重力方向に基づくコントローラの向き決定方法

山口兼太郎*
(株) バンダイナムコゲームス
コンテンツ制作本部 制作ディビジョン 技術部 技術サポート課

2/Sep '09

1 重力方向の検出とコントローラの向き決定

近頃のゲーム機のコントローラや携帯電話には，入力用のデバイスとして，加速度センサが使われている。加速度センサが検出する加速度は，重力加速度と運動加速度 (動きによる加速度) との和である。そのため，加速運動をしていない状態 (例えば静止状態) であれば，加速度センサを使って重力の方向を知ることができる。

これを利用して，3 次元空間内でのコントローラの向きを取得し，ゲーム等の入力として使用することが考えられる。しかし，重力方向から分かるのは上下の向きだけなので，コントローラの向きを決めるには情報が足りない。このままではゲームの入力として使いにくいので，何らかの計算条件を追加することにより向きを決める。当然ながら，得られるコントローラの向きは，現実のコントローラの向きと，必ずしも一致する訳ではない。追加する計算条件次第で，得られるコントローラの向きは違ったものとなり，ゲーム入力としての使い勝手の良し悪しも変わる。

個々の向き決め方法の特徴を調べ改良を図るとともに，向き決め問題の包括的な捉え方についても考察する。

2 座標系とコントローラの向き

ワールド座標系を X_w, Y_w, Z_w とし，各座標軸の向きを $+X_w =$ 右方向，$+Y_w =$ 上方向，$+Z_w =$ 手前方向と定める。一方，コントローラ座標系を図 1 のように X_c, Y_c, Z_c とする。X_c, Y_c, Z_c と X_w, Y_w, Z_w の各軸が一致した状態を基準状態と呼ぶことにする。

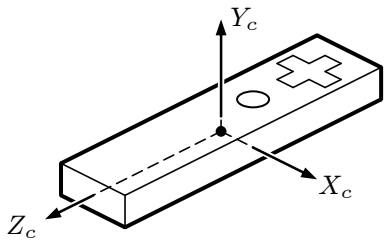

図 1: コントローラ座標系 X_c, Y_c, Z_c

*Kentaro_Yamaguchi@bandainamcogames.co.jp

「ワールド座標系に対するコントローラ座標系の向き」を表すマトリクスを $\mathbf{M}$ とする。マトリクスの要素の並びは，列ベクトルに対して左からマトリクスを掛ける方式とする。$\mathbf{M}$ の要素を次のように定める。

$$\mathbf{M} \equiv \begin{bmatrix} v_{0x} & v_{0y} & v_{0z} \\ v_{1x} & v_{1y} & v_{1z} \\ v_{2x} & v_{2y} & v_{2z} \end{bmatrix} \tag{1}$$

また，ベクトル $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2$ を次のように定義する。

$$\mathbf{v}_0 \equiv \begin{bmatrix} v_{0x} \\ v_{0y} \\ v_{0z} \end{bmatrix}, \quad \mathbf{v}_1 \equiv \begin{bmatrix} v_{1x} \\ v_{1y} \\ v_{1z} \end{bmatrix}, \quad \mathbf{v}_2 \equiv \begin{bmatrix} v_{2x} \\ v_{2y} \\ v_{2z} \end{bmatrix} \tag{2}$$

$\mathbf{M}$ の要素を，3 つの行ベクトル (列ベクトルの転置として表す) の並びと見なして，便宜上次のように書き表すことにする。

$$\mathbf{M} \equiv \begin{bmatrix} \mathbf{v}_0^{\mathrm{T}} \\ \mathbf{v}_1^{\mathrm{T}} \\ \mathbf{v}_2^{\mathrm{T}} \end{bmatrix} \tag{3}$$

$\mathbf{M}$ は向きを表すマトリクス (回転マトリクス) なので，$\mathbf{M}^{-1} = \mathbf{M}^{\mathrm{T}}$ となる。$\mathbf{M}^{-1}$ は,「コントローラ座標系に対するワールド座標系の向き」を表すマトリクスである。$\mathbf{M}^{-1}$ の要素を，3 つの列ベクトルの並びと見なして，便宜上次のように書き表すことにする。

$$\mathbf{M}^{-1} \equiv \begin{bmatrix} v_{0x} & v_{1x} & v_{2x} \\ v_{0y} & v_{1y} & v_{2y} \\ v_{0z} & v_{1z} & v_{2z} \end{bmatrix} \equiv \begin{bmatrix} \mathbf{v}_0 & \mathbf{v}_1 & \mathbf{v}_2 \end{bmatrix} \tag{4}$$

また，各軸方向の単位ベクトル $\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2$ を次のように定義する。

$$\mathbf{e}_0 \equiv \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{e}_1 \equiv \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{e}_2 \equiv \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{5}$$

すると次式が成り立つ。

$$\mathbf{M}^{-1}\,\mathbf{e}_0 = \mathbf{v}_0 \qquad \mathbf{M}^{-1}\,\mathbf{e}_1 = \mathbf{v}_1 \qquad \mathbf{M}^{-1}\,\mathbf{e}_2 = \mathbf{v}_2 \tag{6}$$

$$\mathbf{M}\,\mathbf{v}_0 = \mathbf{e}_0 \qquad \mathbf{M}\,\mathbf{v}_1 = \mathbf{e}_1 \qquad \mathbf{M}\,\mathbf{v}_2 = \mathbf{e}_2 \tag{7}$$

したがって，コントローラ座標系におけるベクトル $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2$ は，それぞれワールド座標系におけるベクトル $\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2$ (即ち $+X_w, +Y_w, +Z_w$ 方向の単位ベクトル) に一致することが分かる。

3 問題の定義

3.1 アップベクトル

加速度センサによって検出された上方向ベクトルを「アップベクトル」と呼ぶことにし，$\mathbf{v}_u$ と表記する。$|\mathbf{v}_u| = 1$ とし，$\mathbf{v}_u$ の要素は次のとおりとする。

$$\mathbf{v}_u \equiv \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \tag{8}$$

加速度センサが検出した加速度ベクトル (重力加速度) を $\mathbf{a} = [a_x\ a_y\ a_z]^{\mathrm{T}}$ とする。この $\mathbf{a}$ はコントローラ座標系における値である。アップベクトルは $\mathbf{v}_u = \mathbf{a}/|\mathbf{a}|$ によって得られる。

