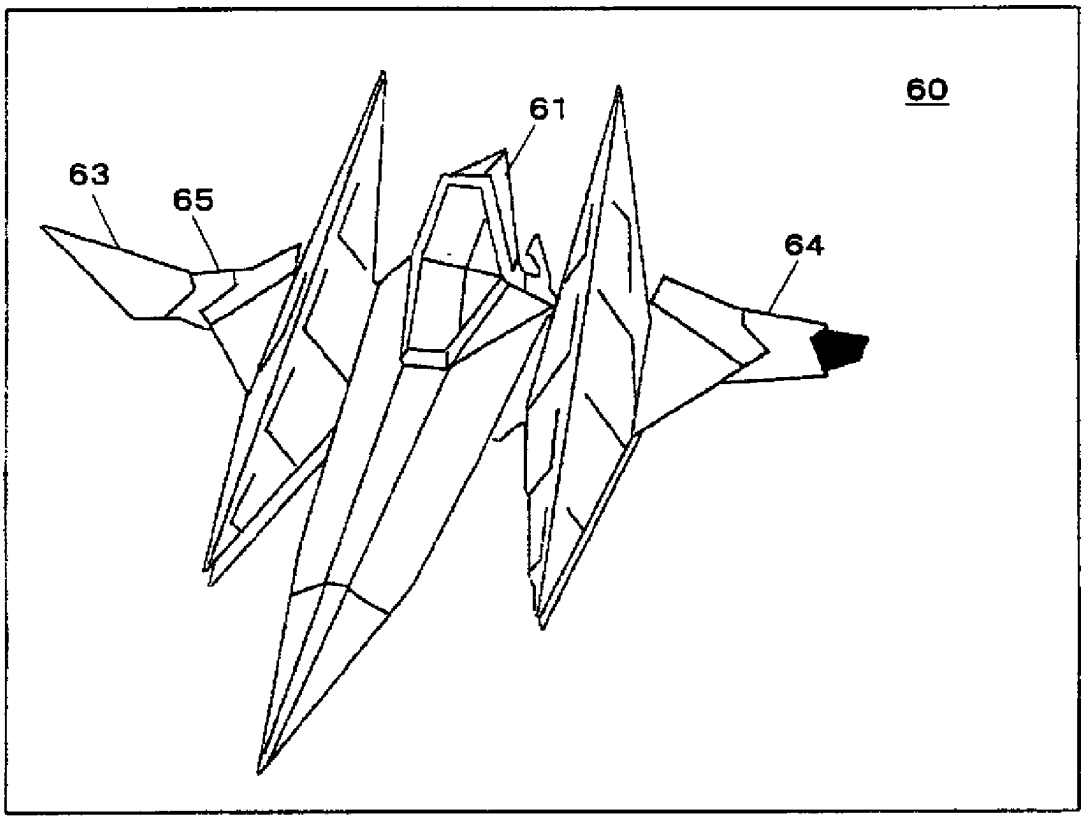


FIG. 42

