

POTW #09-03 Objects In Space

Finding Vectors Sums Of Polyhedra Face Areas

John Snyder, FSA

September 26, 2008

02:00 EDT

Problem

Consider any solid object in space, where each face is flat (a cube is an excellent example). For each face f_i associate a vector \vec{v}_i , where \vec{v}_i points out of the object, perpendicular to the face, and the magnitude of each vector is the area of the associated face, $|\vec{v}_i| = A(f_i)$. What is $\sum_i \vec{v}_i$?

Solution

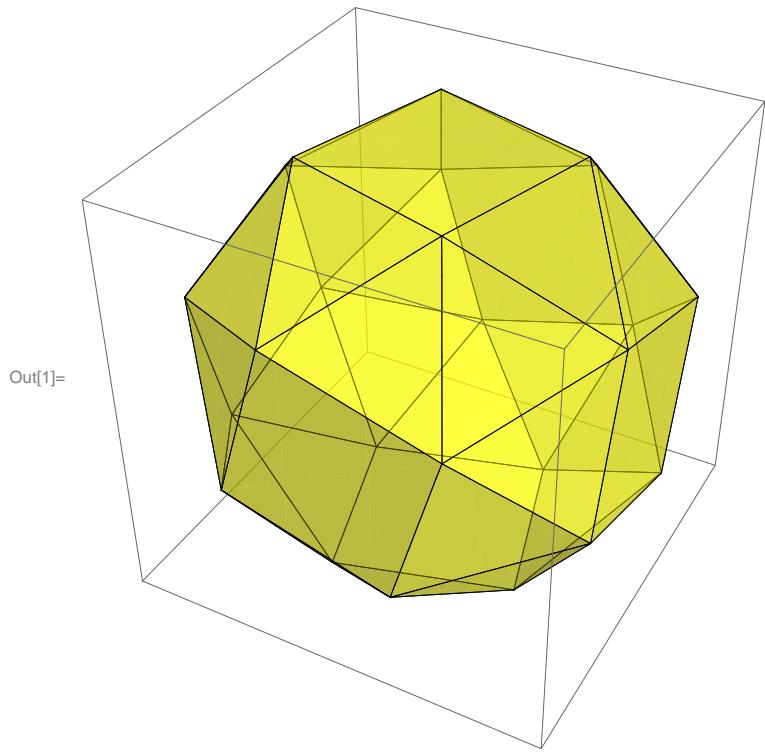
This vector sum in three dimensions is always equal to $\{0, 0, 0\}$.

The computation of the surface area of any polyhedron can be done by using an extension of the familiar formula that the signed area of a triangle is one half the magnitude of the vector cross product of two of its sides. This triangle formula is easily extended to any non-self-intersecting polygon in 2D, and with more effort it can also be extended to 3D. The method rests on breaking any face of a polyhedron into triangles, projecting these triangles into a common plane, applying the vector cross product triangle area formula to each such triangular projection, summing these cross products, and then taking the dot product of the resulting vector sum times a unit normal vector to the face of the polyhedron. This produces a signed calculation of the area of a single face of the polyhedron. This procedure can then be extended over all faces of the polyhedron to obtain the object's total surface area. This methodology is discussed in more detail in the references listed below.

Without getting into the details of a formal proof, this result seems quite logical. Taking cross products of vectors in each planar face of the polyhedron, summing, and then taking a dot product with the unit normal vector would intuitively seem to produce this zero result. We can demonstrate that this is case using *Mathematica* which contains extremely detailed information on 147 different polyhedra. *Mathematica* can tell use the coordinates of each vertex and the vertex numbers on each face of the polyhedra. We can then apply the method outlined above and test that the resulting vector sum is, in fact, $\{0, 0, 0\}$.