3.2 問題の定義

コントローラの向き決めにおいて，解くべき問題は次のように表せる。

> 与えられたアップベクトル $\mathbf{v}_u$ から，コントローラの向き $\mathbf{M}$ を決める。

コントローラ座標系のアップベクトル $\mathbf{v}_u$ は，ワールド座標系では上向きの単位ベクトルになるので，次式が成り立つ。

$$\mathbf{M}\,\mathbf{v}_u = \mathbf{e}_1 \tag{9}$$

したがって，

$$\mathbf{v}_u = \mathbf{M}^{-1}\,\mathbf{e}_1 = \mathbf{v}_1 \tag{10}$$

となり，マトリクス $\mathbf{M}$ の要素の 1 行に相当する $\mathbf{v}_1$ は，必ず $\mathbf{v}_u$ に等しくなる。

マトリクス $\mathbf{M}$ の残り要素 ($\mathbf{v}_0$ 及び $\mathbf{v}_2$) については，何らかの計算条件を追加して，決定しなければならない。追加する計算条件は種々考えられるが，この計算条件次第で，コントローラの向き決めの挙動が変わる。

3.3 フリップ現象

コントローラの向き決めにおいて，次のような現象を「フリップ現象」と呼ぶことにする。

> アップベクトル $\mathbf{v}_u$ の連続的な変化に対して，コントローラの向き $\mathbf{M}$ が不連続に変化する現象。

フリップ現象は好ましくない現象であって，可能な限り抑制すべきである。

4 従来手法と問題点

従来手法の代表的な例を，以下に 2 つ挙げる。これらの手法では，フリップ現象が 2 方向で発生する。

4.1 従来手法 (左右方向を維持)

コントローラの向きを決める際に，$+X_c$ 半直線 (X_c 軸の $X_c \geq 0$ の領域) が X_wY_w 平面内の $X_w \geq 0$ の領域内に入るようにする。$\mathbf{M}$ の要素は次式により得られる。

$$\mathbf{v}_1 = \mathbf{v}_u \tag{11}$$

$$v_{2x} = 0,\ v_{2y} = \frac{-u_z}{\sqrt{u_y^2 + u_z^2}},\ v_{2z} = \frac{u_y}{\sqrt{u_y^2 + u_z^2}} \tag{12}$$

$$\mathbf{v}_0 = \mathbf{v}_1 \times \mathbf{v}_2 \tag{13}$$

この向き決め方法は，次のように表すこともできる。

- X_c は Z_w に垂直。
- Y_w に垂直な平面と Y_cZ_c 平面との交線を Z_w とする。

この手法でフリップが起こるのは，$\mathbf{v}_u$ が $\pm X_c$ 方向を向く場合，即ちプレイヤがコントローラの $+X_c$ を真上または真下に向けて保持した場合である。

4.2 従来手法 (前後方向を維持)

コントローラの向きを決める際に，$+Z_c$ 半直線が Y_wZ_w 平面内の $Z_w \geq 0$ の領域内に入るようにする。$\mathbf{M}$ の要素は次式により得られる。

$$\mathbf{v}_1 = \mathbf{v}_u \tag{14}$$

$$v_{0x} = \frac{u_y}{\sqrt{u_x^2 + u_y^2}},\ v_{0y} = \frac{-u_x}{\sqrt{u_x^2 + u_y^2}},\ v_{0z} = 0 \tag{15}$$

$$\mathbf{v}_2 = \mathbf{v}_0 \times \mathbf{v}_1 \tag{16}$$

この向き決め方法は，次のように表すこともできる。

- Z_c は X_w に垂直。
- Y_w に垂直な平面と X_cY_c 平面との交線を X_w とする。

この手法でフリップが起こるのは，$\mathbf{v}_u$ が $\pm Z_c$ 方向を向く場合，即ちプレイヤがコントローラの $+Z_c$ を真上または真下に向けて保持した場合である。

5 向き決め方法の分類

コントローラの向き決め方法は，次の 2 タイプに分類できる。

- 履歴非依存型:
  コントローラの向き $\mathbf{M}$ は，過去のアップベクトルには依存せず，現在のアップベクトルだけで決まる。
- 履歴依存型:
  コントローラの向き $\mathbf{M}$ は，現在のアップベクトルだけでは決まらず，過去のアップベクトルに依存する。

この分類に基づいて，それぞれの特徴を以下の節で述べる。

6 履歴非依存型と特性グリッド

前述の従来手法は，履歴非依存型である。まず，履歴非依存型の挙動について考える。

図 2 は，コントローラ座標系を基準として，あるアップベクトル $\mathbf{v}_u$ を表示し，さらに $\mathbf{v}_u$ の終点を起点として，これに対応する X_w, Y_w, Z_w の向きを示したものである。Y_w はワールドの上方向なので，Y_w と $\mathbf{v}_u$ の向きは一致する。したがって X_w, Z_w は $\mathbf{v}_u$ に垂直になる。

$\mathbf{v}_u$ の向きを変化させると図 3 のようになる。$|\mathbf{v}_u| = 1$ なので，$\mathbf{v}_u$ の終点は，原点を中心とする半径 1 の球面上に位置する。この球面を「球面 S」とする。X_w, Z_w は，$\mathbf{v}_u$ に垂直なので，球面 S

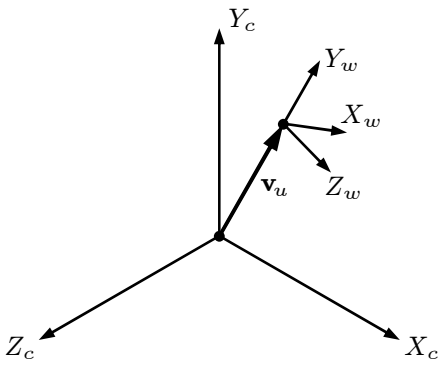

図 2: アップベクトル $\mathbf{v}_u$ と X_w, Y_w, Z_w

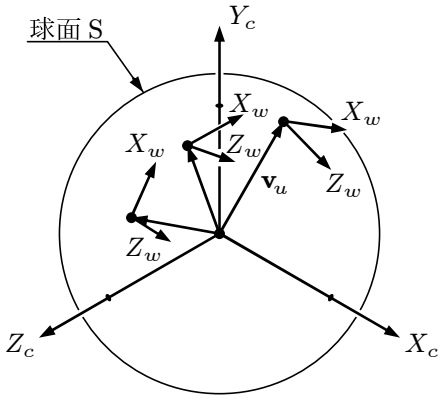

図 3: 個々のアップベクトルに対する X_w, Z_w

に接する。履歴に依存しないので，各 $\mathbf{v}_u$ に対して X_w, Y_w, Z_w の向きは一意に定まる。したがって，球面 S 上の各点に対して接ベクトル X_w, Z_w が一意に定まることになる。

