

6-5

ベクトルとその演算

Keyword

有向線分

内積

外積

基底ベクトル

本節では、位置や速度、図形や座標変換など、ゲーム数学でさまざまなものを表すのに用いられるベクトルについて解説します。

ベクトルとは？

とは、長さや方向を持つ量で、一般にスカラー（ただの数）を複数組み合わせることで表現されます。

例えば、2つのスカラーを組み合わせで作られるのが2次元ベクトルで、2D空間（平面）上の長さや方向を持つ量です。また、3つのスカラーを組み合わせで作られるのが3次元ベクトルで、3D空間中の長さや方向を持つ量です。

同様に4次元ベクトルも考えることができ、ゲームにおいては（297ページ）というものや（349ページ）と呼ばれる量を考えるときに使います。

有向線分との違い

ただし、ベクトルはとは違います。有向線分とは、とがあるのことですが、ベクトルには始点とは関係なく、平行移動したベクトルはすべて同じものとされます。

このことは、例えば2Dの有向線分を表現するには、始点と終点それぞれの xy 座標、つまり4つのスカラーが必要になるのに対して、2次元ベクトルを表現するには2つのスカラーしか必要ないことを考えてもわかると思います。

ベクトルは、例えば空間上の位置を表したり（）、速度や加速度などの、あるいは、直線や円などの図形、また回転などのを表現するのにも使われます。そのように、ゲームプログラミングにおいてベクトルは（特に3Dでは）非常に重要なものといえます。

ベクトルを数式上で表現する

具体的にベクトルを数式上に表現するときには、いくつかの表現方法があります。まず、変数名を太字で a などとした場合には、その変数はベクトルだという意味になります。

NOTE

同じような目的の表現方法で、変数名の上に矢印を付けた \vec{a} という表現も見かけますが、これは大学レベル以上の数学ではあまり使いません。

本書でも、ベクトルの変数については \boldsymbol{a} などと太字で表現することにします。

また、ベクトルを成分、つまりいくつかのスカラーの組み合わせで表示するときには、いくつかの書き方の流儀があります。例えば、3次元ベクトルを例とすれば、

- ・ 3D空間での座標表示と同じ (x, y, z) (x, y, z はスカラー) と表現する方法
- ・ 行ベクトルと呼ばれる と表現する方法
- ・ あるいは列ベクトルと呼ばれる と表現する方法

があります。

ベクトルの属性

ベクトルには、以下のような属性があります。

絶対値 (ノルム)

ベクトルの は、 とも呼ばれ、以下のように定義されます。

2次元ベクトル $\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \end{pmatrix}$ の絶対値は

$$|\boldsymbol{a}| = \text{$$

となり、3次元ベクトル $\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$ の絶対値は

$$|\boldsymbol{a}| = \text{$$

となります。

これはピタゴラスの定理の式ですから、 が絶対値、ということになります。普通の数であるスカラーでも、数直線上での原点からの長さが絶対値ですから、ベクトルでも長さが絶対値、というのは自然なことといえるでしょう。

✦ 単位ベクトル

とは、のベクトルのことです。

2次元ベクトル $\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \end{pmatrix}$ の場合は、

$$|\boldsymbol{a}| = \text{} = \text{$$

のときが単位ベクトル、3次元ベクトル $\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$ の場合は、

$$|\boldsymbol{a}| = \text{} = \text{$$

のときに単位ベクトルになります。

しばしば、単位ベクトルであることを示すために、ベクトルの変数名に $\hat{}$ （ハット）を付けて $\hat{\boldsymbol{a}}$ 、 $\hat{\boldsymbol{b}}$ などと表現されます。この単位ベクトルは、ゲームプログラムでも基準となるベクトルなどとして多用されるので覚えておきましょう。

📦 ベクトルの基本的な演算

また、普通の数であるスカラーと同じように、ベクトルにはいくつかの演算があります。以下に、そのうちでも基本的な演算をいくつか解説します。

✦ 加算（足し算）

2つの2次元ベクトル $\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \end{pmatrix}$ と $\boldsymbol{b} = \begin{pmatrix} b_x \\ b_y \end{pmatrix}$ の足し算は、

$$\boldsymbol{a} + \boldsymbol{b} = \begin{pmatrix} a_x \\ a_y \end{pmatrix} + \begin{pmatrix} b_x \\ b_y \end{pmatrix} = \text{$$

となり、2つの3次元ベクトル $\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$ と $\boldsymbol{b} = \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix}$ の足し算は

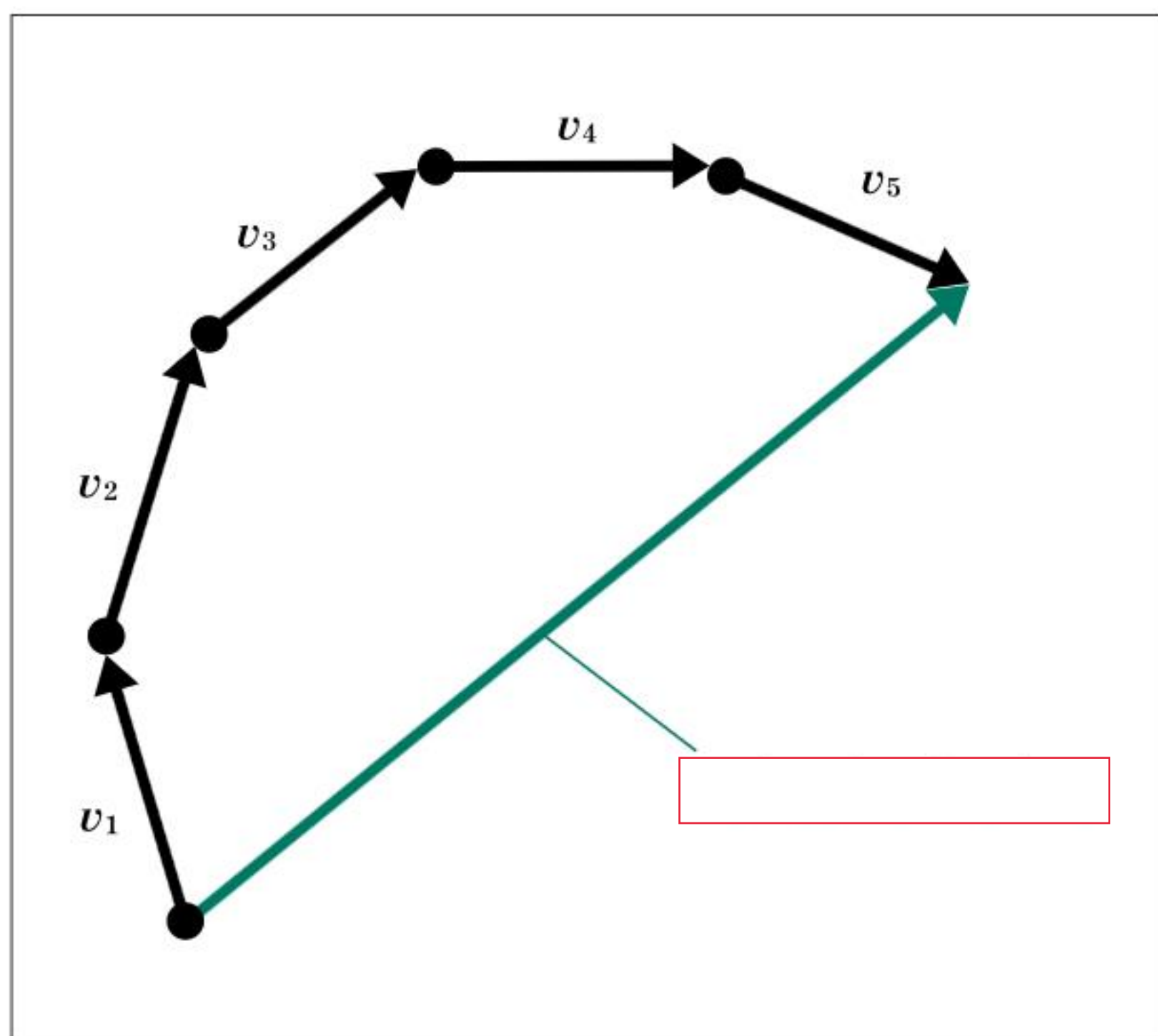
$$\boldsymbol{a} + \boldsymbol{b} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} + \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = \text{$$

となります。

つまり、 x は x 同士、 y は y 同士と、同じ成分同士を足し算した結果を、ベクトルの足し算とします。これについては「なぜ？」と思ってはいけません。そういうふう決められている、つま