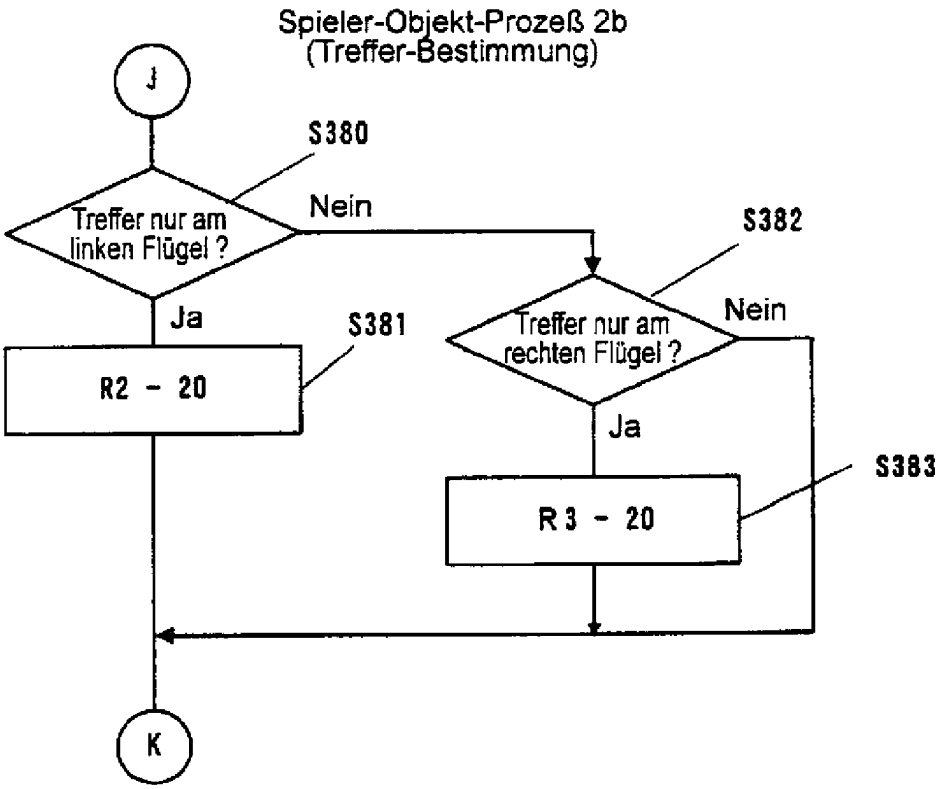
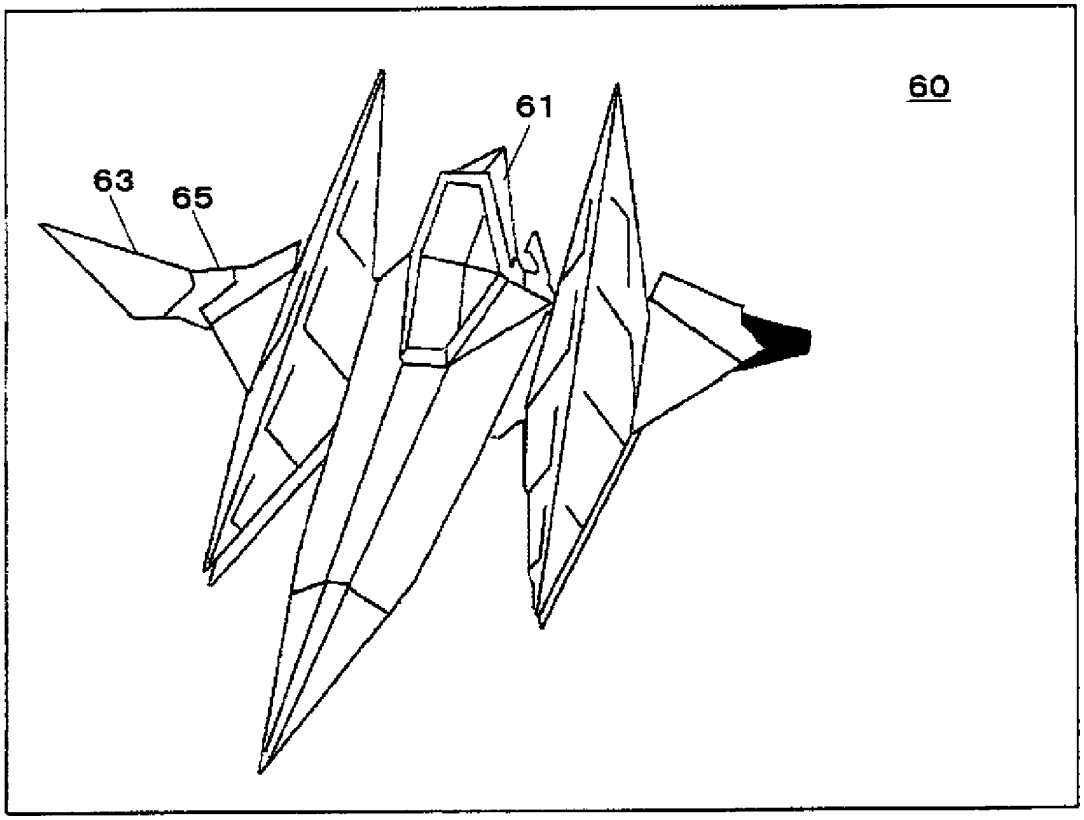