$\mathbf{v}_u$ の変化に対して X_w, Z_w が連続的に変化するならば，図 4 のように，球面 S 上に X_w, Z_w の向きを表すグリッドを構成できる。このグリッドは，コントローラの向き決め方式の挙動特性を表す。そこで，これを「特性グリッド」と呼ぶことにする。さらに，この特性グリッドはコントローラ座標系に固定されているので，そのことを明確にする際には,「コントローラ固定特性グリッド」と称する。このように，履歴非依存型の挙動特性は，コントローラ固定特性グリッドで表すことができる。

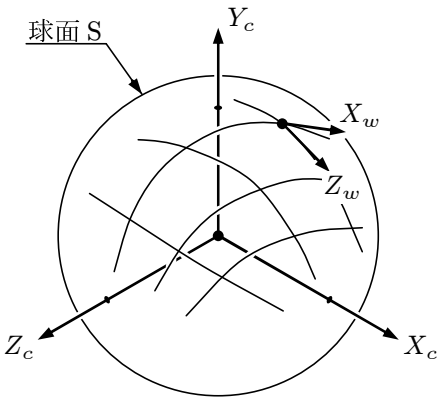

図 4: 球面 S 上に構成された特性グリッド

前述の従来手法の第 1 例 (左右方向を維持) では，特性グリッドは図 5 のように，X_c 軸を地軸とする地球儀の緯線・経線の形状になる。$+X_c$ を地軸の北とすると，球面 S 上の各点において，$+X_w$ は経線の北方向，$+Z_w$ は緯線の東方向である。

この特性グリッドから，北極・南極に相当する 2 点が特異点であることが分かる。アップベクトルがこの特異点方向 ($+X_c$ 方向，$-X_c$ 方向) を向いた場合，コントローラの向きが不定となる。また，アップベクトルが特異点を通過すると，コントローラの向きは不連続に変化し，フリップ現象が起こる。実用上は，特異点において向き不定とする替わりに，何らかの向きをコントローラに

与えることが多いが，その場合でも向きの変化の不連続性は残るためフリップ現象が起こる。

このように，特性グリッドの特異点に着目することにより，コントローラの向き決めにおけるフリップ現象の発生およびフリップ方向を，明確に知ることができる。

同様に，従来手法の第 2 例 (前後方向を維持) では，特性グリッドは図 6 のようになる。この特性グリッドから，$+Z_c$ と $-Z_c$ の 2 方向がフリップ方向となることが分かる。

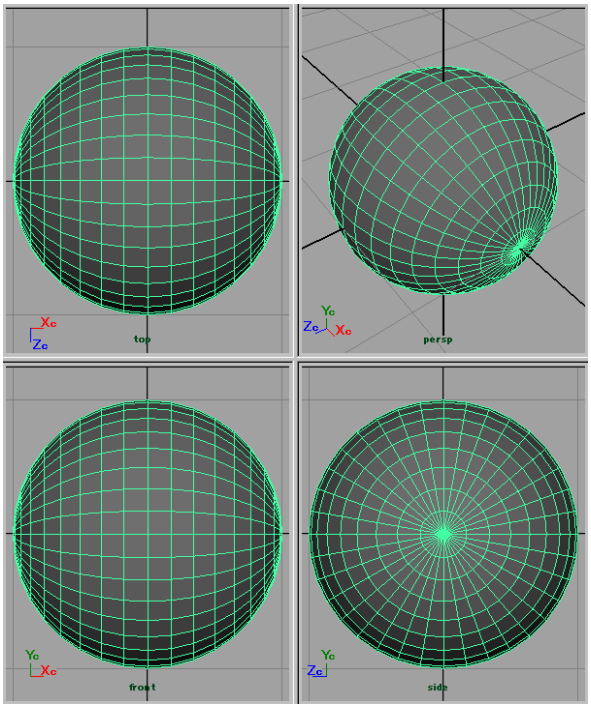

図 5: 従来手法の第 1 例 (左右方向を維持) の特性グリッド

7 単純回転

今後の手法説明で使用するために，ここで以下のように「単純回転」を定義する (図 7 参照)。

始点を共有する任意の単位ベクトル $\mathbf{v}_a, \mathbf{v}_b$ に対して，次の条件を満たす回転を，$\mathbf{v}_a$ から $\mathbf{v}_b$ への単純回転と定める。

- $\mathbf{v}_a, \mathbf{v}_b$ のなす平面に垂直な軸の回りの回転。
- 回転により $\mathbf{v}_a$ は $\mathbf{v}_b$ に重なる。

$\mathbf{v}_a$ と $\mathbf{v}_b$ が同じ向き (即ち $\mathbf{v}_a = \mathbf{v}_b$) の場合は無回転とする (回転マトリクスで表すと単位行列)。
$\mathbf{v}_a$ と $\mathbf{v}_b$ が反対向き (即ち $\mathbf{v}_a = -\mathbf{v}_b$) の場合は定義不能とする。

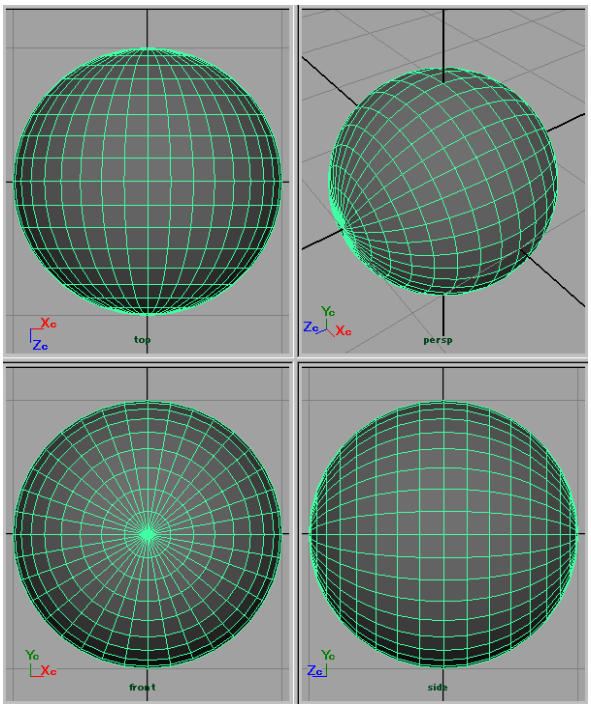

図 6: 従来手法の第 2 例 (前後方向を維持) の特性グリッド

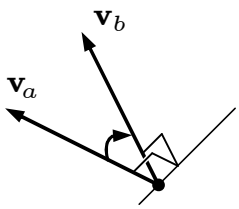

図 7: $\mathbf{v}_a$ から $\mathbf{v}_b$ への単純回転

「$\mathbf{v}_a$ から $\mathbf{v}_b$ への単純回転」を表すマトリクスを，$\mathbf{M}_{SR}(\mathbf{v}_a, \mathbf{v}_b)$ と書き表すことにする。このマトリクスは次式により得られる。