り定義されているのです。

ちなみに、図で示すと、ベクトルの足し算は、各ベクトルの終点と始点をつなぎ合わせていったものになります (図6-5-1)。



● 図6-5-1 5つのベクトル ($v_1 \sim v_5$) ベクトルの足し算

✦ 減算 (引き算)

2つの2次元ベクトル $\mathbf{a} = \begin{pmatrix} a_x \\ a_y \end{pmatrix}$ と $\mathbf{b} = \begin{pmatrix} b_x \\ b_y \end{pmatrix}$ の引き算は、

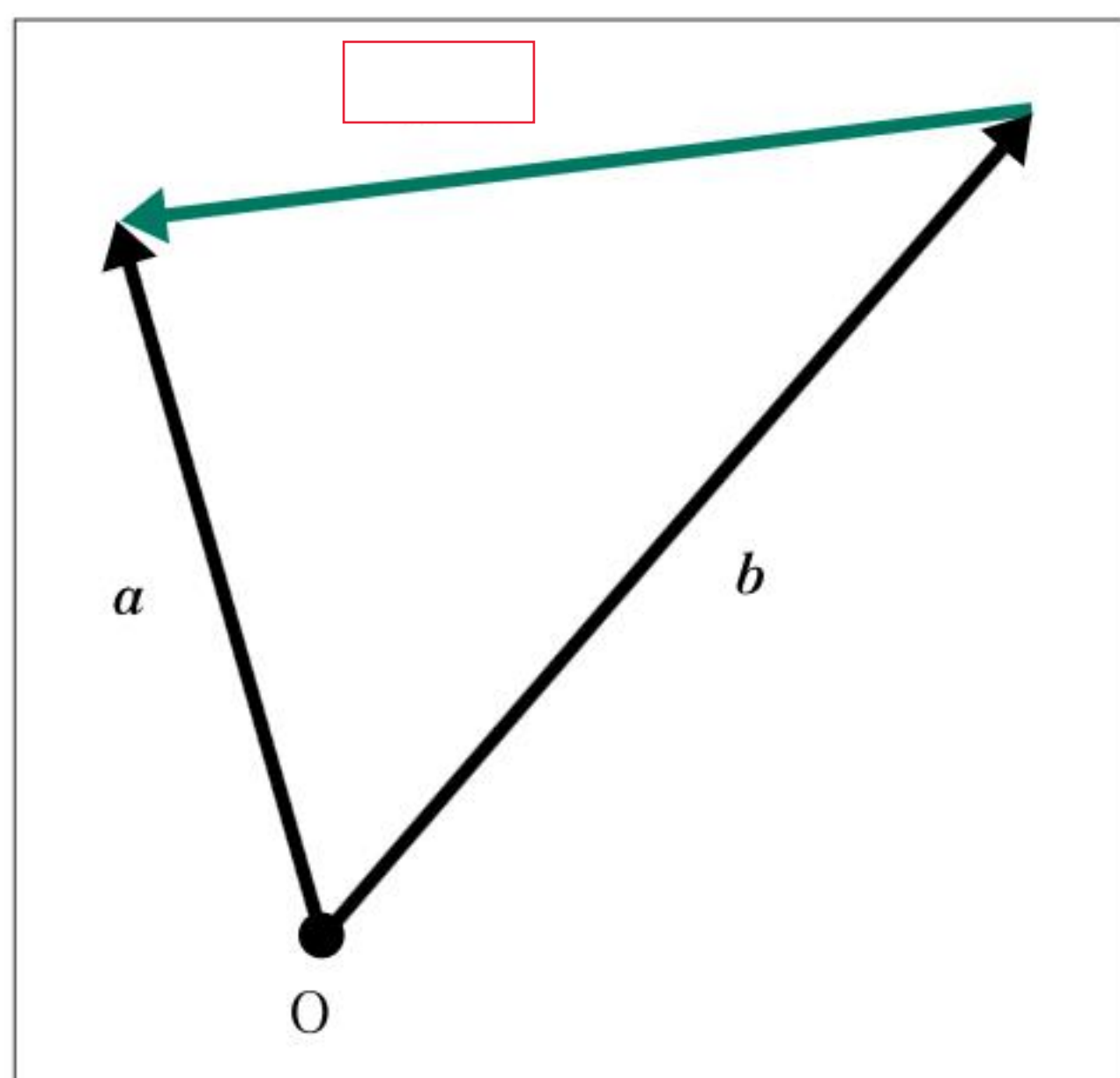
$$\mathbf{a} - \mathbf{b} = \begin{pmatrix} a_x \\ a_y \end{pmatrix} - \begin{pmatrix} b_x \\ b_y \end{pmatrix} = \boxed{}$$

と定義され、また、2つの3次元ベクトル $\mathbf{a} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$ と $\mathbf{b} = \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix}$ の引き算は、

$$\mathbf{a} - \mathbf{b} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} - \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = \boxed{}$$

と定義されています。つまり、同じ成分同士を引き算した結果が、ベクトルの引き算となるのです。

図形的には、 $\mathbf{a} - \mathbf{b}$ というベクトルは、ベクトル \mathbf{b} の先端からベクトル \mathbf{a} の先端へと向かうベクトルになります (図6-5-2)。



● 図 6-5-2 ベクトルの引き算 ($a - b$)

このことから、 $a - b = c$ とすると、

$$a = b + c$$

となりますから、ベクトルで構成された式でも、が普通にできることがわかります。

これは、図形的に考えても、成分的に考えてもわかることでしょう。この性質から、ベクトルの足し算・引き算の部分については、普通の式と同じように式変形をすることができるのです。

✚ 定数（スカラー）倍

2次元ベクトル $\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \end{pmatrix}$ に定数 α （スカラー）を掛けると、

$$\alpha \boldsymbol{a} = \alpha \begin{pmatrix} a_x \\ a_y \end{pmatrix} = \text{$$

となり、同様に、3次元ベクトル $\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$ (a_x, a_y, a_z はスカラー) に定数 α （スカラー）を掛けると

$$\alpha \boldsymbol{a} = \alpha \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} = \text{$$

となります。

つまり、ベクトルの定数倍とは、ベクトルのすべての成分にその定数を掛けたものです。これもやはり定義ですから、「なぜ？」と思っははいけません。そう決められているので、覚えましょう。

❖ 一次結合と線形補間

ベクトルの は、上で述べたベクトルの定数倍と、ベクトルの加算・減算を組み合わせることによって、以下のように表されます。

$$p = \alpha a + \beta b$$

ここで、 α と β は定数です。この α と β を制御することで、元になるベクトル a とベクトル b から、さまざまな新しいベクトルを作り出すことができます。

例えば、 $\alpha + \beta = 1$ の場合を考えます。このとき、 $\beta = t$ と置き換えると、 $\alpha + \beta = 1$ から $\alpha = 1 - t$ となるため、