We first demonstrate with a single polyhedron, in this case a snub cube.

```
In[1]:= Graphics3D[
  {Opacity[.5], FaceForm[Yellow], PolyhedronData["SnubCube", "Faces"]}, Lighting -> "Neutral"]
```



Here are the coordinates of each vertex. We show only an abbreviated list.

```
In[9]:= Short[vertices = PolyhedronData["SnubCube", "VertexCoordinates"] // ToRadicals // Simplify, 5]
```

Out[9]//Short=

$$\begin{aligned} & \left\{ -\frac{1}{2} \sqrt{\frac{1}{3} \left(4 + \left(199 - 3 \sqrt{33} \right)^{1/3} + \left(199 + 3 \sqrt{33} \right)^{1/3} \right)}, \right. \\ & -\frac{1}{2} \sqrt{\frac{1}{3} \left(4 - \frac{2}{\left(-17 + 3 \sqrt{33} \right)^{1/3}} + \left(-17 + 3 \sqrt{33} \right)^{1/3} \right)}, \quad -\frac{1}{2} \sqrt{\frac{1}{3} \left(2 - \frac{2}{\left(17 + 3 \sqrt{33} \right)^{1/3}} + \left(17 + 3 \sqrt{33} \right)^{1/3} \right)}, \\ & \left. \ll 22 \gg, \frac{1}{2} \sqrt{\frac{1}{3} \left(2 - \frac{2}{\left(17 + 3 \sqrt{33} \right)^{1/3}} + \left(17 + 3 \sqrt{33} \right)^{1/3} \right)}, \right. \\ & \left. \frac{1}{2} \sqrt{\frac{1}{3} \left(4 - \frac{2}{\ll 1 \gg} + \left(-17 + 3 \sqrt{33} \right)^{1/3} \right)}, \frac{1}{2} \sqrt{\frac{1}{3} \left(4 + \left(199 - 3 \sqrt{33} \right)^{1/3} + \left(199 + 3 \sqrt{33} \right)^{1/3} \right)} \right\} \end{aligned}$$

We next see that there are 24 vertices.

```
In[8]:= vertices // Length
```

Out[8]= 24

Next get a list of the vertex numbers, in the same order as the above list, which are contained in each planar face.

```
In[11]:= indices = PolyhedronData["SnubCube", "FaceIndices"]

Out[11]= {{3, 1, 17}, {3, 17, 9}, {3, 19, 2}, {3, 9, 19}, {1, 4, 20}, {1, 20, 11}, {1, 11, 17},
{2, 19, 12}, {2, 18, 4}, {2, 12, 18}, {4, 18, 10}, {4, 10, 20}, {17, 11, 13}, {19, 9, 15},
{18, 12, 14}, {20, 10, 16}, {9, 21, 15}, {11, 23, 13}, {12, 24, 14}, {10, 22, 16},
{13, 23, 7}, {13, 7, 21}, {15, 21, 5}, {15, 5, 24}, {16, 22, 6}, {16, 6, 23},
{14, 24, 8}, {14, 8, 22}, {21, 7, 5}, {23, 6, 7}, {24, 5, 8}, {22, 8, 6}, {1, 3, 2, 4},
{21, 9, 17, 13}, {24, 12, 19, 15}, {10, 18, 14, 22}, {11, 20, 16, 23}, {8, 5, 7, 6}}
```

Mathematica can also tell us the object's surface area computed in the case where the length of the shortest edge is taken to be one.

```
In[22]:= PolyhedronData["SnubCube", "SurfaceArea"]
```

```
Out[22]=  $6 + 8\sqrt{3}$ 
```

```
In[23]:= % // N
```

```
Out[23]= 19.8564
```

The following function uses the technique already discussed to compute the signed surface area of a polyhedron. We apply it to the snub cube.

```
area[polyhedron : StringQ] := Module[{vcor, findices, fsums, fvectors, normals, fareas},
  vcor = PolyhedronData[polyhedron, "VertexCoordinates"];
  findices = PolyhedronData[polyhedron, "FaceIndices"];
  (* append first vertex to end of list *)
  findices = Append[#, First[#]] & /@ findices;
  (* one-half the vector cross product of face vertices *)
  fsums =  $\frac{1}{2} \text{Total}[\text{Cross} @@@ \text{Partition}[vcor[##], 2, 1]]$  & /@ findices;
  (* get two non-parallel vectors in each face *)
  fvectors = Rest[Take[vcor[##], 3]] - Table[First[vcor[##]], {2}] & /@ findices;
  (* get unit normal vectors to each planar face *)
  normals =  $\frac{\text{Cross}[\text{Sequence} @ @ \#]}{\text{Norm}[\text{Cross}[\text{Sequence} @ @ \#]]}$  & /@ fvectors;
  (* get list of areas for each face *)
  fareas = Dot[Sequence @ @ \#] & /@ Transpose[{fsums, normals}];
  (* sum the areas of all planar faces *)
  Total[fareas]
]
```

```
In[26]:= area["SnubCube"] // N
```

```
Out[26]= 19.8564
```