$$\mathbf{M}_{SR}(\mathbf{v}_a, \mathbf{v}_b) = \begin{bmatrix} n_x^2 t + \cos\phi & n_x n_y t - n_z \sin\phi & n_z n_x t + n_y \sin\phi \\ n_x n_y t + n_z \sin\phi & n_y^2 t + \cos\phi & n_y n_z t - n_x \sin\phi \\ n_z n_x t - n_y \sin\phi & n_y n_z t + n_x \sin\phi & n_z^2 t + \cos\phi \end{bmatrix} \tag{17}$$

ただし，$[n_x\ n_y\ n_z]^{\mathrm{T}}$ は回転軸を表わす単位ベクトル，ϕ は回転角で，以下の式により得られる。

$$[n_x\ n_y\ n_z]^{\mathrm{T}} = \frac{\mathbf{v}_a \times \mathbf{v}_b}{\sin\phi} \tag{18}$$

$$\sin\phi = |\mathbf{v}_a \times \mathbf{v}_b| \tag{19}$$

$$\cos\phi = \mathbf{v}_a \cdot \mathbf{v}_b \tag{20}$$

$$t = 1 - \cos\phi \tag{21}$$

「$\mathbf{v}_a$ から $\mathbf{v}_b$ への単純回転」の逆変換は,「$\mathbf{v}_b$ から $\mathbf{v}_a$ への単純回転」となる。

$$\mathbf{M}_{SR}(\mathbf{v}_a, \mathbf{v}_b)^{-1} = \mathbf{M}_{SR}(\mathbf{v}_b, \mathbf{v}_a) \tag{22}$$

以上の説明では $\mathbf{v}_a, \mathbf{v}_b$ を単位ベクトルに限定したが，一般のベクトルを扱えるように定義を拡張する。大きさが 1 でないベクトルの場合は，それぞれのベクトルを正規化してから上記の方法で求めた回転を，単純回転と定義する。したがって，k, l を任意の正の実数とするとき，次式が成り立つ。

$$\mathbf{M}_{SR}(k\,\mathbf{v}_a, l\,\mathbf{v}_b) = \mathbf{M}_{SR}(\mathbf{v}_a, \mathbf{v}_b) \tag{23}$$

8 履歴非依存型の改良

8.1 フリップ現象と不動点定理

履歴非依存型ではフリップ現象を回避できない。このことは，以下のようにして不動点定理から導かれる。不動点定理には種々のバリエーションがあるが，ここで扱う不動点定理 (Fixed point theorem, Hairy ball theorem) の内容は次のとおりである。

> 連続関数 $f : S^2 \to S^2$ には，$f(P) = P$ となる P が必ず存在する。

ただし，S^2 は「球面」の意味で，P は球面上の点である。

この定理から得られる結果の一例として,「地球上には無風地点が存在する」という例え話がある。この例では，球の表面に沿った連続的な (不連続箇所がない) 流れを想定している。球面上の各点 P に対して，その点から流れに乗って移動した先を $f(P)$ とする。不動点定理を適用すると，$f(P) = P$ 即ち流速が 0 の点が必ず存在する。

この結果の対偶をとれば,「流速が 0 でない流れを作れば，必ず不連続箇所が存在する」ことになる。さらに，両者を合わせると,「球面上の流れには，不連続箇所または流速 0 の箇所が必ず存在する」ことが言える。

履歴非依存型では，図 3 のように，球面 S 上の各点に対して X_w, Z_w の向きを一意に定める。ここで，X_w を球面 S 上の各点における流れの方向と見なす。すると，X_w が不連続または X_w が不定となる特異点が，球面 S 上に必ず存在することになる。

以上のように，履歴非依存型には特異点が必ず存在するため，フリップ現象を回避できない。そこで改良策として，以下に述べる手法を使って，フリップが生じる方向を 2 方向から 1 方向に減らす。

8.2 単純回転方式 (単回転)

前述の単純回転を用いる。ワールド座標系において，次の手順によりコントローラの向き $\mathbf{M}$ を決める。

1. コントローラの向きをワールド座標系に合わせる (基準状態にする)。
2. アップベクトル $\mathbf{v}_u$ から $+Y_w$ ベクトルへの単純回転を，コントローラに作用させる。

ワールド座標系における $+Y_w$ 方向単位ベクトルは $\mathbf{e}_1$ なので，$\mathbf{M}$ は次式で表される (要素の詳細は付録 A 参照)。

$$\mathbf{M} = \mathbf{M}_{SR}(\mathbf{v}_u, \mathbf{e}_1) \tag{24}$$

前述のように，履歴非依存型の挙動特性は，コントローラ固定特性グリッドで表せる。そこで，特性グリッドの形状を把握するために，コントローラ座標系から見たワールドの向き $\mathbf{M}^{-1}$ を求める。

$$\mathbf{M}^{-1} = \mathbf{M}_{SR}(\mathbf{v}_u, \mathbf{e}_1)^{-1} \tag{25}$$

$$= \mathbf{M}_{SR}(\mathbf{e}_1, \mathbf{v}_u) \tag{26}$$

これは $\mathbf{e}_1$ から $\mathbf{v}_u$ への単純回転である。特性グリッドは図 8 のようになり，特異点は $-Y_c$ 方向の 1 箇所のみとなる。したがって，この手法でフリップが起こるのは，$\mathbf{v}_u$ が $-Y_c$ 方向を向く場合，即ちプレイヤがコントローラの $-Y_c$ を真上に向けて保持した場合のみである。