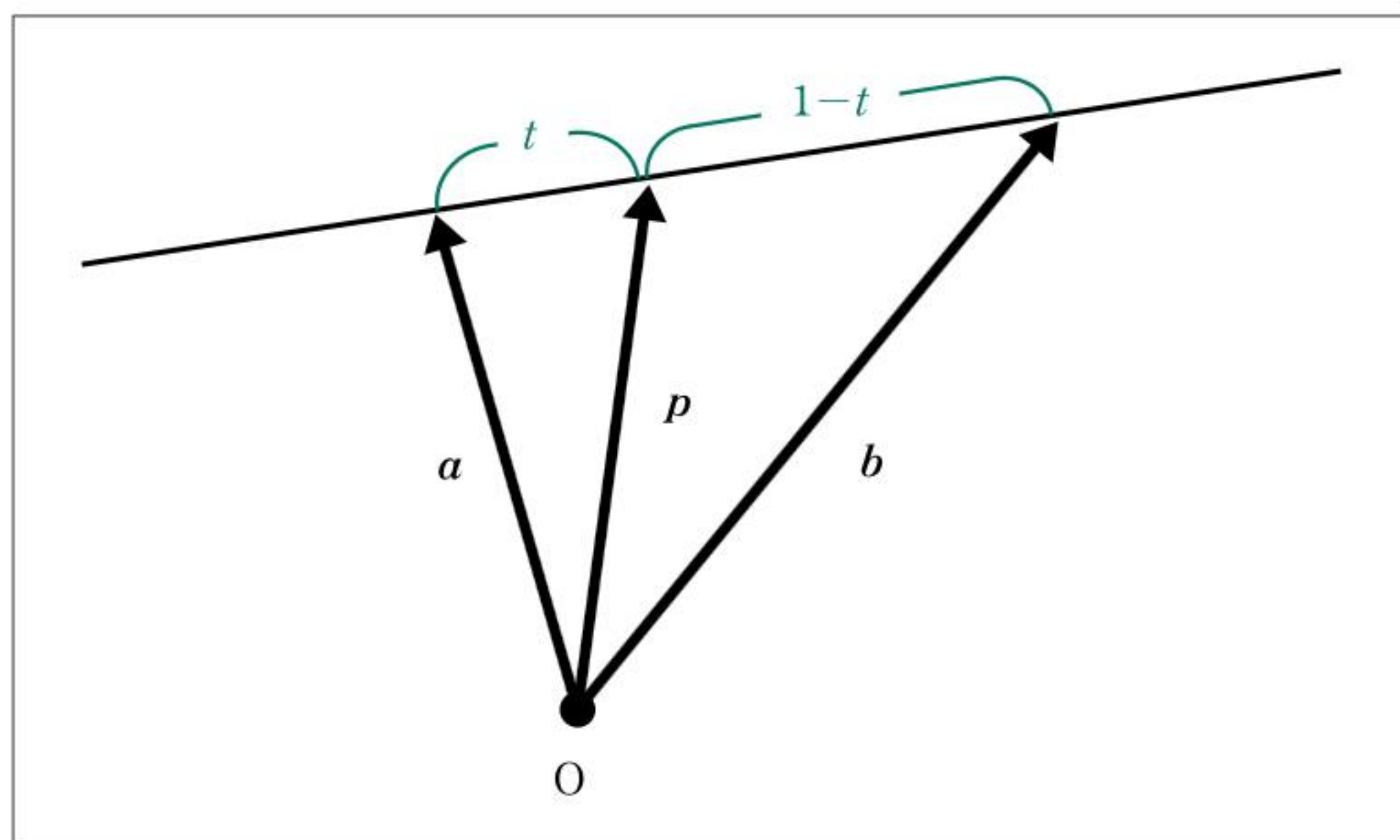
$$p = \text{$$

となります。これを变形すれば

$$p = a + t(\text{$$

となるため、ベクトル p は、ベクトル を起点として、ベクトル の方向（あるいは、 ならばその逆方向）に延びるベクトルになります。

このときベクトル p は、 $t = 0$ であればベクトル a と一致し、 $t = 1$ であればベクトル b と一致するため、 t を まで変化させれば、ベクトル a とベクトル b の間を ようなベクトルを作ることができます（図6-5-3）。



▶ 図6-5-3 ベクトル a とベクトル b の間を直線的につなぐベクトル p

この場合、ベクトル a とベクトル b の間を $t : 1 - t$ に内分することから、

$$p = \text{} \quad (0 \leq t \leq 1)$$

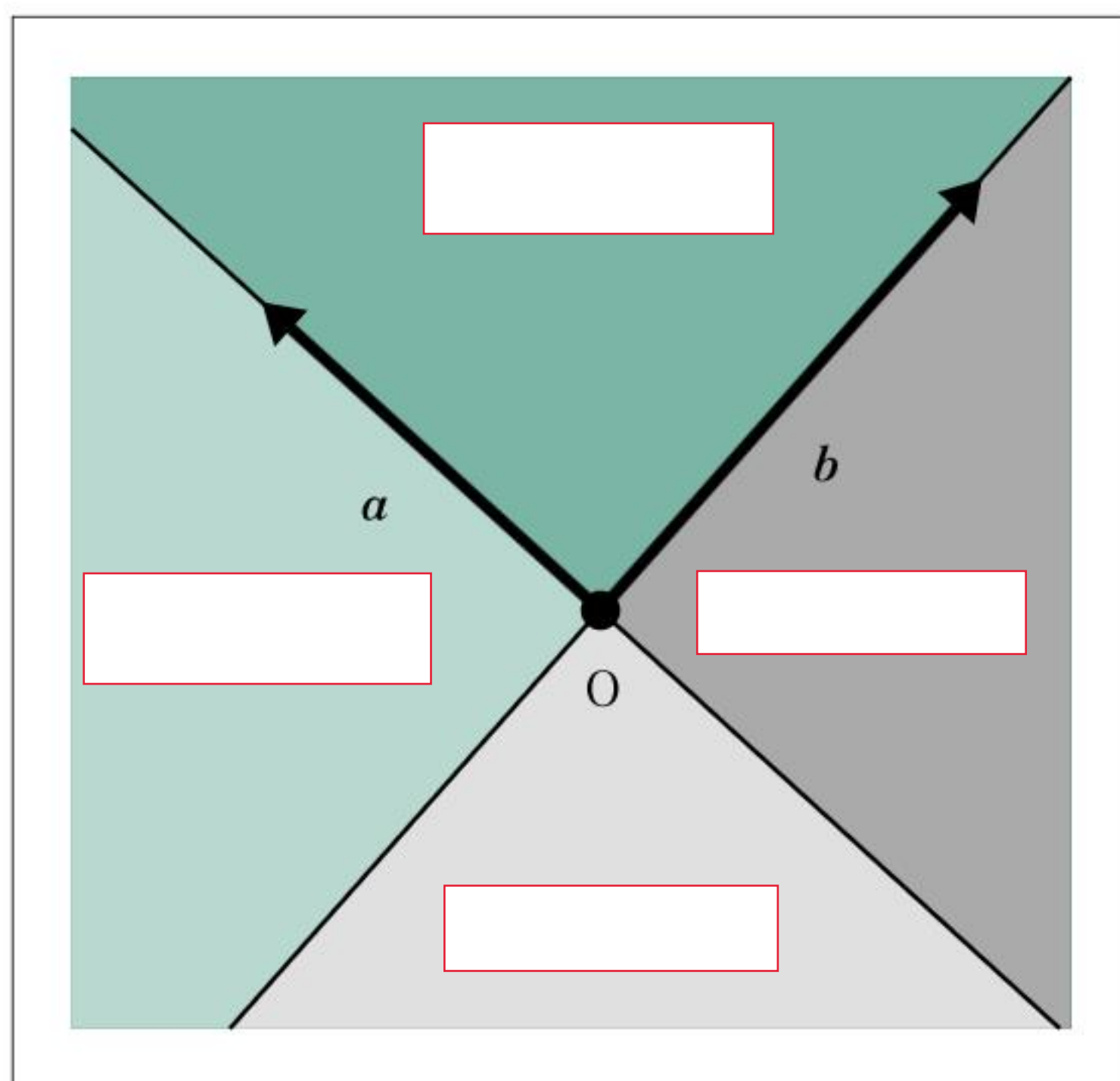
という式を、ベクトルの内分の公式といいます。ゲームプログラムでは例えば、この公式での t を時間と解釈すれば、位置 a と位置 b だけがわかっていて、それらの間に物体を移動させる場合

に、位置 a 、位置 b 間の中間の位置を計算することができ、それは と呼ばれてよく使われます。

また、ベクトルの一次結合

$$p = \alpha a + \beta b$$

では、 α と β の符号によって、ベクトル p がベクトル a 、 b とどのような位置関係にあるかを判定することができます。2次元ベクトルを例に、その様子を図6-5-4に示しました。



● 図6-5-4 α 、 β それぞれの符号による、一次結合ベクトルの位置関係

ゲームでは、ポリゴンなどの自由な形の多角形に対する当たり判定を取るときに、当たっているかどうかだけでなく、その多角形のどちら側に外れているのかまで知りたい場合に便利です。

内積

はベクトル同士の掛け算の一種で、2つのベクトルから1つのスカラー、つまりただの数を得ます。ベクトル a とベクトル b の内積は $a \cdot b$ と表現されるため、英語では Dot product と呼ばれ、具体的には、

$$a \cdot b = \text{$$

と定義されています。なお、 θ はベクトル a とベクトル b のなす角のことです (ただし $0 \leq \theta \leq \pi$)。

このままでは計算するのも大変そうですが (特に、2ベクトルのなす角 θ は求めにくい)、この内積については、2D で $a = (a_x \ a_y)$ 、 $b = (b_x \ b_y)$ とすると

$$a \cdot b = \text{$$

となり、また 3D で $a = (a_x \ a_y \ a_z)$ 、 $b = (b_x \ b_y \ b_z)$ とすると

$$a \cdot b = \boxed{}$$

となることが知られています。

つまり、

- ・ $\boxed{}$ 成分同士、 $\boxed{}$ 成分同士、 $\boxed{}$ 成分同士の $\boxed{}$ を行ったあとで
- ・ それらをすべて $\boxed{}$

