A summary of Projective Geometric Algebra

Patrolin

September 29, 2022

Abstract

This is an explanation of how to calculate PGA, but not necessarily why it works - for that you can try sudgylacmoe on youtube. I would also point you to bivector.net, but their dual calculations are whack, the ganja.js ones are presumably correct, since their demos work.

I. Geometric numbers

$$1 + x^2 = 0$$
$$x = ?$$

x is not a real number; but if it's not real, why should the other numbers be real?

$$(e_1)^2 + (e_2)^2 = (e_0)^2$$
$$(e_1)^2 = 1;\ (e_2)^2 = -1;\ (e_0)^2 = 0$$

In fact, we can define as many of these as we want, the simplest examples being:

$a + be_1$ `// hyperbolic numbers`

$a + be_2$ `// complex numbers`

$a + be_0$ `// dual numbers`

We can multiply these numbers together using the geometric product:

$$e_i e_i = \{1, -1, 0\}$$
$$e_i e_j = -e_j e_i$$

This product is neither commutative nor anticommutative, but it is distributive and associative:

$$AB \neq BA$$
$$AB \neq -BA$$
$$A(B + C) = AB + AC$$
$$(AB)C = A(BC)$$
$$aB = Ba;\ a \in \mathbb{R}$$

Thus the product of two complex numbers:

$$\begin{aligned}
&(A_1 + A_2e_2)(B_1 + B_2e_2) \\
&= A_1B_1 + A_1B_2e_2 + A_2e_2B_1 + A_2e_2B_2e_2 \\
&= A_1B_1 + A_1B_2e_2 + A_2B_1e_2 + A_2B_2 \\
&= (A_1B_1 + A_2B_2) + (A_1B_2 + A_2B_1)e_2
\end{aligned}$$

II. Rotations

A multivector with n basis vectors consists of 2^n blades:

- `scalar = 0-vector =` $\mathbf{1}$
- `vector = 1-vector`
- `bivector = 2-vector`
- `trivector = 3-vector`
- `...`
- `(n-1)-vector = pseudovector`
- `n-vector = pseudoscalar =` $\mathbb{1}$

Where a k-vector has $\binom{n}{k}$ blades, for example:

$$\begin{aligned}
A = A_1 \\
&+ A_2e_0 + A_3e_1 + A_4e_2 \\
&+ A_5e_{01} + A_6e_{02} + A_7e_{12} \\
&+ A_8e_{012}
\end{aligned}$$

We can abbreviate blades like e_1e_2 as e_{12}.

Multiplying two multivectors gives you another multivector, we can use the taylor series expansion of the exponential function to find a rotation e^A:

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}$$

$$\begin{aligned} e^{ae_2} &= 1 + ae_2 - \frac{a^2}{2} - \frac{a^3}{3}e_2 + ... \\ &= (1 - \frac{a^2}{2} + ...) + (a - \frac{a^3}{3} + ...)e_2 \\ &= cos(a) + sin(a)e_2 \end{aligned}$$

Similarly you can find

$$e^{e_i} = \begin{cases} cos(a) + sin(a)e_i & ((e_i)^2 = -1) \\ cosh(a) + sinh(a)e_i & ((e_i)^2 = 1) \\ 1 + e_i & ((e_i)^2 = 0) \end{cases}$$

This gives us rotations, hyperbolic rotations and translations (rotations through infinity) respectively.

Then for a multivector we would have:

$$e^A = e^{A_1} e^{A_2 e_0} e^{A_3 e_1} ... e^{A_n e_I}$$

III. Unary operators

For the i-th blade in a multivector

$$X_i \in \text{k-vector}$$

we can define some operations, like reversing the order of basis vectors in the blade, that amount to flipping some signs:

$$\begin{aligned} \tilde{X}_i &= (-1)^{\lfloor k/2 \rfloor} X_i \texttt{ // reverse} \\ X_i^{\dagger} &= (-1)^{\lfloor k \rfloor} X_i \texttt{ // involute} \\ \bar{X}_i &= (-1)^{\lfloor k+k/2 \rfloor} X_i \texttt{ // conjugate} \\ f(A) &= \sum_i f(X_i) \end{aligned}$$

Poincaré duality states that maps between k-vectors and (n-k)-vectors exist.

$$\begin{aligned} X_i \, dual(X_i) &= \pm \mathbb{1} \\ dual(X_i) &= \pm X_{2^n - i + 1} \\ dual(A) &= \sum_i dual(X_i) \end{aligned}$$

For example:

$$\begin{aligned} \underline{X_i} X_i &= \mathbb{1} \texttt{ // left complement} \\ X_i \overline{X_i} &= \mathbb{1} \texttt{ // right complement} \\ X_i X_i^{\star} &= sign(X_i^{ND} \widetilde{X_i^{ND}}) \mathbb{1} \texttt{ // hodge dual} \end{aligned}$$

Where X_i^{ND} is X_i without degenerate basis vectors, e.g.

$$X_i = e_{012};\ X_i^{ND} = e_{12}$$

Let $\mathbb{G}_{a,b,c}$ be a geometric algebra with a positive, b negative and c zero basis vectors.