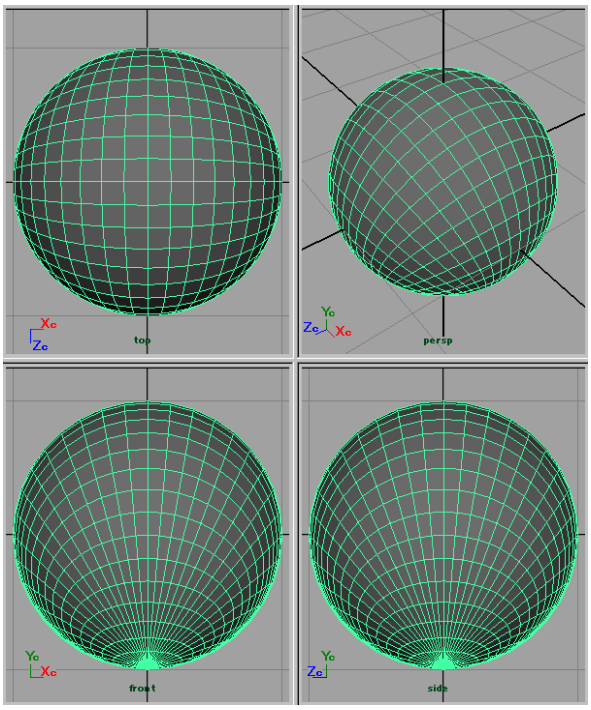

図 8: 履歴非依存型-単純回転方式 (単回転) の特性グリッド

8.3 単純回転方式 (複回転)

前節 (8.2) で述べた単純回転方式 (単回転) は，フリップ方向が 1 方向なので，従来方式よりも使い易くなっている。しかし，プレイヤがコントローラを振りかぶるような向きに持つと，アップベクトルが $-Y_c$ 方向となるためフリップが起こる。そこで，フリップ方向をずらして，フリップをさらに起こりにくくする。

アップベクトルが向く頻度が少ない方向をフリップ方向とするのが良い。コントローラ座標系において，このような方向の単位ベクトルを一つ決めて，これを $\mathbf{v}_r$ とする。$\mathbf{v}_t = -\mathbf{v}_r$ を目標ベクトル (target vector) とする。ワールド座標系において，次の手順によりコントローラの向き $\mathbf{M}$ を決める。

1. コントローラの向きをワールド座標系に合わせる (基準状態にする)。
2. アップベクトル $\mathbf{v}_u$ から目標ベクトル $\mathbf{v}_t$ への単純回転を，コントローラに作用させる。
3. 目標ベクトル $\mathbf{v}_t$ から $+Y_w$ ベクトルへの単純回転を，コントローラに作用させる。

$\mathbf{M}$ は次式で表される (要素の詳細は付録 A 参照)。

$$\mathbf{M} = \mathbf{M}_{SR}(\mathbf{v}_t,\, \mathbf{e}_1)\, \mathbf{M}_{SR}(\mathbf{v}_u,\, \mathbf{v}_t) \tag{27}$$

コントローラ座標系から見たワールドの向きは次式のようになる。

$$\mathbf{M}^{-1} = \mathbf{M}_{SR}(\mathbf{v}_u,\, \mathbf{v}_t)^{-1}\, \mathbf{M}_{SR}(\mathbf{v}_t,\, \mathbf{e}_1)^{-1} \tag{28}$$

$$= \mathbf{M}_{SR}(\mathbf{v}_t,\, \mathbf{v}_u)\, \mathbf{M}_{SR}(\mathbf{e}_1,\, \mathbf{v}_t) \tag{29}$$

この方式の特性グリッドは,「単純回転方式 (単回転)」の特性グリッドを $\mathbf{M}_{SR}(\mathbf{e}_1,\, \mathbf{v}_t)$ で回転させたものになる。例えば $\mathbf{v}_r$ を，図 9 のように，$-Y_c$ 方向から $+Z_c$ 方向に $\theta = 50°$ 回転させた方向に設定した場合，特性グリッドは図 10 のようになる。$\mathbf{v}_r$ 方向が特異点となることが分かる。この手法でフリップが起こるのは，$\mathbf{v}_u$ が $\mathbf{v}_r$ 方向を向く場合，即ちプレイヤがコントローラの $\mathbf{v}_r$ を真上に向けて保持した場合のみである。

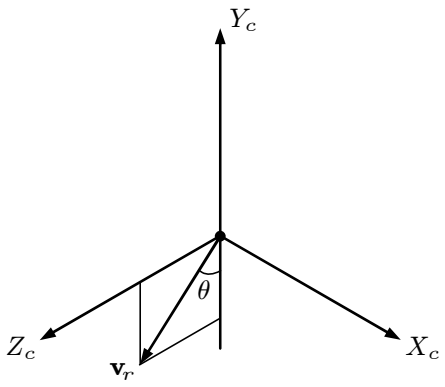

図 9: $\mathbf{v}_r$ の設定の一例

9 履歴依存型

履歴依存型のうち最も単純なモデルでは，問題を次のように捉えることができる。ある時点において，アップベクトルが $\mathbf{v}_p$ で，これに対応するコントローラの向き $\mathbf{M}_p$ が既に得られていると

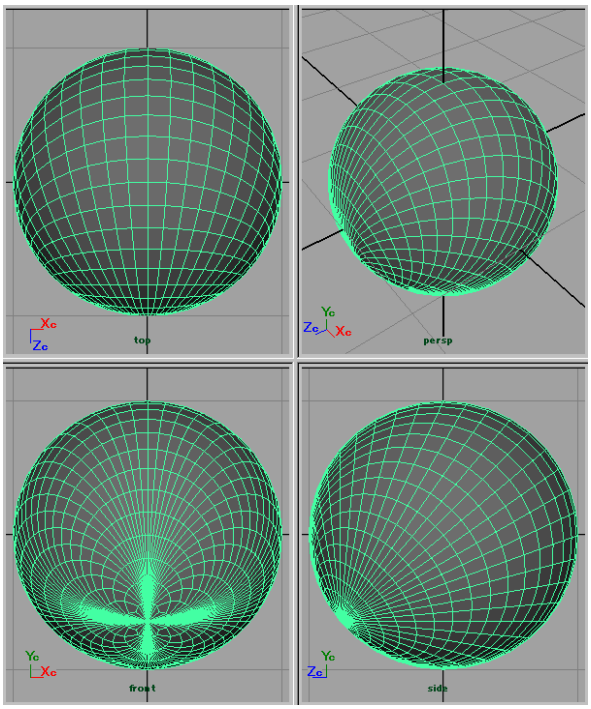

図 10: 履歴非依存型-単純回転方式 (複回転) の特性グリッド

する。プレイヤがコントローラを動かすのに伴って，新たなアップベクトル $\mathbf{v}_u$ が得られる。この $\mathbf{v}_u$ に対するコントローラの向き $\mathbf{M}$ を決める。