ということをすればよく、内積を求めるだけなら、この式を使って簡単確実にできるというわけです。

さてここで、例えば3Dの式が成り立つためには

$$\boxed{} = \boxed{}$$

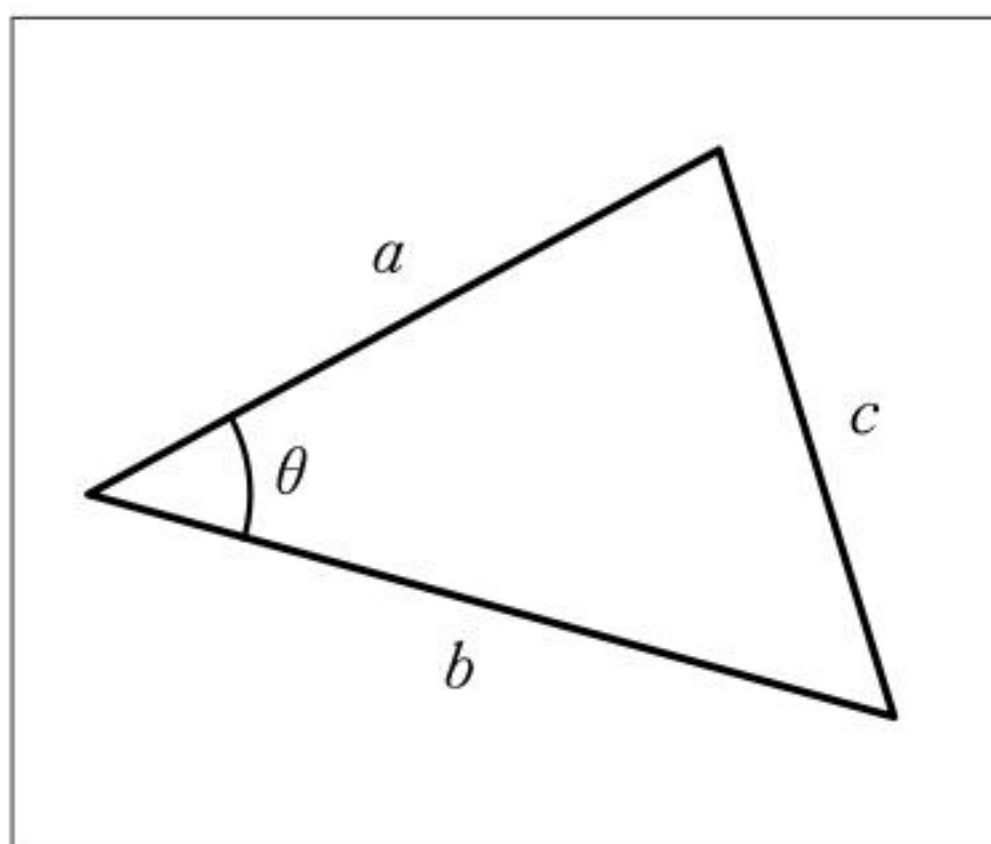
が成り立たなければなりませんから、この式の証明を行ってみましょう。

✦ 余弦定理とは？

それには、三角形の^{よげんていり}余弦定理というものを使うのが便利です。余弦定理とは、三辺の長さがそれぞれ a 、 b 、 c で図6-5-5に示した角度が θ のとき

$$\boxed{} = \boxed{}$$

となる、という定理です。

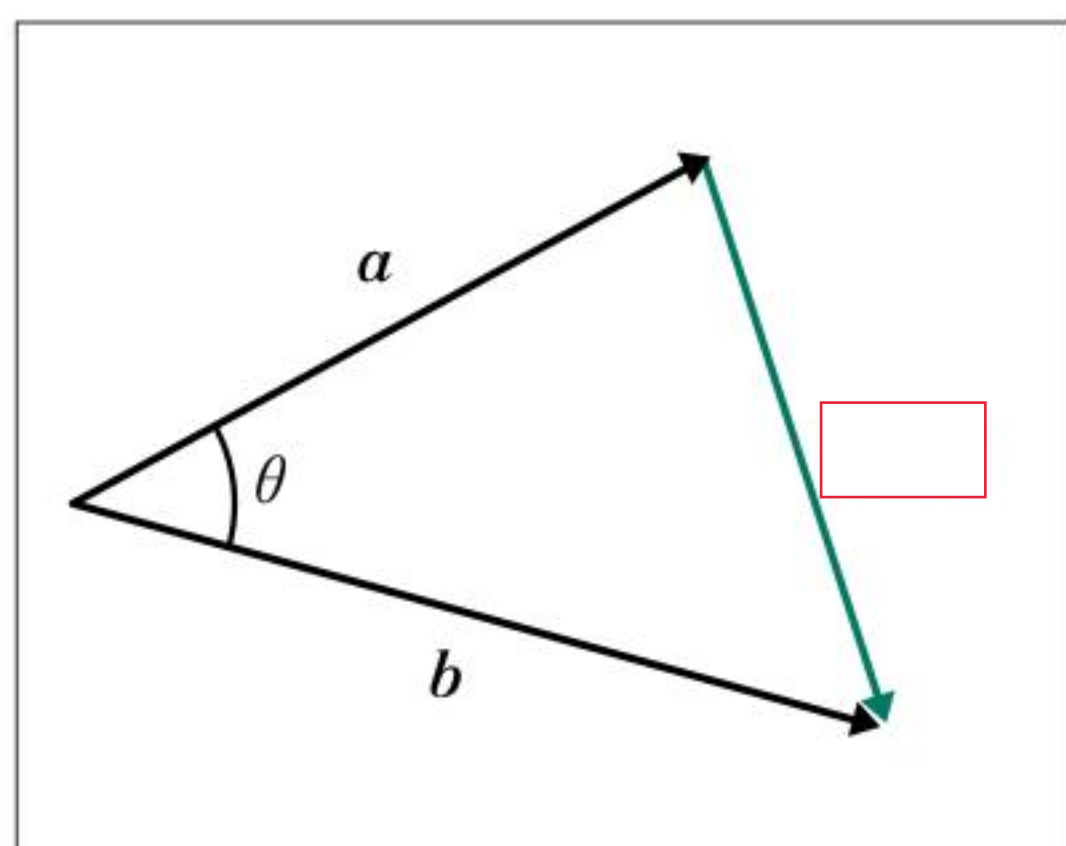


● 図6-5-5 a 、 b 、 c の三辺からなる三角形と、辺 a と辺 b がなす角 θ

この角度 θ が $\frac{\pi}{2}$ のときには、 $\cos \theta = 0$ となってピタゴラスの定理に一致することに注意しましょう。

✦ 余弦定理を使った証明

さて、ベクトル a 、ベクトル b 、ベクトル $b - a$ を三辺とする三角形（図6-5-6）に上記の余弦定理を当てはめてみましょう。すると、



● 図 6-5-6 a 、 b の 2 つのベクトルと、なす角 θ

$$|b-a|^2 = |a|^2 + |b|^2 - 2|a||b|\cos\theta$$

となりますが、ここで、ピタゴラスの定理から、

$$\begin{aligned} |a|^2 &= \boxed{} \\ |b|^2 &= \boxed{} \\ |b-a|^2 &= \boxed{} \end{aligned}$$

となり、これらを上式に代入すると

$$(b_x - a_x)^2 + (b_y - a_y)^2 + (b_z - a_z)^2 = a_x^2 + a_y^2 + a_z^2 + b_x^2 + b_y^2 + b_z^2 - 2|a||b|\cos\theta$$

となります。

ここで、左辺を展開してみると

$$\begin{aligned} (b_x - a_x)^2 + (b_y - a_y)^2 + (b_z - a_z)^2 &= b_x^2 - 2b_x a_x + a_x^2 + b_y^2 - 2b_y a_y + a_y^2 + b_z^2 - 2b_z a_z + a_z^2 \\ &= a_x^2 + a_y^2 + a_z^2 + b_x^2 + b_y^2 + b_z^2 - 2b_x a_x - 2b_y a_y - 2b_z a_z \end{aligned}$$

となります。

さて、この結果から改めて元の式を見てみると、両辺に $a_x^2 + a_y^2 + a_z^2 + b_x^2 + b_y^2 + b_z^2$ という共通項が現れていますから、それらを両辺から消去すると

$$\begin{aligned} -2b_x a_x - 2b_y a_y - 2b_z a_z &= -2|a||b|\cos\theta \\ -2(a_x b_x + a_y b_y + a_z b_z) &= -2|a||b|\cos\theta \\ \therefore a_x b_x + a_y b_y + a_z b_z &= |a||b|\cos\theta \end{aligned}$$

