Journal for Geometry and Graphics
Volume 23 (2019), No. 2, 157–165.

Euclidean Realizations of Triangulated Polyhedra

Pascal Honvault

LMPA – Laboratoire de Mathématiques Pures et Appliquées Joseph Liouville
Université Littoral Côte d'Opale, EA 2597, F-62228 Calais, CNRS, FR 2956, France
email: pascal.honvault@univ-littoral.fr

Abstract. Let $\mathcal{C} = (d_0, \ldots, d_n)$ be an admissible degree sequence for a triangulated polyhedron $\mathcal{P}_n$ with $n+1$ vertices. We give necessary and sufficient conditions on its Euclidean parameters (angles, lenghts, ...) for beeing realized in the usual 3D-space.

Key Words: polyhedron, combinatorics, triangulation, Euclidean parameters, quaternions

MSC 2010: 52B05, 51N20, 52-04

1. Introduction

In the following, we will deal with polyedra of genus 0 in the usual 3D-space. In order to be buildable, a triangulated polyhedron $\mathcal{P}_n$ with $n+1$ vertices ($n \geq 3$) must satisfy some necessary conditions. First of all, its degree sequence, depending on the numbering of the vertices, must be "admissible" (cf. [4]), which is a combinatorial condition. On the other hand, the lenghts of the edges, the angles of the faces ("internal angles") and between the faces ("external angles") must fulfill some Euclidean requirements. We fix now the notations.

2. Combinatorial conditions

Before to speak about degree sequence, we have to number the vertices. We do it in the following way (cf. [4]). We start with some (positively oriented) face of a polyhedron with $n+1$ vertices $\mathcal{P}_n$ and will denote it by $(0,1,2)$. By induction, we can number the adjacent vertices of 0: if i is still found, then $i+1$ is the point such that $(0, i-1, i)$ is adjacent to $(0, i, i+1)$, for $i = 2$ to $d_0 - 1$. Of course, all the triangles are positively oriented. Then, we can continue this process with the vertex 1 and so on. The corresponding degree sequence is the list $C = (d_0, \ldots, d_n)$ if $\mathcal{P}_n$ has $n+1$ vertices, where $d_i = \deg(i)$ for each $i \in [0, n]$. Obviously, this degree sequence changes when you start from another triangle, but they all represent the same triangulation (cf. [3, 4, 6] for the Schlegel diagram) and are called "equivalent". Some algorithm allows us (see [4]) to find all these equivalent classes by induction, but its complexity is exponential (see [4, 7]). So, it is more efficient to put the results in a database:

ISSN 1433-8157/$ 2.50

`DATABASE polyhedra`.
This base will contain the following tables.

* `TABLE number_classes`

id	*number of vertices*	*number of classes*
1	04	1
2	05	1
3	06	2
4	07	6
...	...	...

which indicates the number of equivalent classes.

* `TABLE degree_sequence_Pn (example for n=5)`

id	*number of the class*	*degree sequence*
1	1	$(3,5,5,4,4,3)$
2	1	$(3,4,5,5,4,3)$
3	1	$(3,5,4,5,3,4)$
4	1	$(4,4,5,3,5,3)$
...	...	...

which gives the degree sequences of each class of $\mathcal{P}_n$.

* `TABLE triangulation_Pn (example for n=5)`

id	*number of the class*	*triangulation*
1	1	$(0,1,2)$
2	1	$(0,1,3)$
3	1	$(0,1,4)$
4	1	$(0,1,5)$
5	1	$(0,3,2)$
6	1	$(1,2,5)$
7	1	$(2,3,4)$
8	1	$(2,4,5)$
9	2	$(0,1,2)$
...	...	...

where one representative triangulation is shown for each equivalent class.

As a result, we are able to respond quickly and efficiently to the following questions:

- What is the number of "different" polyhedra with $n+1$ vertices?
- Which degree sequences are possible for a fixed number of vertices?
- If a degree sequence C is admissible, it is under which triangulation? Here, we have just indicated a triangulation per class, but it is easy to compute the good one from it by a renumbering.