履歴依存型の最大の長所は，フリップ現象を回避できることである。履歴非依存型では，図 3 のように，球面 S 上の各点に対して X_w, Z_w の向きを一意に定めたが，履歴依存型では，X_w, Z_w の向きは履歴に依存し一意に定まらない。そのため，不動点定理の縛りを受けないので，フリップを起こさない向き決めが可能となる。

一方，短所となるのは，得られるコントローラの向きが履歴の影響でずれるため，意図した方向に向けるのが難しいことである。例えば，プレイヤがある状態からコントローラを動かし，再び元の状態に戻しても，得られるコントローラの向きは，履歴の影響を受けるため元に戻るとは限らない。

履歴依存型の一手法として，単純回転を用いる方法がある。この手法では，コントローラ座標系を基準として，ワールドに対して $\mathbf{v}_p$ から $\mathbf{v}_u$ への単純回転を掛ける。コントローラの向きは，以下のようにして求められる。

$$\mathbf{M}^{-1} = \mathbf{M}_{SR}(\mathbf{v}_p,\, \mathbf{v}_u)\, \mathbf{M}_p^{-1} \tag{30}$$

$$\therefore\ \mathbf{M} = \mathbf{M}_p\, \mathbf{M}_{SR}(\mathbf{v}_p,\, \mathbf{v}_u)^{-1} \tag{31}$$

$$= \mathbf{M}_p\, \mathbf{M}_{SR}(\mathbf{v}_u,\ \mathbf{v}_p) \tag{32}$$

10 ハイブリッド型

これまで述べてきたように，履歴非依存型ではフリップ現象を回避できない。一方，履歴依存型は，フリップ現象は回避できるものの，得られるコントローラの向きを意図した方向に向けにく

い。そこで，両者ををを融合させて，以下のようなハイブリッド型を作る。

球面 S 上の各点において，特性グリッドを基準として，X_w(または Z_w) が向くことのできる方向の範囲 (旋回範囲) を設定する。この旋回範囲の制約のもとで，履歴依存型の手法を用いて X_w(または Z_w) の向きを決める。処理手順は次のようになる。

1. 履歴依存型の手法を使って X_w, Z_w の向きを求める。
2. 求めた結果が旋回範囲内かどうかを調べる。
   (a) 旋回範囲内であれば X_w, Z_w をそのまま使う。
   (b) 旋回範囲外であれば，旋回範囲内に収まるように X_w, Z_w を修正する。

旋回範囲を設定する際には，次の 2 点に留意する。

- 特異点の解消
  特性グリッドの特異点付近では，向き制限なし (旋回範囲が全方位) に設定する。この設定により，特異点付近では，履歴依存型の向き決めのみによって X_w, Z_w が決まるので，フリップ問題が解消される。

- 連続性の確保
  球面 S 上の位置に対して，旋回範囲が連続的に変化するように設定する。この設定により，向き制限なしの領域から向き制限された領域へとアップベクトルが移動した際に，X_w, Z_w の向きが連続的に変化する。

10.1 ハイブリッド型の実装例 (その 1)

従来手法 (前後方向を維持) の特性グリッドを用いる。この特性グリッドは，図 6 のように，地球儀の緯線・経線に相当する形状をしており，北極・南極に相当する $+Z_c, -Z_c$ 方向が特異点となっている。この特性グリッドをベースとし，図 11 のように，球面 S 上の各点に対して旋回範囲を設定する。図中の扇型のうち青色 (色) の部分が $+Z_w$ の向くことのできる範囲である。赤道上では，$+Z_w$ 方向を特性グリッドの向き ($+Z_c$ の向きと同じ) に拘束する。緯度に比例して旋回範囲を広げ，緯度 λ 以上の領域では向き制限なし (旋回範囲が全方位) とする。(図 11 では $\lambda = \pi/3$ とした。)

この設定により，コントローラの向き決めは次のような挙動となる。

- $\mathbf{v}_u$ が X_cY_c 平面内にある場合，即ちプレイヤがコントローラの Z_c 軸を水平に保持した場合には，履歴に依存せず，$+Z_w$ は $+Z_c$ と一致する。

- $\mathbf{v}_u$ が $\pm Z_c$ 方向を向く場合，即ちプレイヤがコントローラの Z_c 軸を垂直に保持した場合には，Z_w の向きは履歴によって決まり，フリップは起こらない。

アップベクトル $\mathbf{v}_u$ に対して，コントローラの向き $\mathbf{M}$ を求める処理は，以下のようになる。

10.1.1 履歴依存型による向き決め

まず，9 節の履歴依存型-単純回転 (単回転) の手法でコントローラの向きを求め，これを $\mathbf{M}_d$ とする。$\mathbf{M}_d$ の要素は次式のとおりとする。コントローラの向きが $\mathbf{M}_d$ のとき，コントローラ座標系における $+X_w, +Z_w$ の向きは，それぞれ $\mathbf{v}_d, \mathbf{v}_e$ となる。

$$\mathbf{M}_d \equiv \begin{bmatrix} \mathbf{v}_d^{\mathrm{T}} \\ \mathbf{v}_u^{\mathrm{T}} \\ \mathbf{v}_e^{\mathrm{T}} \end{bmatrix} \tag{33}$$

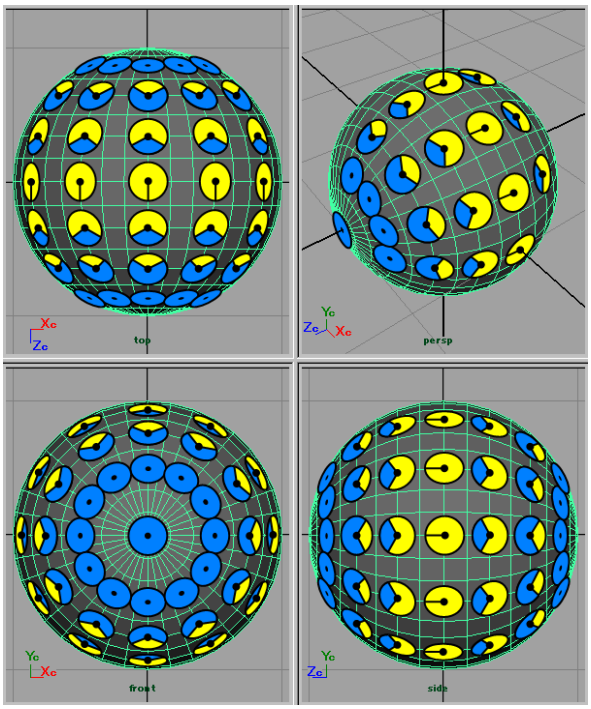

図 11: ハイブリッド型の実装例 (その 1)