Spieler-Objekt-Prozeß 2b
(Treffer-Bestimmung)
J
S380
Treffer nur am linken Flügel ?
Nein
S382
Ja
S381
Treffer nur am rechten Flügel ?
Nein
R2 - 20
Ja
S383
R3 - 20
K

FIG. 43



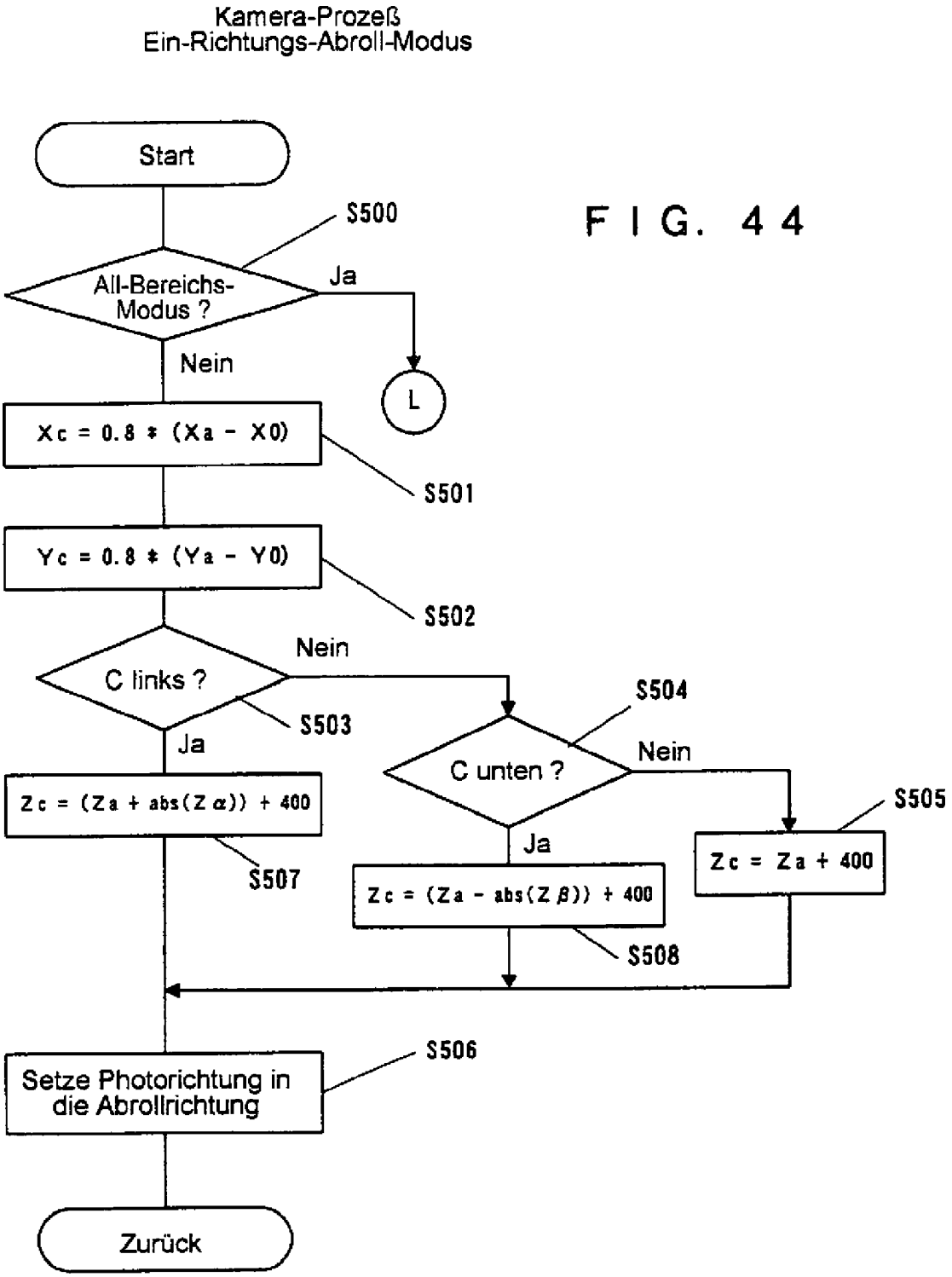
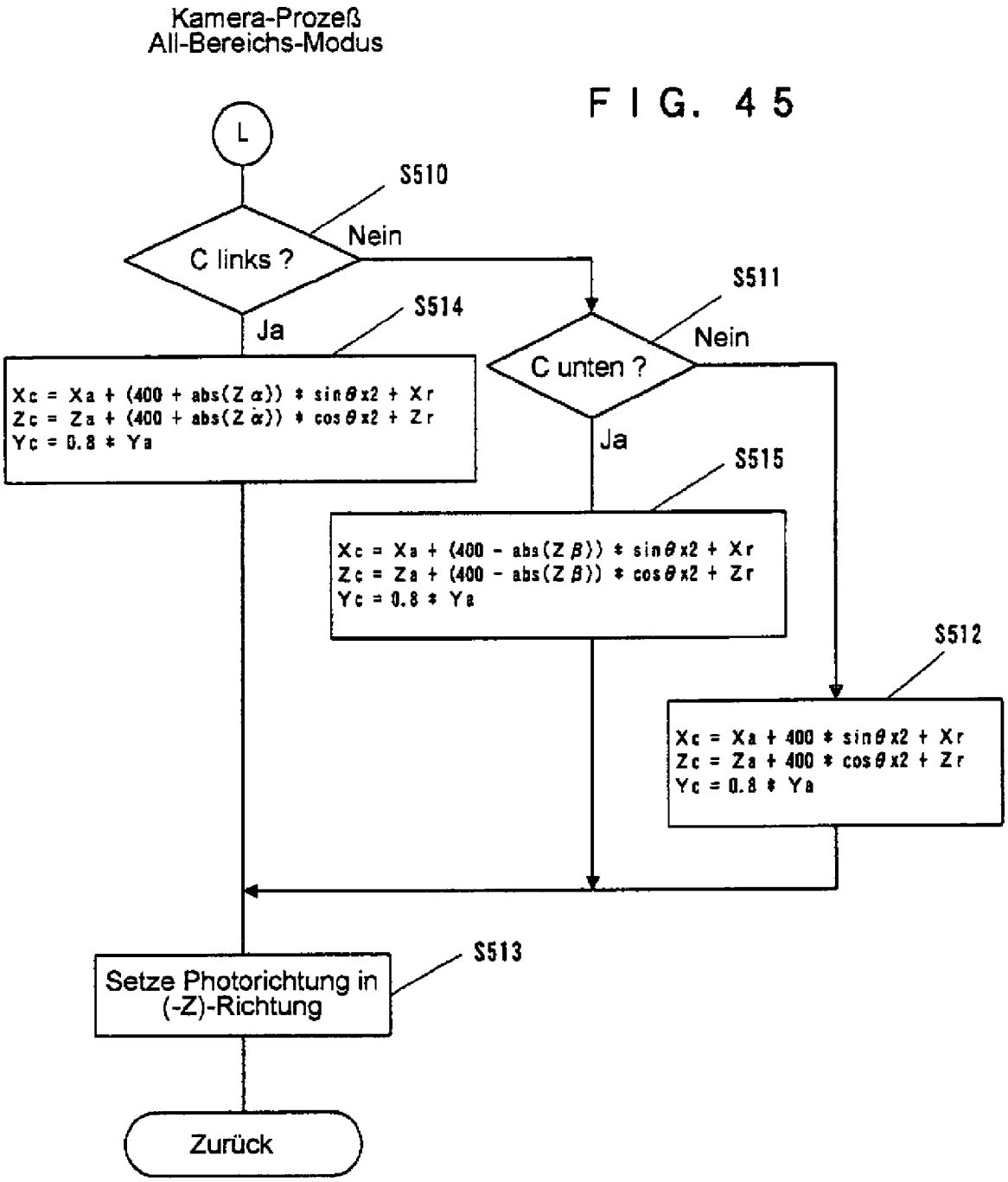