となります。これで、示したかった関係

$$|a||b|\cos\theta = \boxed{}$$

が成り立つことが証明できたこととなります (2D の場合でも同様です)。よって、内積 $a \cdot b$ については、

$$a \cdot b = \boxed{}$$

でもあり、また

$$a \cdot b = \boxed{}$$

でもあるわけです。

✚ 内積の使い道①：2つのベクトルのなす角度（とコサイン）を求められる

さて、問題は、この内積というものが何の役に立つのか、ということです。それを考えるためには、まず

$$a \cdot b = a_x b_x + a_y b_y + a_z b_z$$

であることから、ベクトル a と b との内積は、2つのベクトルの成分から確実に、しかも簡単に計算できる、という性質を押さえておきましょう。

そして、

$$a \cdot b = |a| |b| \cos \theta$$

でもあることから、

- ・ 確実に $\boxed{}$ を使い
- ・ 2つのベクトルの $\boxed{}$ として、 $\boxed{}$ を計算することができる

という重要性が見えてきたことでしょう。

実際、上記の式から

$$\cos \theta = \boxed{}$$

となるため、 $|a| \neq 0$ かつ $|b| \neq 0$ でさえあれば、どんな状況であっても2ベクトルのなす角 θ のコサインを得ることができるといえます。2ベクトルの $\cos \theta$ を計算したい場合というのはよくあることですから、内積のこの性質は重要です。

また、2ベクトルのなす角 θ そのものを求めたい場合には、逆三角関数であるアークコサイン (\cos^{-1}) を用いて

$$\theta = \boxed{}$$

と求めることも可能です。3D空間内の2ベクトルのなす角を計算するのは、これ以外の方法はあまり簡単ではないため、この方法は非常に重宝します。

ただし、角度 θ まで求めるのは比較的計算コストがかかりますから、本当に θ そのものを求めなければならない状況なのか、 $\cos \theta$ のままで何とかできる状況ではないのか、というのはよく検討する必要があるでしょう。

✚ 内積の使い道②：2つのベクトルが直交していることを確かめられる

また、 $\cos\theta$ や θ を求める必要まではない状況でも、

$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}| |\mathbf{b}| \cos\theta$$

という式から、 かつ であれば、

$$\mathbf{a} \cdot \mathbf{b} = 0 \Leftrightarrow \text{} \text{ と } \text{} \text{ が } \text{}$$

という関係が成り立ちます。つまり、内積 $\mathbf{a} \cdot \mathbf{b} = \text{}$ であれば2つのベクトルは直交していて、また逆に、2つのベクトルが直交していれば、内積 $\mathbf{a} \cdot \mathbf{b} = \text{}$ となる、ということです。

このことは、 $\mathbf{a} \cdot \mathbf{b} = 0$ で $|\mathbf{a}| \neq 0$ かつ $|\mathbf{b}| \neq 0$ であれば $\cos\theta = \text{}$ も0になることから $\theta = \frac{\pi}{2}$ 、つまり2つのベクトルが直交することになること、また2つのベクトルが直交するということは $\theta = \frac{\pi}{2}$ であることから $\cos\theta$ が0になって内積も0になることからわかります。

これによって、2つのベクトルが直交しているかどうかを判定することもできますし、

$$\mathbf{a} \cdot \mathbf{b} = a_x b_x + a_y b_y + a_z b_z$$

という関係を使って、この $a_x b_x + a_y b_y + a_z b_z$ を0にするように成分を調整することで、あるベクトル \mathbf{a} になベクトルを作り出すこともできます。

NOTE

それを系統的に行う方法として、グラム・シュミットの方法というものもあります。
本書では扱いませんが、興味のある方は調べてみるとよいでしょう。

外積

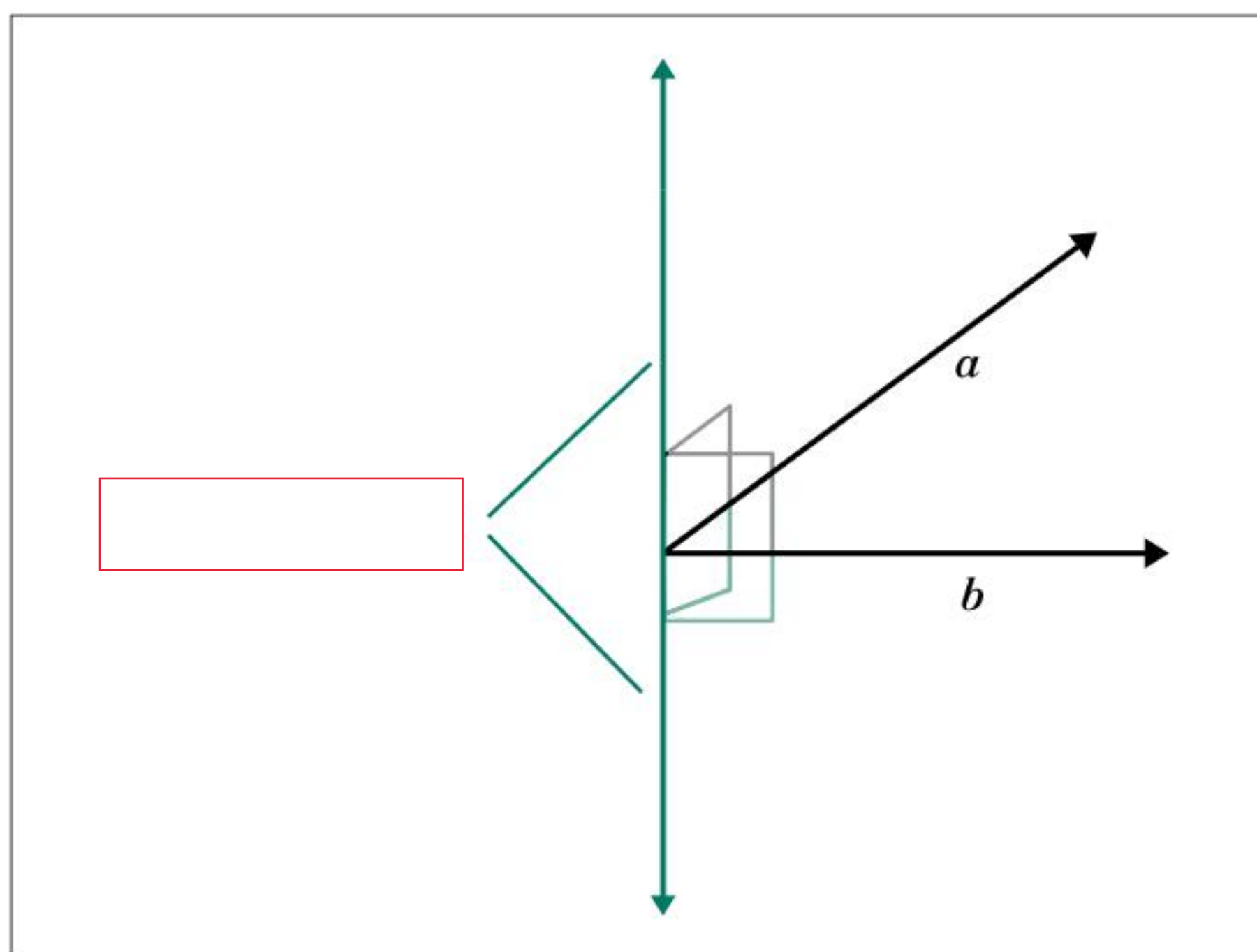
というのは、内積とは別のもう1つのベクトル同士の掛け算です。この外積は、2Dのベクトルにはなかったもので、内積とは違い、計算結果もベクトルになります。

外積の定義は、

$$\mathbf{a} \times \mathbf{b} = \text{}$$

というものです。ここで、 θ はベクトル \mathbf{a} と \mathbf{b} のなす角、 \hat{n} はベクトル \mathbf{a} と \mathbf{b} のする単位ベクトルです。

ここで、「ベクトル \mathbf{a} と \mathbf{b} の」というのは、互いに方向が逆の2本が存在しますが(図6-5-7)、このうちどちらを取るのかは、座標系の取り方によるため、以下のように決まると覚えておくとよいでしょう。



● 図6-5-7 ベクトル a と b の両方に直交する単位ベクトルは、互いに向きが逆な2本が存在する

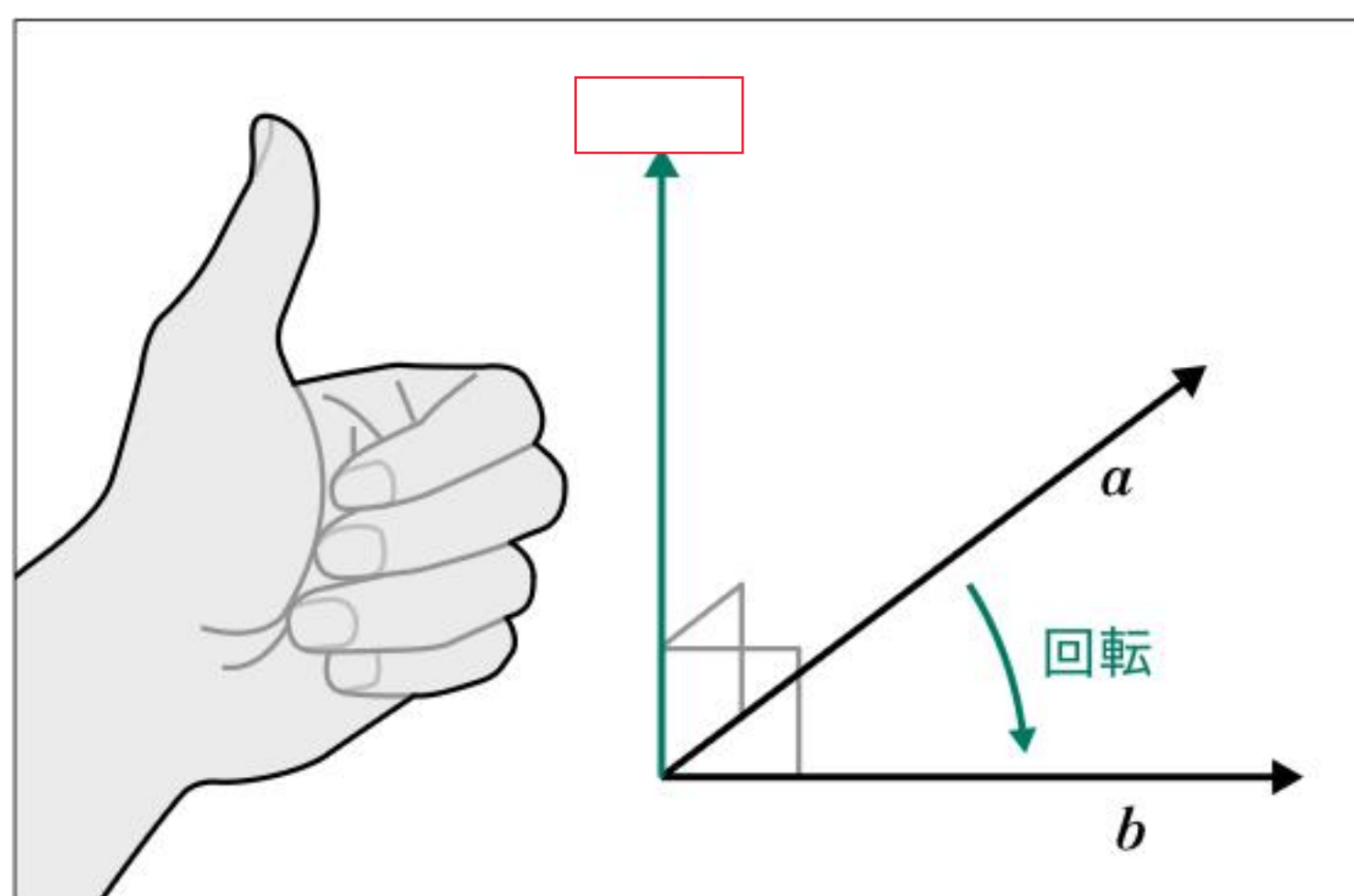
ある座標系で \hat{n} の方向を決める場合、 x 軸方向の単位ベクトル（基底ベクトル）を i 、 y 軸方向の単位ベクトルを j 、 z 軸方向の単位ベクトルを k としたとき \hat{n} の向きは、

$$\boxed{} = \boxed{}$$

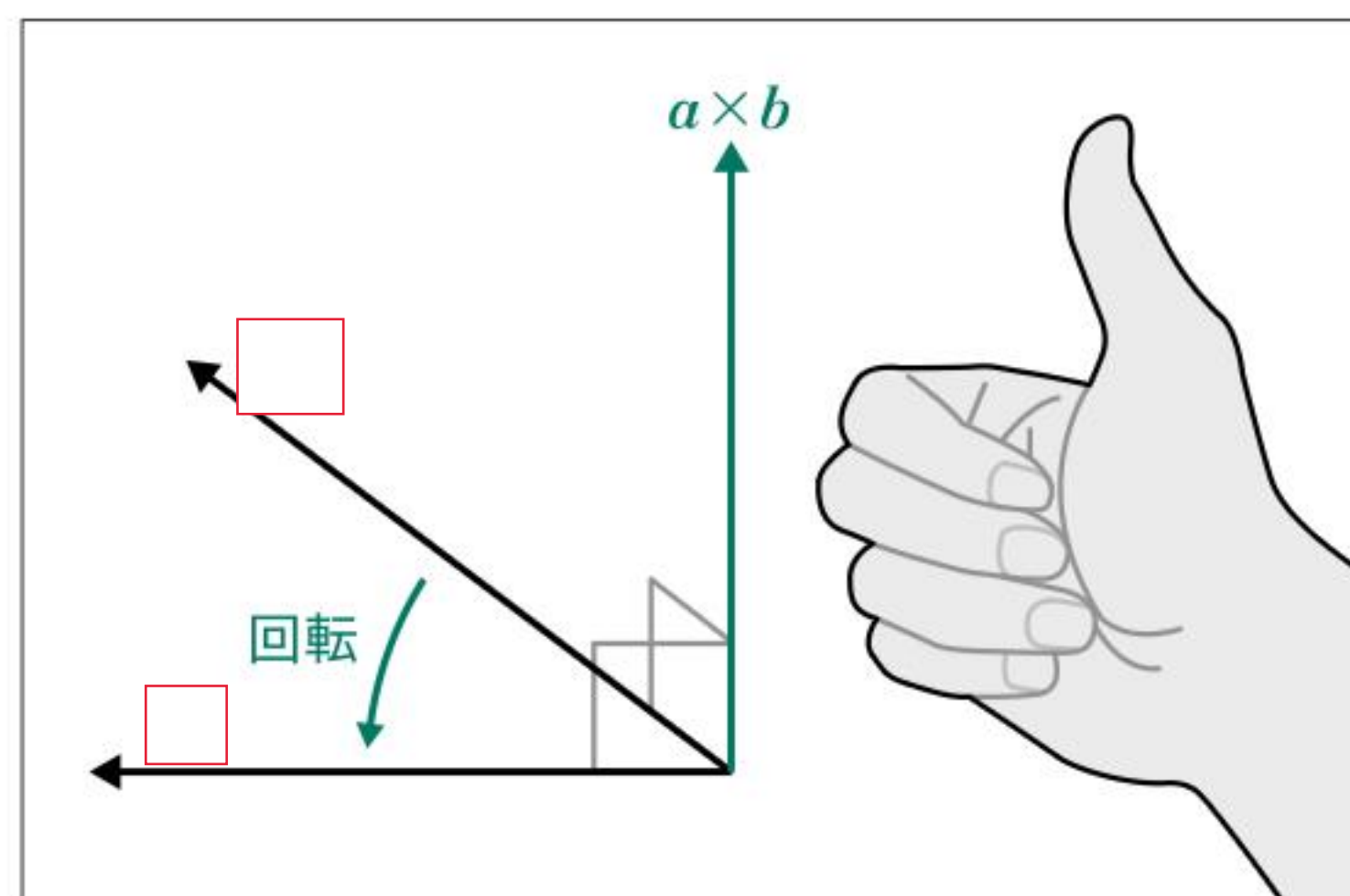
となるような向きになります。

例えば、 $a \times b$ であれば、Unityなどで普通に使う左手系の場合には、 a を b に一致するように回転したとき、その回転方向に丸めた左手指（親指を除く）の方向を合わせたときに、親指が向いている方向に \hat{n} を取ります（図6-5-8）。

また、数学の教科書などでよく用いられる右手系の場合には、同じく a を b に一致するように回転したとき、その回転方向に丸めた右手指（親指を除く）の方向を合わせたときに、親指が向いている方向に \hat{n} を取ります（図6-5-9）。



● 図6-5-8 左手系の座標における $a \times b$



● 図6-5-9 右手系の座標における $a \times b$

教科書などの資料では右手系を前提に書かれていることが多く、その場合Unityなどでよく使われている左手系での外積とは向きが真逆になるため、資料を参照するときには注意が必要です。