ここで，もし $|u_z| \geq \sin\lambda$ であれば，アップベクトルは球面 S 上で向き制限なしの領域にあるので，$\mathbf{M} = \mathbf{M}_d$ として終了となる。そうでない場合は，処理は以下に続く。

10.1.2 履歴非依存型による向き決め

4.2 節の履歴非依存型 (前後方向を維持) の手法でコントローラの向きを求め，これを $\mathbf{M}_g$ とする。$\mathbf{M}_g$ の要素は次式のとおりとする。$\mathbf{v}_g, \mathbf{v}_h$ は，それぞれ特性グリッドから決まる $+X_w, +Z_w$ 方向に相当する。

$$\mathbf{M}_g \equiv \begin{bmatrix} \mathbf{v}_g^{\mathrm{T}} \\ \mathbf{v}_u^{\mathrm{T}} \\ \mathbf{v}_h^{\mathrm{T}} \end{bmatrix} \tag{34}$$

旋回範囲の大きさ ρ (単位はラジアン) を，次式により設定する。

$$\rho = \frac{\pi}{\lambda} \left| \sin^{-1} u_z \right| \tag{35}$$

図 12 のように，$+Z_w$ が向くことのできる範囲 ($+Z_w$ の旋回範囲) は，$\mathbf{v}_h$ から $\pm\rho$ の範囲となる。この ρ を使って，$\mathbf{M}_d$ の向きが旋回範囲内にあるかどうかを調べる。もし $\mathbf{v}_e \cdot \mathbf{v}_h \geq \cos\rho$ であれば，旋回範囲内にあるので，$\mathbf{M} = \mathbf{M}_d$ として終了となる。そうでない場合は，処理は以下に続く。

10.1.3 旋回範囲内への修正

旋回範囲外にある $\mathbf{v}_e$ を修正し，旋回範囲内に収まるようにする。このとき，図 13 のように，$\rho, -\rho$ のうちいずれか近い方に修正する。もし $\mathbf{v}_e \cdot \mathbf{v}_g \geq 0$ であれば $\rho' = \rho$ とし，そうでなければ

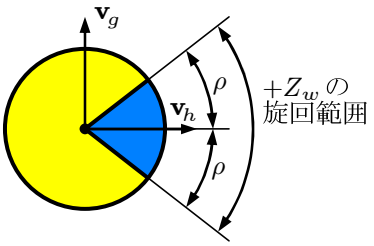

図 12: $+Z_w$ の旋回範囲

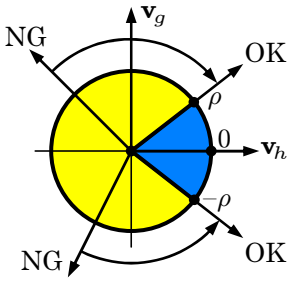

図 13: 旋回範囲外から旋回範囲内への修正

$\rho' = -\rho$ とする。$\mathbf{v}_e$ の修正結果は，$\mathbf{v}_h$ を $\mathbf{v}_u$ 軸回りに ρ' だけ回転させたものになる。次式のように回転マトリクス $\mathbf{M}_r$ を作る。

$$\mathbf{M}_r = \begin{bmatrix} \cos\rho' & 0 & \sin\rho' \\ 0 & 1 & 0 \\ -\sin\rho' & 0 & \cos\rho' \end{bmatrix} \tag{36}$$

コントローラの向き $\mathbf{M}$ は，以下のようにして得られる。

$$\mathbf{M}^{-1} = \mathbf{M}_g^{-1}\mathbf{M}_r \tag{37}$$

$$\therefore\ \mathbf{M} = \mathbf{M}_r^{-1}\mathbf{M}_g \tag{38}$$

10.2 ハイブリッド型の実装例 (その 2)

単純回転方式 (単回転) の特性グリッドを用いる。この特性グリッドは，図 8 のように，$-Y_c$ 方向のみが特異点となる。この特性グリッドをベースとし，図 14 のように，球面 S 上の各点に対して旋回範囲を設定する。図中の扇型のうち青色 (■色) の部分が $+Z_w$ の向くことのできる範囲である。球面 S と $+Y_c$ との交点を P_1 とする。点 P_1 では，$+Z_w$ 方向を特性グリッドの向き ($+Z_c$ の向きと同じ) に拘束する。点 P_1 からの距離 (球面 S 上での距離とする) に比例して旋回範囲を広げ，距離 λ 以上の領域では向き制限なし (旋回範囲が全方位) とする。(図 14 では $\lambda = \pi/2$ とした。)

この設定により，コントローラの向き決めは次のような挙動となる。

- $\mathbf{v}_u$ が $+Y_c$ 方向を向く場合，即ちプレイヤがコントローラの $+Y_c$ を真上に向けた場合には，履歴に依存せず，$+Z_w$ は $+Z_c$ と一致する。
- $\mathbf{v}_u$ が $-Y_c$ 方向を向く場合，即ちプレイヤがコントローラの $+Y_c$ を真下に向けた場合には，Z_w の向きは履歴によって決まり，フリップは起こらない。

アップベクトル $\mathbf{v}_u$ に対して，コントローラの向き $\mathbf{M}$ を求める処理は，以下のようになる。

10.2.1 履歴依存型による向き決め

まず，9 節の履歴依存型-単純回転 (単回転) の手法でコントローラの向きを求め，これを $\mathbf{M}_d$ とする。$\mathbf{M}_d$ の要素は式 (33) と同様とする。ここで，もし $u_y \le \cos\lambda$ であれば，アップベクトルは球面 S 上で向き制限なしの領域にあるので，$\mathbf{M} = \mathbf{M}_d$ として終了となる。そうでない場合は，処理は以下に続く。

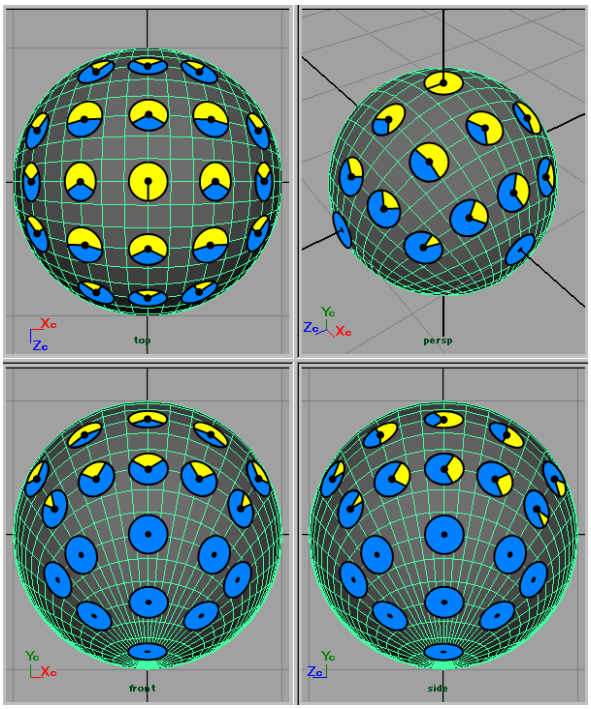

図 14: ハイブリッド型の実装例 (その 2)

10.2.2 履歴非依存型による向き決め

8.2 節の履歴非依存型-単純回転方式 (単回転) の手法でコントローラの向きを求め，これを $\mathbf{M}_g$ とする。$\mathbf{M}_g$ の要素は式 (34) と同様とする。旋回範囲の大きさ ρ (単位はラジアン) を，次式により設定する。

$$\rho = \frac{\pi}{\lambda} \cos^{-1} u_y \tag{39}$$

$+Z_w$ の旋回範囲は，10.1 節の実装例 (その 1) と同様に，図 12 のようになる。もし $\mathbf{v}_e \cdot \mathbf{v}_h \geq \cos\rho$ であれば，旋回範囲内にあるので，$\mathbf{M} = \mathbf{M}_d$ として終了となる。そうでない場合は，処理は以下に続く。

10.2.3 旋回範囲内への修正

旋回範囲外にある $\mathbf{v}_e$ を修正し，旋回範囲内に収まるようにする。処理内容は，実装例 (その 1) の 10.1.3 節と同様である。

11 まとめ

コントローラの向き決め方法は，履歴非依存型と履歴依存型の 2 つに分類できる。さらに両者を融合させたハイブリッド型を作ることができる。それぞれの特徴は次のとおりである。

履歴非依存型

向き決めの挙動を特性グリッドで表すことができる。フリップ現象は回避できない。しかし，

フリップ方向を 2 方向から 1 方向に減らすことは可能で，さらにフリップ方向を意図した方向に設定することもできる。

履歴依存型

フリップ現象を回避できる。履歴に依存するため，向きのずれが生じる (元の向きに戻らない)。

ハイブリッド型

フリップ現象を回避できる。向きのずれを矯正できる。

ハイブリッド型は，履歴非依存型と履歴依存型のそれぞれの欠点を取り除いたものになっている。しかし，ハイブリッド型と言えども万能ではない。そもそも，現実のコントローラの向きを知る術がない以上，オールマイティの方法は存在しない。個々の方法の特徴を掴んだ上で，ゲームシーンに合った選択をすることが望ましい。

付録 A　単純回転マトリクスの詳細

単純回転マトリクスの要素の詳細を以下に記す。

A.1　$\mathbf{M}_{SR}(\mathbf{v}_u,\, \mathbf{e}_1)$ の詳細

8.2 節の $\mathbf{M}_{SR}(\mathbf{v}_u,\, \mathbf{e}_1)$ の要素は次のようになる。

$$\mathbf{M}_{SR}(\mathbf{v}_u,\, \mathbf{e}_1) = \begin{bmatrix} K_1 u_z^2 + u_y & -u_x & -K_1 u_x u_z \\ u_x & u_y & u_z \\ -K_1 u_x u_z & -u_z & K_1 u_x^2 + u_y \end{bmatrix} \tag{40}$$

ただし，K_1 は以下のとおりである。

$$K_1 = \frac{1 - u_y}{u_x^2 + u_z^2} \tag{41}$$

次式を利用すると計算式を簡略化できるが，$u_y \approx -1$ のときに桁落ちによる計算精度の劣化を招くので，このような簡略化は使うべきではない。

$$K_1 = \frac{1 - u_y}{u_x^2 + u_z^2} = \frac{1 - u_y}{1 - u_y^2} = \frac{1}{1 + u_y} \tag{42}$$

A.2　$\mathbf{M}_{SR}(\mathbf{v}_t,\, \mathbf{e}_1)\, \mathbf{M}_{SR}(\mathbf{v}_u,\, \mathbf{v}_t)$ の詳細

8.3 節において，$\mathbf{v}_r$ を図 9 のように設定した場合，即ち $\mathbf{v}_t = [0\ \cos\theta\ -\sin\theta]^{\mathrm{T}}$ の場合，$\mathbf{M}_{SR}(\mathbf{v}_t,\, \mathbf{e}_1)\, \mathbf{M}_{SR}(\mathbf{v}_u,\, \mathbf{v}_t)$ の要素は次のようになる。

$$\mathbf{M}_{SR}(\mathbf{v}_t,\, \mathbf{e}_1)\, \mathbf{M}_{SR}(\mathbf{v}_u,\, \mathbf{v}_t) = \begin{bmatrix} K_2 u_1^2 + u_0 & -K_2 u_1 u_s - u_c & -K_2 u_1 u_c + u_s \\ u_x & u_y & u_z \\ -K_2 u_1 u_x & K_2 u_x u_s - u_z & K_2 u_x u_c + u_y \end{bmatrix} \tag{43}$$

ただし，u_0, u_1, u_c, u_s, K_2 は以下のとおりである。

$$u_0 = u_y \cos\theta - u_z \sin\theta \tag{44}$$

$$u_1 = u_y \sin\theta + u_z \cos\theta \tag{45}$$

$$u_c = u_x \cos\theta \tag{46}$$

$$u_s = u_x \sin\theta \tag{47}$$

$$K_2 = \frac{1 - u_0}{u_x^2 + u_1^2} \tag{48}$$

次式を利用すると計算式を簡略化できるが，$u_0 \approx -1$ のときに桁落ちによる計算精度の劣化を招くので，このような簡略化は使うべきではない。

$$K_2 = \frac{1 - u_0}{u_x^2 + u_1^2} = \frac{1 - u_0}{1 - u_0^2} = \frac{1}{1 + u_0} \tag{49}$$

参考文献

[1] (株) バンダイナムコゲームス. 加来 量一『プログラム、情報記憶媒体及び画像生成システム』公開特許公報 特開 2007-267851 (2007-10-18)