Then for $\mathbb{G}_{a,0,c}$:

$$\overline{X_i} = X_i^{\star}$$

And if that wasn't confusing enough, applying a dual twice changes the signs, so we also want the inverses of these duals:

$$\begin{aligned} (X_i^{\star})^{\star^{-1}} &= X_i \\ \overline{\overline{X_i}} &= X_i \\ \underline{\underline{X_i}} &= X_i \end{aligned}$$

In 2D and 3D PGA, we can simplify implementation by swapping two basis vectors in some blades such that $\overline{X_i}$ does not flip signs, e.g.

$$\begin{aligned} A &= A_1 + A_2 e_0 + A_3 e_1 + A_4 e_2 \\ &\quad + A_5 e_{01} + A_6 e_{20} + A_7 e_{12} + A_8 e_{012} \\ \overline{\overline{A}} = \overline{A} &= A_8 + A_7 e_0 + A_6 e_1 + A_5 e_2 \\ &\quad + A_4 e_{01} + A_3 e_{20} + A_2 e_{12} + A_1 e_{012} \end{aligned}$$

IV. Shapes and sizes

Let $\mathbb{G}_{d,0,1}$ be a d-dimensional PGA:

$$(e_0)^2 = 0;\ (e_i)^2 = 1$$

Now we have a choice to make:

1. `Point-based PGA`
   - `vectors are points`
   - `(n-1)-vectors are hyperplanes`
2. `Plane-based PGA`
   - `vectors are hyperplanes`
   - `(n-1)-vectors are points`

Both of these are equally valid and many operations make use of computing in the dual algebra via $\overline{\overline{A} \text{ op } \overline{B}}$.

$$
\begin{aligned}
point &= e_0 + xe_1 + ye_2 + ... + we_d \\
hyperplane &= \overline{e_0 + xe_1 + ye_2 + ... + we_d}
\end{aligned}
$$

$$
\boxed{
\begin{aligned}
hyperplane &= e_0 + xe_1 + ye_2 + ... + we_d \\
point &= \overline{e_0 + xe_1 + ye_2 + ... + we_d}
\end{aligned}
}
$$

We will denote Plane-based operations inside boxes whenever they differ.

Let $\langle A \rangle_k$ be the grade selection operator:

$$\langle A \rangle_k = \sum_i \langle X_i \rangle_k$$

$$\langle X_i \rangle_k = \begin{cases} X_i & (X_i \in \text{k-vector}) \\ 0 & (X_i \notin \text{k-vector}) \end{cases}$$

Then we can start defining binary operators:

$$A \cdot B = \sum_{j,k} \left\langle \langle A \rangle_j \langle B \rangle_k \right\rangle_0 \texttt{ // dot product}$$

$$A \wedge B = \sum_{j,k} \left\langle \langle A \rangle_j \langle B \rangle_k \right\rangle_{j+k} \texttt{ // wedge product}$$

$$A \vee B = \overline{\overline{A} \wedge \overline{B}} \texttt{ // antiwedge product}$$

In other words the wedge product is equal to the geometric product if the grades add together:
j-vector $\wedge$ k-vector = (j+k)-vector, otherwise it's 0

All of these operators retain distributivity and associativity

The antiwedge product allows you to join points into lines, planes, ...

$$
\boxed{
\begin{aligned}
line &= point_1 \vee point_2 \\
plane &= point_1 \vee point_2 \vee point_3
\end{aligned}
}
$$

TODO: Point-based join

In fact this works with any two geometric objects, operators that don't have this property aren't really worth your time.

And the wedge product allows you to meet two objects:

$$
\boxed{
\begin{aligned}
point &= line_1 \wedge line_2 \\
line &= plane_1 \wedge plane_2
\end{aligned}
}
$$

TODO: Point-based meet

If you meet two parallel lines, you get an infinite point = a point at infinity:

$$line_1 \wedge line_1 = \overline{xe_1 + ye_2 + ...}$$

All objects in GA have an orientation, so if you flip the direction of one of the lines, you would get an infinite point in the other direction

TODO: something about sizes of line segments / volumes

TODO: projection to camera plane + depth buffer

V. Motors

TODO: something about bivector blades being rotations

$$rotor = e^{bivector^{ND}}$$

$$motor = e^{bivector}$$

In 3D, rotors are quaternions.

TODO: applying motors and line forces