This result matches what *Mathematica* has in its packlet database and indicates that we have correctly implemented the algorithm.

With only a small change this same function can be used to obtain the vector sum of the surface area of each face times a unit normal vector. The following function implements this procedure.

```
In[27]:= test[polyhedron : StringQ] := Module[{vcor, findices, fsums, fectors, normals, fareas},
  vcor = PolyhedronData[polyhedron, "VertexCoordinates"];
  findices = PolyhedronData[polyhedron, "FaceIndices"];
  (* append first vertex to end of list *)
  findices = Append[#, First[#]] & /@ findices;
  (* one-half the vector cross product of face vertices *)
  
$$\text{fsums} = \frac{1}{2} \text{Total}[\text{Cross} @@@ \text{Partition}[vcor[[#]], 2, 1]] & /@ \text{findices};$$

  (* get two non-parallel vectors in each face *)
  fectors = Rest[Take[vcor[[#]], 3]] - Table[First[vcor[[#]], {2}] & /@ findices;
  (* get unit normal vectors to each planar face *)
  
$$\text{normals} = \frac{\text{Cross}[\text{Sequence} @@ \#]}{\text{Norm}[\text{Cross}[\text{Sequence} @@ \#]]} & /@ \text{fectors};$$

  (* get list of areas for each face *)
  fareas = Dot[Sequence @@ #] & /@ Transpose[{fsums, normals}];
  Total[normals fareas]
]
```

We apply the function to the snub cube and obtain a result of {0, 0, 0}.

```
In[30]:= test["SnubCube"] // N
Out[30]= {0., 0., 0.}
```

For a more rigorous test we can apply this procedure to all 147 polyhedra in the *Mathematica* database. First we get a list of the names of these polyhedra; we print only an abbreviated list.

```
In[31]:= Short[names = PolyhedronData[All, "StandardName"], 15]
Out[31]/Short=
{{Antiprism, 4}, {Antiprism, 5}, {Antiprism, 6}, {Antiprism, 7}, {Antiprism, 8},
 {Antiprism, 9}, {Antiprism, 10}, AugmentedDodecahedron, AugmentedHexagonalPrism,
 AugmentedPentagonalPrism, AugmentedSphenocorona, AugmentedTriangularPrism,
 AugmentedTridiminishedIcosahedron, AugmentedTruncatedCube, AugmentedTruncatedDodecahedron,
 AugmentedTruncatedTetrahedron, <<116>>, TriangularCupola, TriangularHebesphenorotunda,
 TriangularOrthobicupola, TriaugmentedDodecahedron, TriaugmentedHexagonalPrism,
 TriaugmentedTriangularPrism, TriaugmentedTruncatedDodecahedron, TridiminishedIcosahedron,
 TridiminishedRhombicosidodecahedron, TriglyrateRhombicosidodecahedron, TruncatedCube,
 TruncatedDodecahedron, TruncatedIcosahedron, TruncatedOctahedron, TruncatedTetrahedron}
```

```
In[32]:= names // Length
Out[32]= 147
```

Finally, we apply our function to all 147 polyhedra and, in each case, we obtain a result of {0, 0, 0}.

References

Berg, Kreveld, Overmars, and Schwarzkopf, *Computational Geometry*, Springer, 2000

Sunday, Dan, "Areas of Triangles and Polygons (2D & 3D)", http://geometryalgorithms.com/Archive/algorithm_0101/#3D%20-Standard%20Formula