✦ 外積の計算方法

さて、この外積

$$\boldsymbol{a} \times \boldsymbol{b} = \boxed{}$$

ですが、このままでは（内積の場合と同様に）具体的にどう計算してよいのかわかりません。取りあえず \boldsymbol{a} と \boldsymbol{b} のなす角 θ もわかりませんし、 \boldsymbol{a} と \boldsymbol{b} の両方に直交するベクトル $\hat{\boldsymbol{n}}$ も、どう出したらいいかかわからないでしょう。

しかし、幸いなことに、外積についても内積の場合と同様に、ベクトル \boldsymbol{a} と \boldsymbol{b} の成分を使って、簡単確実に計算をすることができます。

$\boldsymbol{a} = (a_x \ a_y \ a_z)$ 、 $\boldsymbol{b} = (b_x \ b_y \ b_z)$ とすると

$$\boldsymbol{a} \times \boldsymbol{b} = (\boxed{})$$

となることが知られています。これについては、なぜか、という証明は少し複雑なためここでは省略します。

この「成分での式」を使えば、先ほどの、基底ベクトル同士の関係

$$\boxed{} = \boxed{}$$

は簡単に確かめることができます。

✦ 外積の使い道①：あるベクトルに垂直なベクトルを得られる

先ほど、外積の計算方法についての証明は省きましたが、そこで重要なのは2つ、

- ・この成分計算はどんなベクトル同士でも確実にできる

という性質と、この成分計算によって外積のベクトルを作ることので得られる

$$\boldsymbol{a} \times \boldsymbol{b} = |\boldsymbol{a}| |\boldsymbol{b}| \sin \theta \cdot \hat{\boldsymbol{n}}$$

という関係から、

- ・ベクトル \boldsymbol{a} と \boldsymbol{b} の $\boxed{}$ で、長さが $\boxed{}$ であるようなベクトルが得られる

という性質です。

この「2つのベクトルに垂直なベクトルを得る」処理は、あるポリゴンの法線ベクトルを得られるため、ゲームでは非常に重要になってきます。

NOTE

ただし、外積を使って法線ベクトルを得る場合、その法線ベクトルがどちらを向くのかを、きちんと把握しておく必要があります。先ほど触れたように、Unityなどでよく使われている左手系では、左手を使って外積結果の方向を知ることができます。

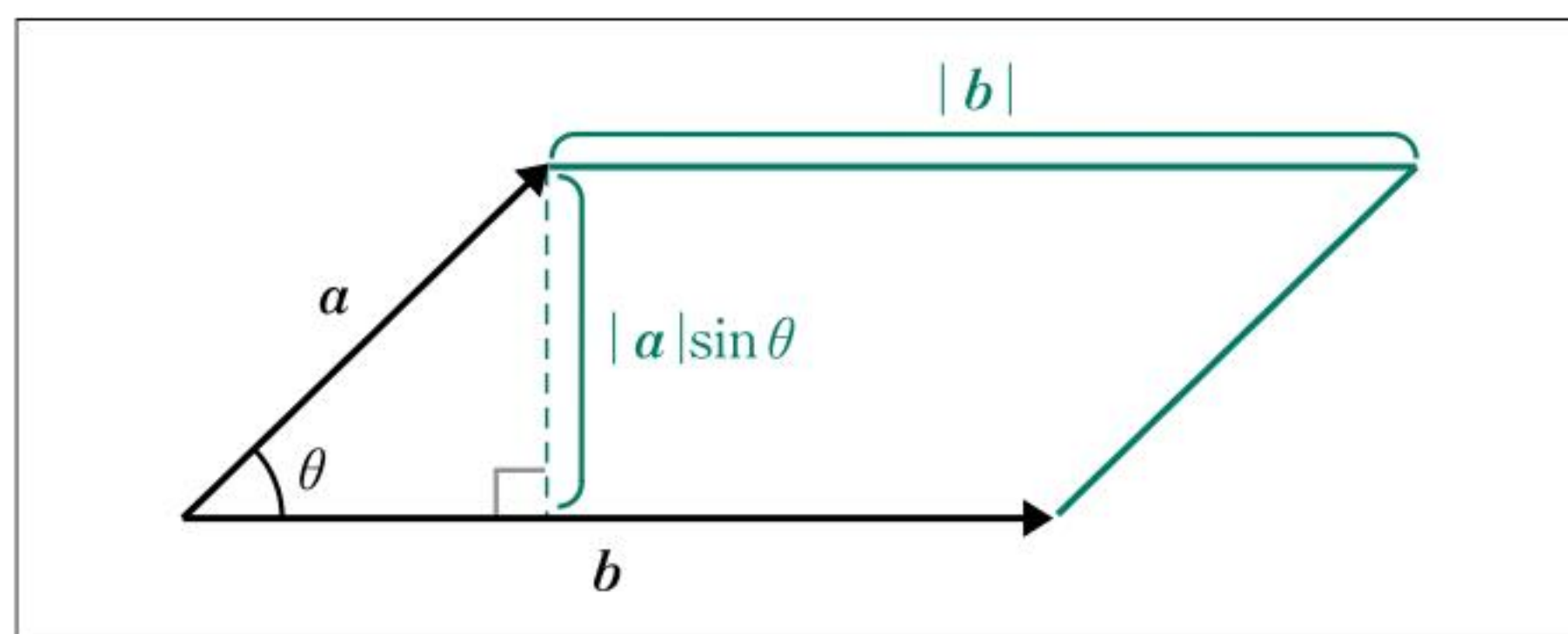
また、外積を取る順番を変えるとベクトルの方向は真逆になるということ、つまり

$$a \times b = -b \times a$$

となるため、意図しない方向に外積のベクトルが向いてしまうのであれば、外積を取る順番を変えるだけで逆方向のベクトルを求めることができます。

✦ 外積の使い道②：2つのベクトルの方向がどのくらい違うかを得られる

外積結果の長さである $|a| |b| \sin \theta$ という値をよく見ると、これはベクトル a と b から作られる 平行四辺形 になっています (図6-5-10)。



● 図6-5-10 ベクトル a 、 b から作られる平行四辺形と、 $|a| \sin \theta$ および $|b|$ の関係

この性質を使うと、2つのベクトルの方向がどれだけ違っているかを、外積結果のベクトルの長さによって評価することができます。この場合、外積のベクトルの長さが小さいほど、2ベクトルは同じような方向を向いていて、外積のベクトルの長さが大きいほど、2ベクトルは違った向きを向いていることになります。

NOTE

ただし、この結果は 0 の場合に限られます。

$\frac{\pi}{2}$ つまり直角よりも大きく方向が違っていた場合、今度は方向が違うほど外積のベクトルは短くなります。

この外積は、3Dプログラミングをする際にキーポイントとなり得る重要なものであり、ぜひとも覚えておくことをおすすめします。

基底ベクトル

とは、各座標軸の方向を向き、長さ1のベクトルのことです。


例えば、2Dの座標 (x, y) は、 x 方向の基底ベクトル $i = (1, 0)$ と y 方向の基底ベクトル $j = (0, 1)$ を使ってと表すことができます。このような表現の利点というのは、 $i = (1, 0)$ や $j = (0, 1)$ という単純なベクトルの組み合わせで座標を表現することで、単純な基底ベクトルのみを変換すればあらゆる座標に対する変換を表現できる点にあります。