3. Euclidean conditions

Our interest here is the actual construction of the polyhedra $\mathcal{P}_n$. We will suppose that the vertices are numbered and that the corresponding degree sequence C is admissible, i.e., belong to the `TABLE degree_sequence_Pn` of the `DATABASE polyhedra`. Moreover, we have at our disposal the triangulation T obtained from the `TABLE triangulation_Pn`, up to some renumbering. We will call it a *combinatorial polyhedron.*

Thus, we know all the edges $[p,q]$ of $\mathcal{P}_n$, and for each such edge the (positively oriented) triangles (p,r,q) and (p,q,s) adjacent to it. The needed Euclidean parameters for the construction will be those seen in the Section 1, to which must be added the local geometry near each vertex. Indeed, recall that (cf. [5]) the internal and external angles α_i and δ_i at a vertex p of degree d $(i = 1,\dots,d)$ must verify some equations for the local constructibility. In fact, we can compute some of these parameters by the use of the quaternionic algebra in the following way.

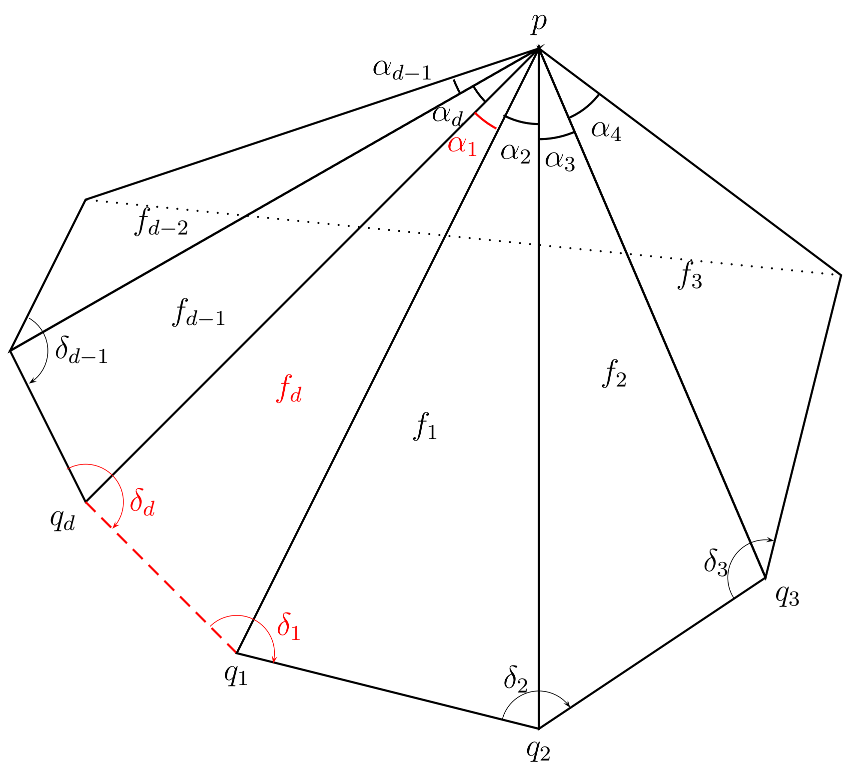

Figure 1: Local polyhedron $S(p)$

Denote by f_i the faces containing p, $\delta_i = \widehat{f_{i-1}, f_i}$ the external angles, and α_i the internal angles for $i = 1,\dots,d$. The unit sphere of centre p intersects $S(p)$ in a union of great circles $\widehat{q_i, q_{i+1}}$, and for $q_1 = (0,1,0,0)$, $q_2 = (0,\cos(\alpha_1),\sin(\alpha_1),0)$ in the canonical base of the space $\mathbb{H}$ of quaternions, we have for $i = 1,\dots,d-1$ (see [5]):

$$(R):\ q_{i+1} = \frac{\sin(\alpha_i - \alpha_{i+1})q_i + \sin(\alpha_{i+1})Q_i q_{i-1} Q_i^*}{\sin(\alpha_i)} \quad \text{with} \quad Q_i = \cos\Big(\frac{\delta_i}{2}\Big) + \sin\Big(\frac{\delta_i}{2}\Big)q_i.$$

This formula permits us to compute q_{i+1} by knowing the values of q_{i-1}, q_i, α_i, α_{i+1}, and δ_i. Here $Q^* = (A,-B,-C,-D)$ refers to the conjugate of $Q = (A,B,C,D)$. The following result gives us the last Euclidean values δ_d, α_1 and δ_1 of the local polyhedron.

Proposition 1. *The local polyhedron $S(p)$ is constructible if and only if*

$$(\Sigma_d)\begin{cases} \alpha_1 = \arccos\left(\dfrac{q_d\, q_1^* + q_1\, q_d^*}{2}\right), \\ \cos(\delta_d)+\sin(\delta_d)q_d = \dfrac{\cos(\alpha_1).(\cos(\alpha_d)+q_dq_{d-1})+q_1.(\cos(\alpha_{d-1})q_d-q_{d-1})}{\sin(\alpha_d)\sin(\alpha_1)}, \\ \cos(\delta_1)+\sin(\delta_1)q_1 = \dfrac{\cos(\alpha_2).(\cos(\alpha_1)+q_1q_d)+q_2.(\cos(\alpha_d)q_1-q_d)}{\sin(\alpha_1)\sin(\alpha_2)}. \end{cases}$$

So, for a vertex of degree d, we have $d-1$ parameters for the internal angles $\alpha_1,\dots,\alpha_{d-1}$, $d-2$ parameters for the external angles $\delta_1,\dots,\delta_{d-2}$, and d parameters for the lenghts of the edges, that is $3d-3$ parameters.

Theorem 1. *Let $\mathcal{P}_n$ be a combinatorial polyhedron and T one of his representative triangulation. Then it is constructible if and only if the following constraints (E_n) are respected:*

1. *("faces" condition): for each triangle (p,r,q) in T, $\widehat{p,r,q}+\widehat{r,q,p}+\widehat{q,p,r}=\pi$.*
2. *("edges" condition): for each edge $[p,q]$ in T, $|p,q|=|q,p|$ and $\widehat{q,p}=2\pi-\widehat{p,q}$.*
3. *("vertices" condition): for each vertex of degree d, the conditions (Σ_d) must be verified for the internal and external angles.*

Proof. These conditions are obviously necessary. We do the converse by induction on n. First, if $n=3$, then $\mathcal{P}_n$ is a tetrahedron, so any star $S(p)$, $p=0,\dots,3$, determines entirely the polyhedron, and the theorem is true in this case. Next, suppose it is also true until a rank $n\geq 3$ and consider a polyhedron $\mathcal{P}_{n+1}$. Let be $d=\min\{d_i\,|\,i=0,\dots,n+1\}$ and suppose without loss of generality that $d_0=d$.
If $d=3$, then the local polyhedron $S(0)$ is a tetrahedron $(0,p,q,r)$. If we remove the vertex 0 and the edges $[0,p]$, $[0,q]$, $[0,r]$, then we obtain a combinatorial polyhedron $\mathcal{P}_n$ which satisfies the faces and the edges conditions, because (p,q,r) is a face of the tetrahedron. Moreover, the vertices conditions remain also true for p,q,r in $\mathcal{P}_n$: the new external (oriented) angles are computed by deletion of the tetrahedron's faces. We deduce, by applying the induction hypothesis, that $\mathcal{P}_n$ is constructible. It remains to glue the tetrahedron to it, which proves that $\mathcal{P}_{n+1}$ is also constructible.
We suppose now that the result is true up to a certain $d\geq 3$ and consider a polyhedron $\mathcal{P}_{n+1}$ with minimal degree $d+1=d_0$. We note p,q,r three consecutive vertices adjacents to 0. Recall that a *"flip"* consists of exchanging the diagonals of a quadrilateral (see, for instance, [3, 6, 9]). We can argue like in the case $d=3$, because we see that $(0,p,q)$ and $(0,q,r)$ are two faces of a tetrahedron with fixed parameters. Indeed, the internal and external angles in 0 permit us to compute the lenght $|p,r|$, as well as the internal angle $\widehat{p,0,r}$, and we obtain a new polyhedron $\mathcal{P}'_{n+1}$ with $n+1$ vertices but of minimal degree d, thanks to the flip. The induction hypothesis tells us that $\mathcal{P}'_{n+1}$ is constructible, and it suffices to apply the same flip for coming back to our original polyhedron. This proves the theorem. □

We also refer the reader to Alexandrov's famous theorem that each convex metric defines a convex polyhedron uniquely [1]. There is a constructive (algorithmic) proof by A.I. Bobenko and I. Izmestiev in [2]. In Theorem 1, the convexity condition is satisfied when the internal angles are between 0 rad and π rad.

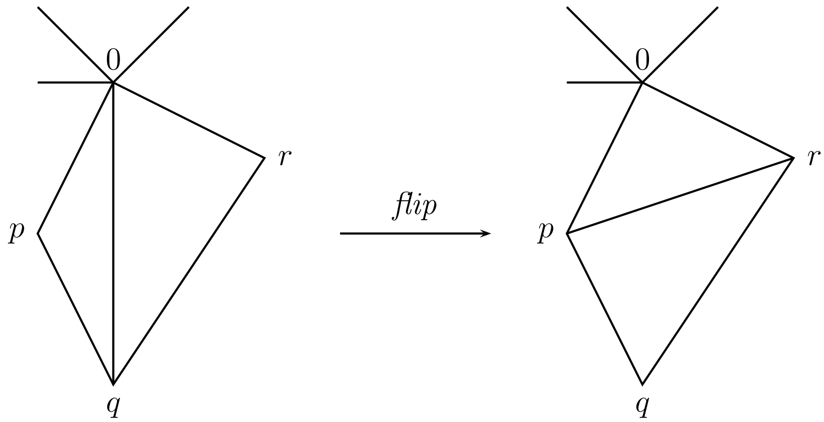

Figure 2: Diagonal flip in a quadrilateral

4. Filling algorithm

We recall that we are dealing with a combinatorical polyhedron $\mathcal{P}_n$ with $n+1$ vertices. Let C be an admissible degree sequence and T the corresponding triangulation. The constraints of Theorem 1 are in practice difficult to verify. Our purpose is to realize these conditions along the way, because the values of the parameters in some vertex would interfere with those of the adjacent points. For instance, the internal angle at some vertex and the lenghts of the incident edges will fix automatically the three internal angles of the faces containing it. Of course, this should be done in accordance with the conditions of Theorem 1. In fact, we will process star by star, and in each star, triangle by triangle, thanks to the quaternionic algebra. Our aim is to compute the quaternions $q_0, \dots, q_n$ assigned to the vertices $0, \dots, n$ of the polyhedron, with the following initialization: $q_0 = (0,0,0,0)$, $q_1 = (0,1,0,0)$ and $q_2 = (0, \cos(\widehat{1,0,2}), \sin(\widehat{1,0,2}), 0)$. The Euclidean values will then be computed easily.

We first construct $S(0)$ by setting the values of $\widehat{2,0,3}$, ..., $\widehat{d-1,0,d}$, as well as $\widehat{0,2}$, ..., $\widehat{0,d-1}$ and $|0,1|$, ..., $|0,d|$. The formula (R) leads to the values of q_3,..., q_d. Here, $d = d_0 = \deg(0)$. Next, suppose that $S(p)$ has already been constructed for a vertex $0 \le p < n$; we want to construct $S(p+1)$. For this purpose, we write the adjacent points of $p+1$ in a list: $(r_1, \dots, r_d)$ where $d = d_{p+1} = \deg(p+1)$. This list is positively oriented around $p+1$, this means that for each $i \in [1, p-1]$, (r_{p+1}, r_i, r_{i+1}) is adjacent to $(r_{p+1}, r_{i+1}, r_{i+2})$. