

# ZONOEDROS

**Nelo D. Allan**<sup>1</sup>

Depto de Matematica, Faciex, UNEMAT,  
78200-000, Caceres, MT  
E-mail: [neloallan@yahoo.com](mailto:neloallan@yahoo.com),  
[neloallan@gmail.com](mailto:neloallan@gmail.com)

**Resumo:** Uma classe de poliedros interessantes, a dos zonoedros, é composta pelos poliedros onde todas as faces são paralelogramos. Esta classe contém os chamados zonoedros polares, que são poliedros que possuem dois vértices opostos onde incidem pelo menos três faces e em todos os outros vértices incidem quatro faces. Nosso objetivo é exibir uma função construída passo a passo no Mathematica a fim de exibir exemplos destes objetos.

## Introdução e curta história

O primeiro matemático a introduzir romboedros, zonoedros cujas faces são losangos regulares, foi Kepler. Ele apresentou o dodecaedro rômbo e o triacontaedro rômbo construídos respectivamente, através do processo chamado de estrelagem do cubo e do icosaedro. Ele postula o problema de encontrar todos estes poliedros. O cristãgrafo E. Fedorov [2] no início do século passado estudou as pavimentações do plano e do espaço, como também os zonoedros. Mais tarde Coxeter [1] estende este estudo a poliedros em dimensões superiores como também ao espaço hiperbólico. Na década dos 60 Johnson [4] apresenta uma lista de 92 poliedros convexos com faces regulares e que hipoteticamente representa todos poliedros deste tipo. Mais recentemente Hart [3] e Towlet [5] apresentam a função PolarZonohedron.

## Detalhes técnicos

Voltemos aos romboedros. Os mais simples são obtidos a partir dos poliedros regulares onde colamos em cada face uma pirâmide regular, calibrando suas alturas de tal modo que dois ápices adjacentes sejam coplanares com uma aresta do poliedro. No caso do tetraedro obteremos um hexaedro rômbo, no do cubo um dodecaedro rômbo e no caso do icosaedro, um triacontaedro (trinta faces). A partir deste último podemos construir o icosaedro rômbo.

Dado um zonoedro, cada par de lados de um de seus paralelogramos determina uma família de arestas paralelas. Se  $n$  for o número de famílias de arestas distintas podemos verificar [1] que o zonoedro tem  $n(n-1)$  faces,  $2n(n-1)$  arestas e  $n(n-1)+2$  vértices. Para cada direção ele possui

um cinturão central de  $n$  paralelogramos com  $n$  arestas paralelas. A classificação dos zonoedros é simples [1],p27-29].

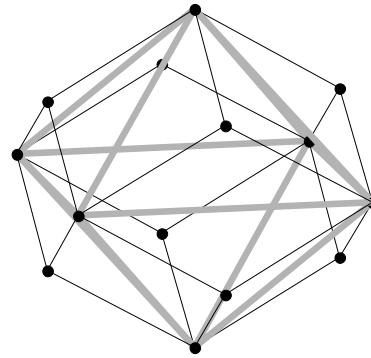


Figura 1: Dodecaedro rômbo com octaedro inscrito

Chamaremos de estrela a um conjunto de segmentos que tem seus pontos médios em comum. Uma estrela é não degenerada se qualquer terna de segmentos distintos são não coplanares. Um zonoedro determina uma estrela: o conjunto de segmentos equi-polentes às suas arestas. Reciprocamente qualquer estrela não degenerada determina um zonoedro. Uma estrela é determinada por um conjunto de  $n$  vetores  $\mathbf{v}(\mathbf{OP}_i)$ . Os vértices do poliedro são obtidos por  $\mathbf{v}(\mathbf{OP})$ , combinações lineares destes vetores a coeficientes iguais a zero ou um. O poliedro é a envoltória convexa dos pontos  $\mathbf{P}$ .

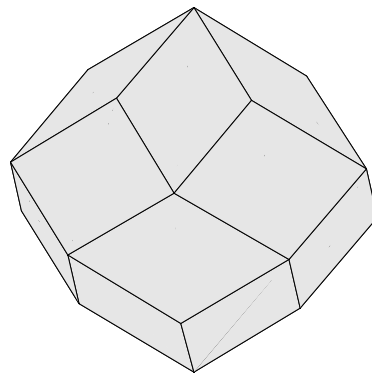


Figura 2: Icosaedro rômbo

<sup>1</sup> Este trabalho teve o apoio da Fapemat e do FidPex/Unemat.

Uma estrela degenerada também determina um poliedro onde as faces são paralela aos pares e ele é também central. Vamos chamar de zonoedro qualquer poliedro cujas faces são paralelogramos.

Em geral um zonoedro pode ser bem complicado porém, em princípio eles podem ser grosseiramente classificado pelo gênero ou número de buracos. Mesmo em gênero zero ele pode ser obtido por colagem de um número qualquer de prismas.

## Construção do Zonoedro

Seja um conjunto ordenado de  $n$  vetores  $v(OP_i)$  de tal modo que existe um plano  $\Pi$  e um ponto  $Q$  de  $\Pi$  tal que as reta  $OP_i$  interceptam  $\Pi$  em pontos  $Q_i$  de modo eles determinam um polígono tal que os segmentos  $QQ_i$  estão contido neste polígono (estrela-côncavo). Assim todos nossos vetores estão de um mesmo lado de um plano. Obtemos uma primeira bateria de  $n$  faces, paralelogramos  $\{O, P_i, P_i + P_{i+1}, P_{i+1}\}$ , que incidem em  $O$ . Aqui pomos  $P_{n+1} = P_1$ . Repetimos o processo para  $P_i$  construindo os paralelogramos  $\{P_i, P_i + P_{i+1}, P_{i+1} - 2P_i + P_{i-1}, P_{i-1} + P_i\}$  e assim sucessivamente. Por construção obtaremos um zonoedro onde todos os paralelogramos terão arestas iguais se todos os vetores  $v(OP_i)$  tiverem mesmo comprimento. Apresentamos assim um algoritmo para a construção de zonoedros que depende somente do número de vetores satisfazendo as nossas restrições. Eles terão dois vértices opostos onde incidem  $n$  arestas e em todos os outros incidem quatro arestas. Nosso algoritmo produz neste caso todos os vértices do poliedro.

## O algoritmo:

Trabalhemos primeiramente com uma família de vetores que é base para  $R^n$ . Comecemos exibindo a família de vetores da base canônica de  $R^n$ . Vamos designá-los por  $e_{nj} = ee[n, j]$ ; partimos da origem em  $R^{n-1}$  e inserimos  $1$  em cada posição de  $\{0, \dots, 0\}$ . Em comandos temos

```
ee[n_,j_]:=Table[If[k==j,1,0],{k,1,n}]
```

Tomemos  $vertGer = e_{(m+1)} + \dots + e_{(m+k)}$  ou em comandos:

```
vertGer[n_,0,k_]:=If[k==0,Table[0,{j,n}],
Sum[ee[n,j],{j,k}]
vertGer[n_,m_,k_]:=
RotateRight[vertGer[n,0,k],m]
```

Fixado  $k$  chamamos de nível  $k$  a união de todos as  $n$ -uplas com  $k$  elementos iguais a um.

```
nivelZ[n_,k_]:=
```

```
If[k>0,Union[Table[vertGer[n,m,k],
{m,1,n}],{ee[n,0]}]
```

para cada elemento do nível  $k$ ,  $e_{(m+1)} + \dots + e_{(m+k)}$  temos dois vértices adjacentes que são obtidos adicionando respectivamente  $e_{(m+k+1)}$  e  $e_{(m+k-1)}$  a ele, que correspondem aos comandos

```
vertGer[n,m,k+1]; e vertGer[n,m-1,k+1];
```

Cada vértice  $vertGer[n,m,k]$  e dois lados adjacentes que determinam três vértices do paralelogramo face. Este paralelogramo será

```
paralelogramo[n_,m_,k_] := {vertGer[n,m,k],
vertGer[n,m-1,k+1],vertGer[n,m,k+1]+
vertGer[n,m-1,k+1]-vertGer[n,m,k],
vertGer[n,m,k+1], vertGer[n,m,k]}
```

Assim todos os vértices formais ou pré-vértices do zonoedro serão dados por

```
vertices[n_] :=
```

```
Flatten[Table[nivelZ[n,j],{j,0,n}],1]
```

e todas as suas pré-faces laterais serão

```
preFaixa[n_,k_] :=
Table[paralelogramo[n,m,k],{m,1,n}]
todasFaces[n_] := Flatten[Table[
paralelogramo[n,m,k],{m,1,n},{k,0,n-2}],1]
```

O que fizemos acima foi descrever o politopo hipercubo no espaço a  $n$  dimensões. Vamos projetar este politopo e seu esqueleto no espaço a três dimensões. Primeiramente introduzimos famílias simétricas de vetores.

```
pontoEspaco[{r_,teta_,phi_}] := {r * Cos[teta] *
Sin[phi], r * Sin[teta] * Sin[phi], r * Cos[phi]}
ptEspaco[{r_,s_,teta_,phi_}] := {r * Cos[teta] *
Sin[phi], r * Sin[teta] * Sin[phi], s * Cos[phi]}
```

Uma família de  $n$  vetores distribuídos simetricamente em torno do eixo  $z$  é dada por

```
sim[{r_,s_,teta_,phi_}] :=
Table[ptEspaco[{r,s,2 k Pi/n,phi}],{k,n}]
ciclicos[{r_,n_,phi_}] := sim[{r,r,n,phi}]
```

Esta família tem o eixo  $z$  como eixo de uma rotação de ângulo  $2\pi/n$ , e assim nosso poliedro alem de ser central vai ter também este eixo de simetria. Ele vai ser equilátero, i.e., todas suas arestas serão iguais.

Sejam dados agora  $n$  vetores no espaço  $R^3$ :  $(V_1, \dots, V_n) = lyt$  e uma família de coeficientes  $P = (a_1, \dots, a_n)$ . Seu produto escalar será

```
prodEscalar[lyt_][P_] :=
Sum[lyt[[j]] P[[j]],{j,Length[P]}]
```

## Zonoedro

Até aqui trabalhamos formalmente, e agora vamos modelar o zonoedro abstrato substituindo os  $ee[n,i]$  por pontos do espaço que chamaremos de básicos. Nos nossos exemplos os básicos serão o comando **ciclicos**. Assim a  $k$ -ésima seção do zonoedro gerado por básicos, ou paralelogramos do nível  $k$  é dada por

```
faixaNivel[bas_,k]:=
Map[prodEscalar[bas],
preFaixa[Length[bas],k],{2}]
```

O zonoedro será:

```
preZonoedro[bas_]:=
Flatten[Table[faixaNivel[bas,k],
{k,0,Length[bas]-2},1],1]
zonoedro[bas_]:=
Map[Polygon,preZonoedro[bas]]
```

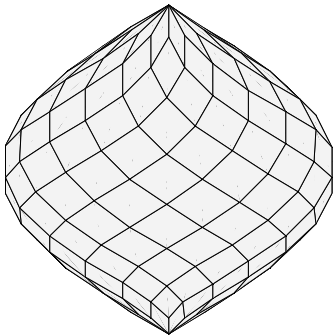


Figura 3: Zonoedro , n=14, phi=1.

Este comando funciona bem quando os vetores  $v[P_i]$  são tais que todos os vetores estão do mesmo lado relativos ao um plano pela origem e esta é um vértice do zonoedro.

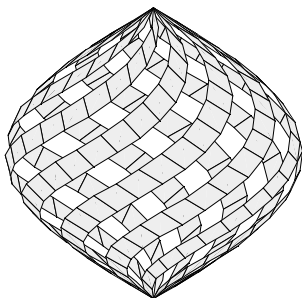


Figura 4: Zonoedro com faixas verticais alternadas

O poliedro gerado pelo comando `zonoedro` não será necessariamente convexo como podemos ver na figura 11. Ele depende da ordem dos vetores. Ele será convexo se os planos determi-

nados pela origem e dois pontos consecutivos determinem um conjunto convexo.

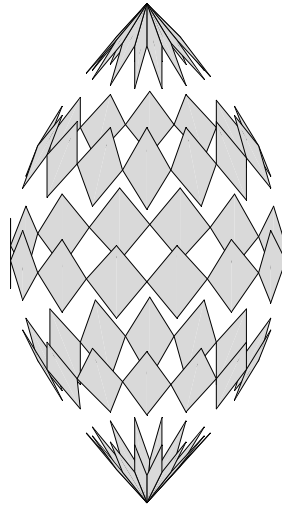


Figura 5: Zonoedro com faixas horizontais

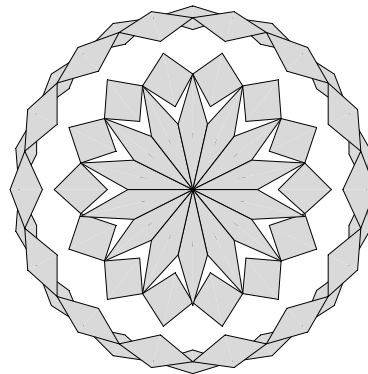


Figura 6: Visão vertical da figura anterior

## Colagem

O processo de colagem no nosso caso será feito da seguinte maneira. Extraímos do poliedro dado uma poligonal fechada simples. Traducamos o resto do poliedro numa direção  $V$  conveniente e inserimos paralelogramos do seguinte modo: se  $\{A,B\}$  é um lado da poligonal então  $\{A+V, B+V\}$  será o transladado deste lado e o paralelogramo terá vértices

$$\{A, A+V, B+V, B\}$$

Os comandos auxiliares serão:

```
add[V_][P_]:=V+P;
addPll[V_][{A_,B_}]:={A,A+V,B+V,B,A}
so234[{A_,B_,C_,D_,A_}]:={B,C,D}
```

A faixa central do zonoedro é uma poligonal fechada que parece os dentes de um serrate. A uma poligonal deste tipo chamaremos de polígono de Petrie.

```
faixaPetrie[bas_,k_]:=Flatten[Map[so234,
    faixaNivel [bas,k],1]
faixaVertical[V_,bas_,k_]:=
Table[addPll[V][{faixaPetrie[bas,k][[j]],
    faixaPetrie[bas,k][[j+1]]}], {j,1,Length[
    faixaPetrie[bas,k]-1}]
```

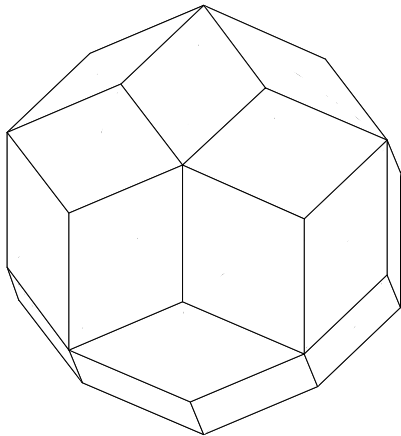


Figura 7: Triacotaedro rômico

Assim o novo zonoedro terá um pólo na origem e uma faixa de arestas paralelas ao eixo z. Ele será dado pelo comando:

```
preZonoExt[bas_, Ve_, k_] := Flatten[
{Map[
add[Ve],Flatten[Table[faixaNivel[bas,j],
{j, k + 1, Length[bas] - 2}], 1], {2}],
faixaVertical[Vê,bas, k],Flatten[
Table[faixaNivel[bas, j],{j, 0, k}], 1],1]
zonoedroEstendido[bas_, Ve_, k_] :=
Polygon/@preZonoExt[bas,Ve,k]
```

## Vértices e arestas

Uma vez que as figuras estejam prontas podemos calcular numericamente as coordenadas dos vértices e as arestas do poliedro: pegamos todos os dados numéricos, por exemplo no comando acima, usamos o **round** para identificar os valores que diferem na quarta casa decimal extraímos todos os parênteses extra e através do **Union** eliminamos os pontos repetidos.

```
round[x_]:= 10^(-3) Round[10^3 x];
verticesPoli[poliedro_]:=Union[Flatten[
round /@ poliedro,1]]
```

Adicionamos os pontos ao gráfico através do comando **Point** que tem uma opção de tamanho. Para facilitar a notação introduzimos o comando **pontos**

```
pontos[lyt_]:=PointSize[.03],Map[Point,lyt]]
```

Para determinar as arestas de um poliedro já conhecido procedemos de maneira semelhante. Temos aqui as faces que são paralelogramos que denominamos ABCDA e temos dois pares de arestas paralelas e duas diagonais, a saber

```
arestasface[{A_,B_,C_,D_,A_}]:=
{{A,B},{C,D},{A,D},{B,C}}
```

Para determinar as diagonais AC e BD é conveniente sabermos qual é a maior, caso o paralelogramo não seja um quadrado. Basta calcular os comprimentos.

```
dist3D[{P_,Q_}]:=Sqrt[(P-Q).(P-Q)]
diagonalMaior[{A_,B_,C_,D_,A_}]:=
If[dist3D[{A,C}]>dist3D[{B,D}], {A,C},{B,D}]
diagonalMenor[{A_,B_,C_,D_,A_}]:=
If[dist3D[{A,C}]>dist3D[{B,D}], {B,D},{A,C}]
arestasZono[dados_List]:= Union[ Sort/@
round/@Flatten[arestasFace/@dados,1]]
```

O esqueleto de um poliedro pode ser obtido se nos comandos zonoedros trocamos **Polygon** por **Line**

## Figuras

Pimeiramente vamos introduzir alguns comando gerais para simplificar a notação. No comando **Show Graphics3D** vamos sistematicamente omitir a “**Box**” por razões de visualização, e também apagamos o **Lighting** para poder recolorir as figuras. Usaremos também a visão aproximação da perspectiva cavaleira. Por questão de nossa visualização usamos **phi** como **1.** radianos. Nossos poliedros não são sombras, i.e., projeções ortogonais de poliedros em dimensão superior.

```
SG3D[{exxp_},opts_]:=
Show[Graphics3D [exxp], opts]
SG3DBL[exxp_, opts_]:=Show[
Graphics3D [exxp], opts, Boxed->False,
Lighting->False]
cavalera:= ViewPoint->{63,290,64}
```

Os zonoedros das figuras 2 e a 3 são polares e assim são direta aplicação de nossos comandos. O dodecaedro rômico é dado por

```
SG3DBL[{cinza[.9],
zonoedro[ciclicos[{1,5,1.}]]},cavalera]
```

Para colorir poliedros no espaço ou usamos o sistema embutido no programa ou usamos a opção **Lighting->False** e daí temos que designar uma cor para cada polígono, caso contrário o programa usa automaticamente o negro. Uma maneira mais fácil é abrir buracos e colorir o resto com a mesma cor; é isto que fazemos com as faixas.

```
SG3DBL[Table[{cinza[.93],Polygon[
Map[prodEscalar[ciclicos[{1,14,1}]], paralelogramo[14,m,h]]],{m,1,14,2},{h,0,12}],cavalera]; cinza[g_]:=GrayLevel[g]
```

Já na figura 5 usamos  $s=1.2$ ,  $\phi=.8$ , e o passo do Table de três em três para h.

```
SG3DBL[Table[{cinza[.85],Polygon[
Map[prodEscalar[sim[{1,1.2,14,.8}]],paralelogramo[14,m,h]]],{m,1,14},{h,0,12,3}],cavalera]
```

No caso do triacontaedro usamos o comando **zonoEstendido** estendendo a faixa central do icosaedro rômboico na direção do eixo z, i.e.,

```
SG3DBL[{cinza[.98],Map[Polygon,
preZonoExt[ciclicos[{1,5,1}],{0,0,1,1}], cavalera]
```

A figura 1 é obtida pelo comando

```
tospo[n_]:=ciclicos[{1,n,Pi/3}];
SG3D[{pontos[verticesPoli[preZonoedro[tospo[4]]],
{Map[Line,preZonoedroAll[tospo[4]]],
{AbsoluteThickness [2], cinza[ .7] , Line/@Map[diagonalMenor,preZonoedro[tospo[4]]]}, cavalera, Boxed->False]
```

Se usarmos acima o comando **diagonalMaior** obteremos um quadrado inscrito neste dodecaedro (figura 9). A figura 7 nos mostra um triacontaedro construído a partir de **tospo[5]**, estendendo por **{0,0,1}**.

Na figura 8 temos o esqueleto do triacontaedro realçando seus vértices. As arestas verticais limitam a parte estendida.

```
SG3DBL[{cinza[.98],pontos[
verticesPoli[preZonoedroExt[tospo[5]],
{0,0,1,1}], Map[Line,preZonoExt[tospo[5],
{0,0,1,1}],cavalera]
```

A figura 10 é interessante e reflete o que acontece quando colocamos junto família de vetores em outra situação:

```
barca:= Join[ciclicos[{1,7,1}],
Table[{0,.5k,3},{k,-5, 5 }]];
```

```
SG3DBL[Table[{cinza[.9],Polygon[
Map[prodEscalar[barca],paralelogramo[
18,m,h]]],{m,1,18},{h,0,16}],
ViewPoint-> {1,0, 0}]
```

Já a figura 11 é obtida por

```
SG3D[{cinza[.9],zonoedro[Join[ ciclicos[
{1.5,7,Pi/3}],{0,1,3}]]],
Boxed->False,Lighting->False ]
```

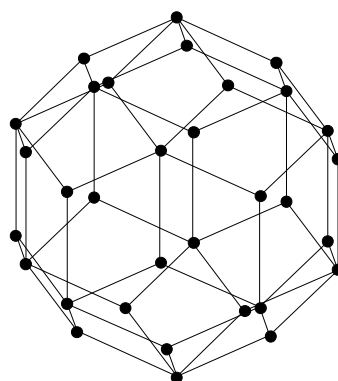


Figura 8: Esqueleto do triacontaedro

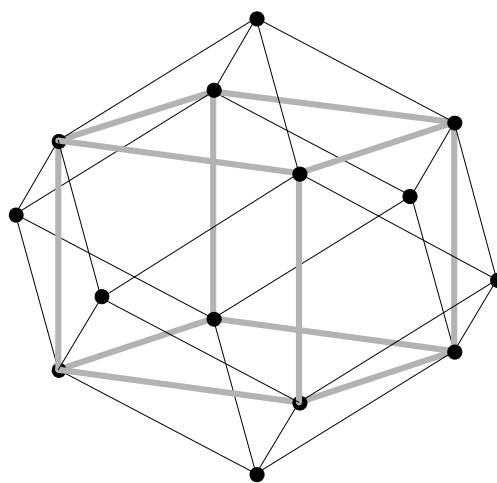


Figura 9: Cubo inscrito no dodecaedro rômboico

## Observações finais

A razão pela qual Fedorov se interessou por zonoedros é que eles estão presentes em várias famílias de cristais e como também determinam pavimentações do espaço. Das sete classes de cristais cinco são obtidas por zonoedros: o cubo, o prisma hexagonal, o dodecaedro rômboico, o dodecaedro alongado, e o octaedro truncado. Uma boa referência para o estudo dos poliedros, bem como uma vasta bibliografia encontramos na respectiva seção do MathWorld, e particularmente o trabalho de E. Weisstein.[6]. O co-

mando **PolarZonohedron** é de G. Hart[3], melhorado por R. Towle[5]; a diferença do que apresentamos está nos dados “básicos”. O artigo de Towle contém também uma boa explicação deste conteúdo. Quanto ao histórico sobre

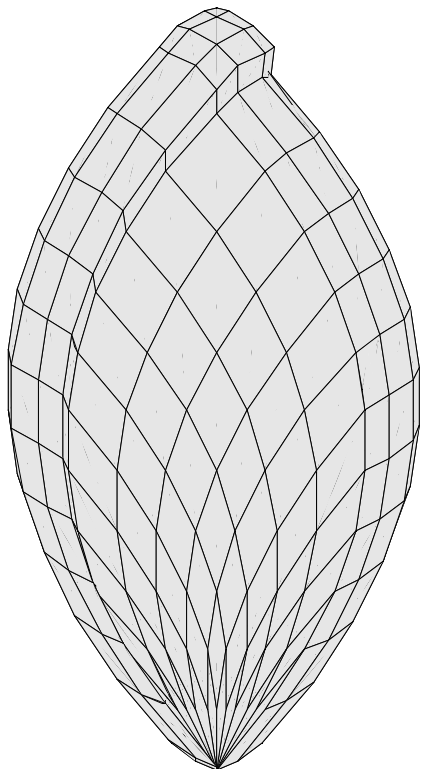


Figura 10: Zonoedro estendido: barca

poliedros que apresentam regularidade [6] e [7] são excelentes referências. Hart, em sua página contém uma extensa bibliografia, bem como uma vasta galeria de poliedros. Os zonoedros polares são objetos matemáticos de fácil construção e excelentes para introdução à programação necessitando apenas de um conhecimento rudimentar da geometria analítica vetorial bem como de coordenadas de poliedros. Uma observação geral quanto a nossa programação é que nossa preocupação de manter o comando dentro da linha de pensamento matemático estrito o que acarreta muito uso do **Table** e do **Map**; evitamos as chamadas funções puras preferindo duplas chaves como **prodEscalar[ ][ ]**. Ao invés de usar **(bas.#)&/@...** e é claro que esta última é mais elegante e diminui o número de chaves finais. O uso do **Table** com duas variáveis também dá como resultado uma matriz e aí usamos o **Flatten[\_1]** para torná-la um vetor. Em muitos casos temos a escolha de ou usar **Map[\_,{2}]**, usar a nível **2**, ou reduzir tudo a vetores e usar **/@..**. Usamos mais o **Map**

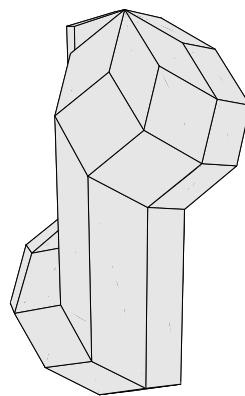


Figura 11: Zonoedro estendido simples

## Conclusão

Os zonoedros polares são objetos matemáticos de fácil construção e excelentes para introdução à programação necessitando apenas de um conhecimento rudimentar da geometria analítica vetorial e construção de poliedros regulares. As funções **zonoedroPolares** e **zonoedroEstendido** dependem de muitos parâmetros dando margem à criatividade. Todos estes comandos se encontram em

<http://www.unemat.br/faciex/professores/nelo/zonoedros.m>

## Referências

- [1] H.S.M. Coxeter, *Regular Polytopes*, 3rd ed. New York: Dover, 1973.
- [2] E.S.Fedorov, "The Symmetry of Regular Systems of Figures." *Zap. Mineralog. Obsc.* (2) **28**, 1-146, 1891. Reprinted as *Symmetry of Crystals. American Crystallographic Assoc.*, 1971.
- [3] G.W. Hart, "Zonohedrification," *Mathematica Journal*, Vol. 7 no. 3, 1999.
- [4] N.W. Johnson, "Convex Solids with Regular Faces," *Canadian Journal of Mathematics*, 18, 1966, pp. 169-200.
- [5] R. Towle, "Graphics Gallery: Polar Zonohedra." *Mathematica J.* **6**, 8-12, 1996.
- [6] E. Weisstein, "Rhombic Triacontahedron." - MathWorld--A Wolfram Web Resource. <http://mathworld.wolfram.com/RhombicTriacontahedron.html>
- [7] M.J. Wenninger, *Dual Models* Cambridge, England: Cambridge University Press, 1983.