NOTE

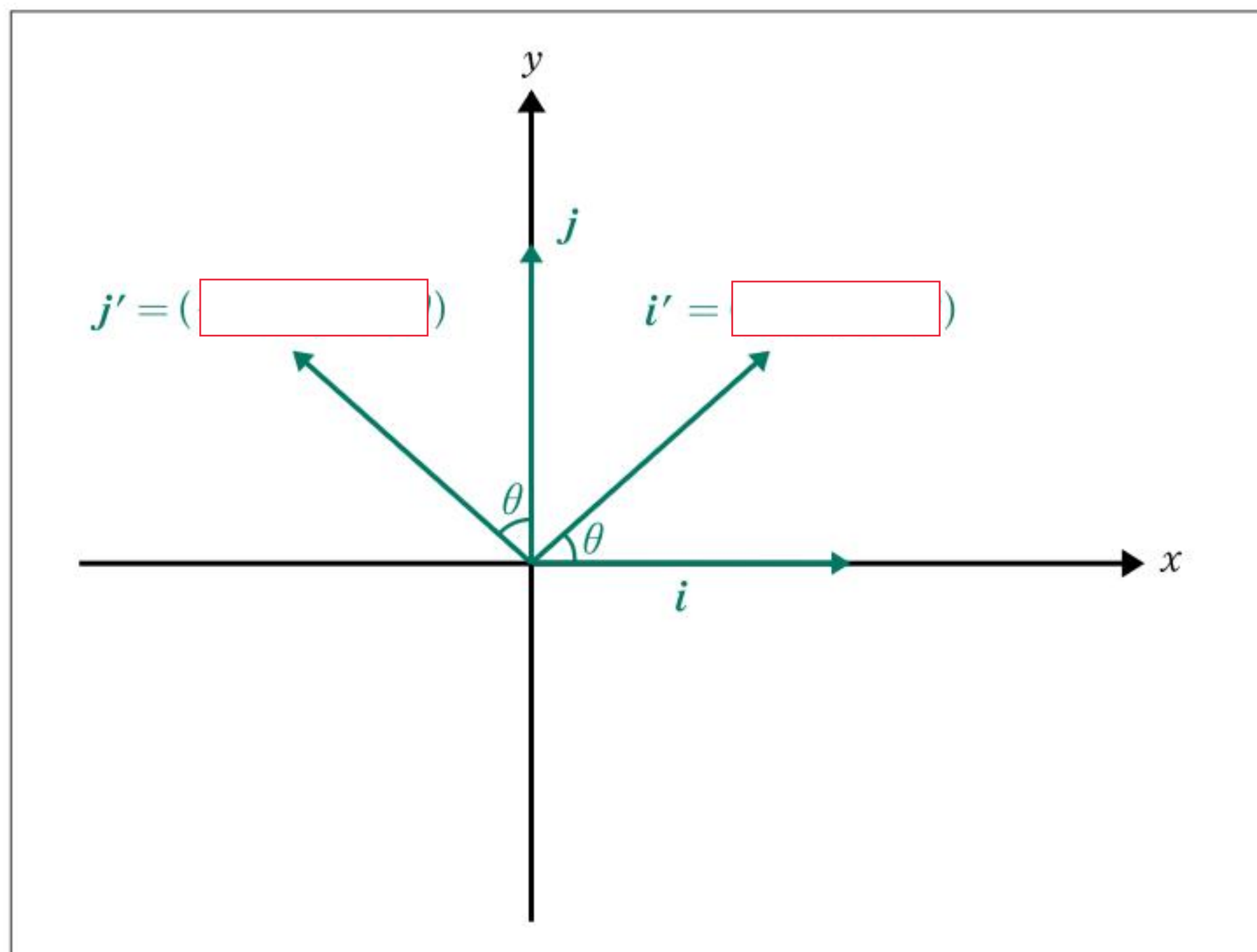
ただし、その場合の変換は、行列で表現できるである必要があります。


回転変換の例


例えば、回転変換について考えてみましょう。

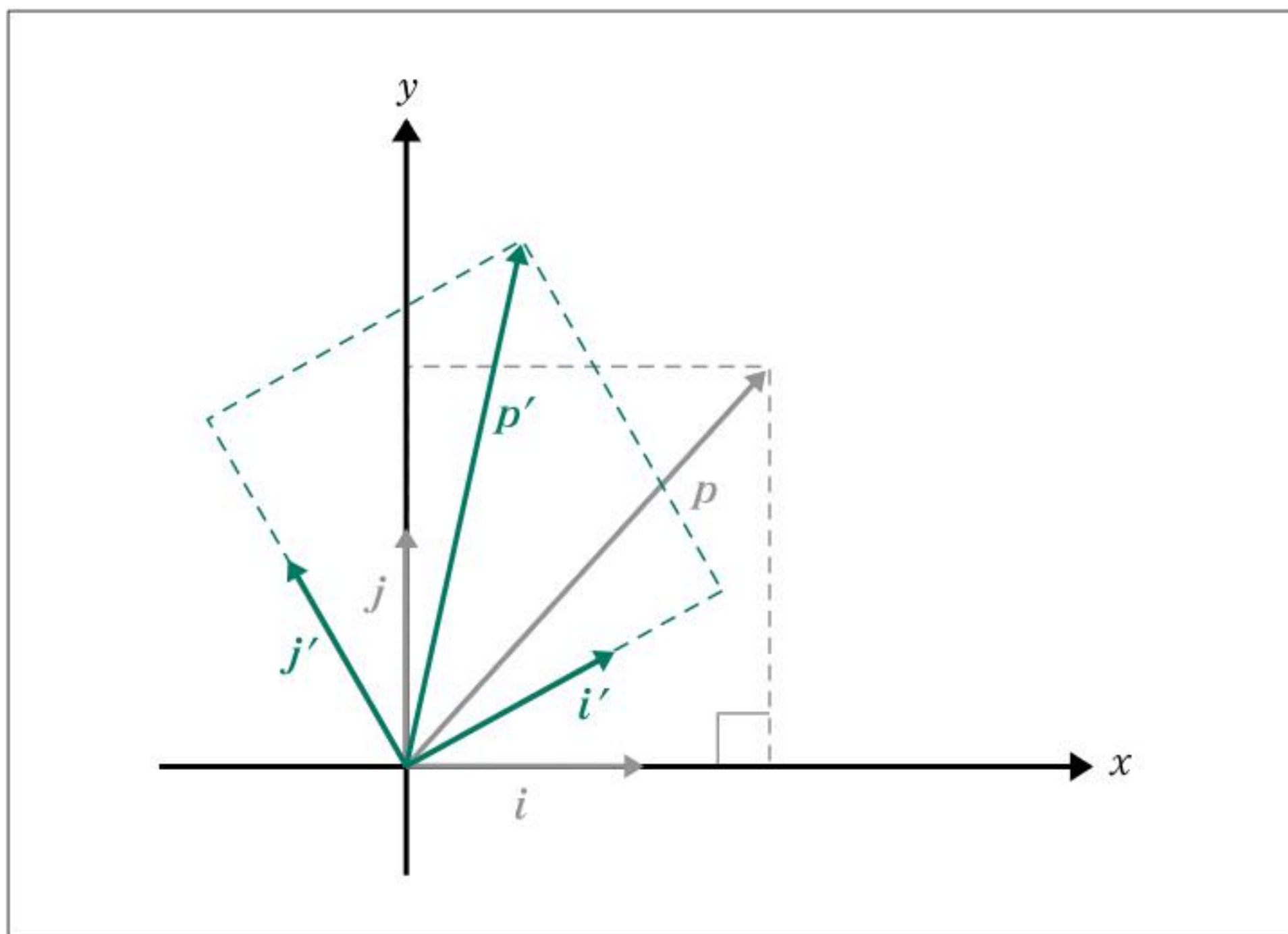
一般の座標を θ だけ回転する変換というのは少し考えにくいものがありますが、 $i = (1, 0)$ を θ だけ回転するだけなら、から結果が()であることがすぐわかります。これを i' としておきましょう。

また、 $j = (0, 1)$ を θ だけ回転すれば()となることもからすぐにわかります。これは j' としましょう。



●  基底ベクトル i および j を θ だけ回転させる

そこで、ある点 $p = xi + yj$ を θ だけ回転した点 p' を考えると、 $p' =$ となります ( 6-5-12)。



● 図 6-5-12 点 p ($xi+yj$) を θ だけ回転させる

各ベクトルを列ベクトルで表して、 $p' = \begin{pmatrix} x' \\ y' \end{pmatrix}$ とすると、 $i' = \begin{pmatrix} \square \\ \square \end{pmatrix}$ 、 $j' = \begin{pmatrix} \square \\ \square \end{pmatrix}$ なので、 $p' = xi' + yj'$ から

$$\begin{aligned} \begin{pmatrix} x' \\ y' \end{pmatrix} &= x \begin{pmatrix} \square \\ \square \end{pmatrix} + y \begin{pmatrix} \square \\ \square \end{pmatrix} \\ &= \begin{pmatrix} \square \\ \square \end{pmatrix} \\ \therefore \begin{cases} x' = \square \\ y' = \square \end{cases} \end{aligned}$$

となります。この結果は、2Dの回転の行列を使って変換を行ったものと同じになっていることを確かめてみてください。

それと同じように、3D空間でも

- ・ x 方向の基底ベクトル $i = \begin{pmatrix} \square \\ \square \\ \square \end{pmatrix}$
- ・ y 方向の基底ベクトル $j = \begin{pmatrix} \square \\ \square \\ \square \end{pmatrix}$
- ・ z 方向の基底ベクトル $k = \begin{pmatrix} \square \\ \square \\ \square \end{pmatrix}$

を考えて、

- ・ 座標を $\begin{pmatrix} \square \\ \square \\ \square \end{pmatrix}$ と表現し
- ・ それぞれの基底ベクトルを特定の方向に向けることによって $\begin{pmatrix} \square \\ \square \\ \square \end{pmatrix}$ を作り出す

というテクニックがよく使われるので、ぜひ覚えておきましょう。