Kamera-Prozeß
Ein-Richtungs-Abroll-Modus
Start
S500
All-Bereichs-Modus ?
Ja
Nein
L
Xc = 0.8 * (Xa - X0)
S501
Yc = 0.8 * (Ya - Y0)
S502
Nein
C links ?
S503
Ja
S504
C unten ?
Nein
S505
Zc = Za + 400
Ja
Zc = (Za + abs(Zα)) + 400
S507
Zc = (Za - abs(Zβ)) + 400
S508
S506
Setze Photorichtung in die Abrollrichtung
Zurück

FIG. 44


Kamera-Prozeß
All-Bereichs-Modus
L
S510
C links ?
Nein
Ja
S514
Xc = Xa + (400 + abs(Z α)) * sinθx2 + Xr
Zc = Za + (400 + abs(Z α)) * cosθx2 + Zr
Yc = 0.8 * Ya
S511
C unten ?
Nein
Ja
S515
Xc = Xa + (400 - abs(Z β)) * sinθx2 + Xr
Zc = Za + (400 - abs(Z β)) * cosθx2 + Zr
Yc = 0.8 * Ya
S512
Xc = Xa + 400 * sinθx2 + Xr
Zc = Za + 400 * cosθx2 + Zr
Yc = 0.8 * Ya
S513
Setze Photorichtung in
(-Z)-Richtung
Zurück

F I G. 4 5

FIG. 46

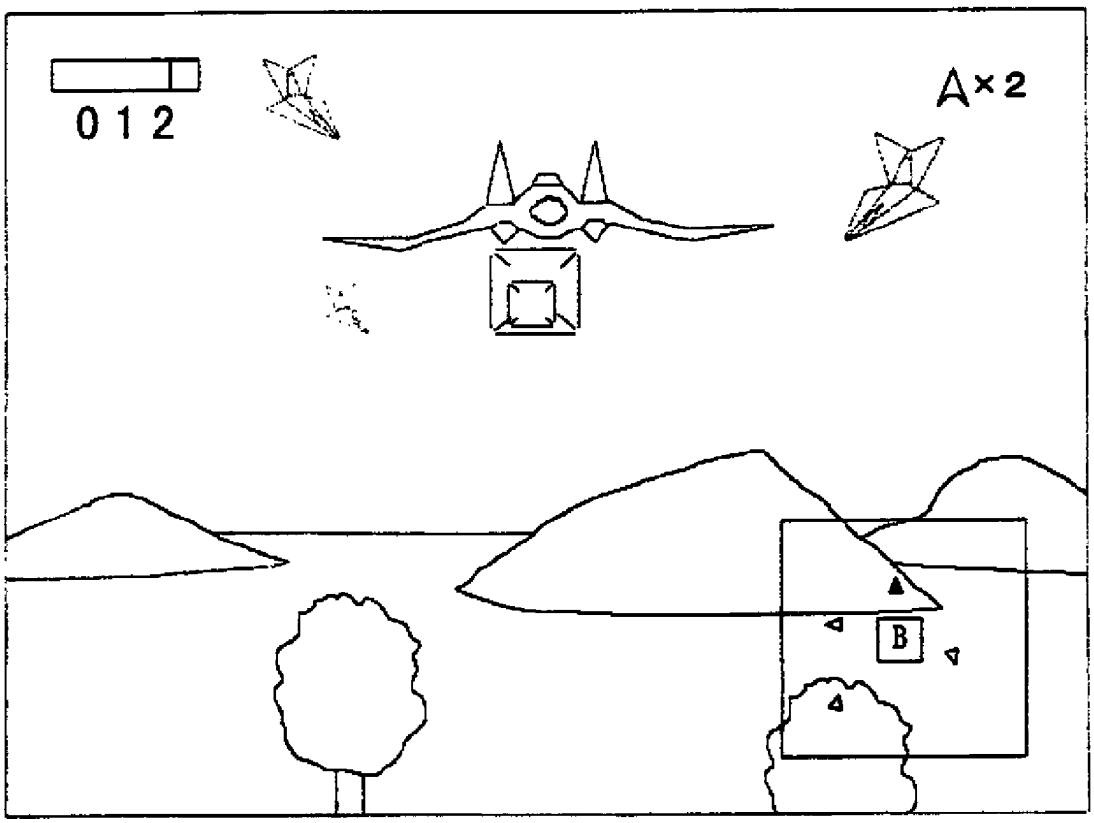


FIG. 47

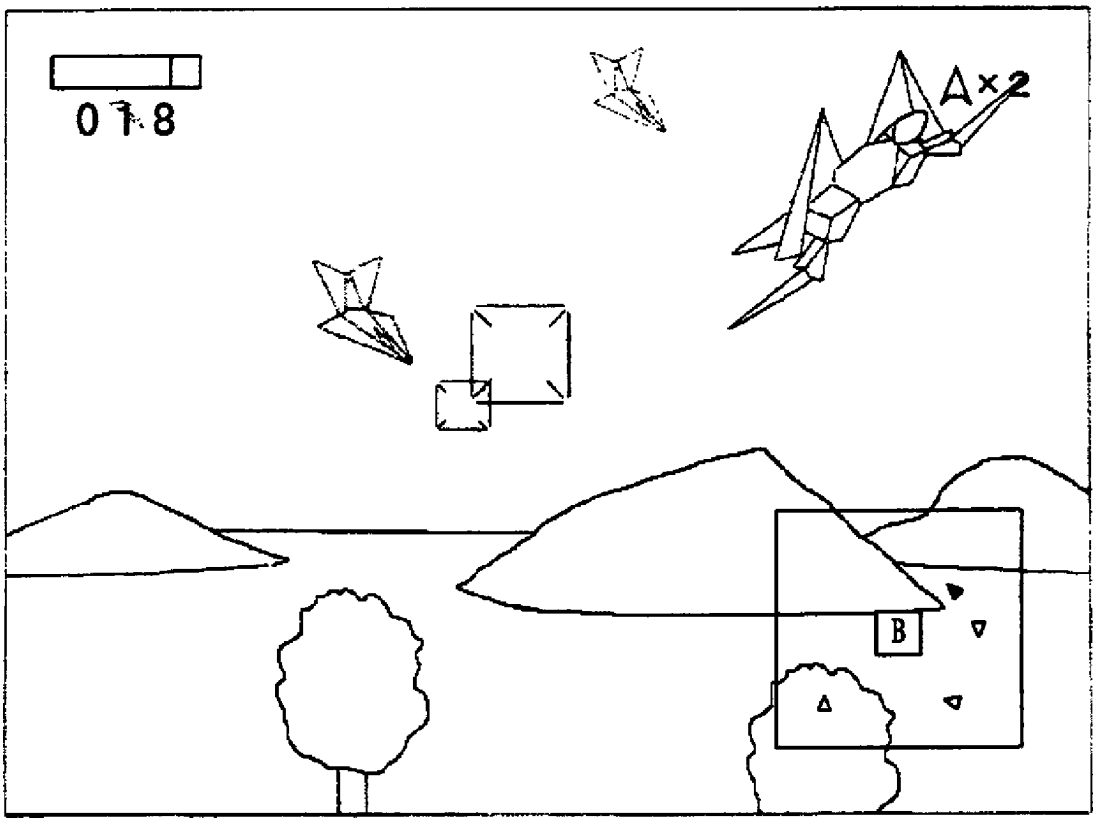
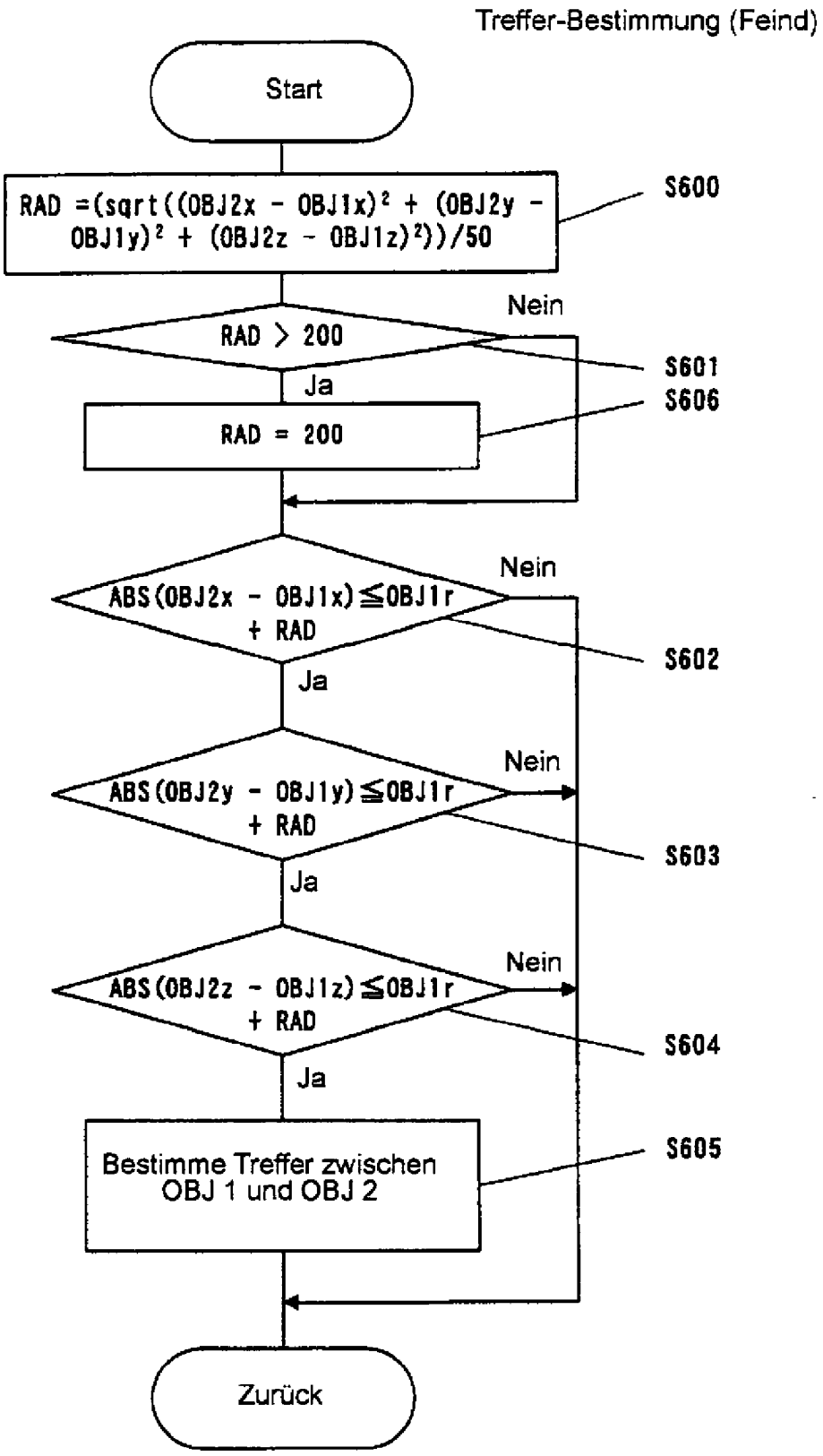


FIG. 48


Treffer-Bestimmung (Feind)
Start
RAD =(sqrt((OBJ2x - OBJ1x)² + (OBJ2y - OBJ1y)² + (OBJ2z - OBJ1z)²))/50
S600
RAD > 200
Nein
S601
Ja
RAD = 200
S606
ABS(OBJ2x - OBJ1x)≦OBJ1r + RAD
Nein
S602
Ja
ABS(OBJ2y - OBJ1y)≦OBJ1r + RAD
Nein
S603
Ja
ABS(OBJ2z - OBJ1z)≦OBJ1r + RAD
Nein
S604
Ja
Bestimme Treffer zwischen OBJ 1 und OBJ 2
S605
Zurück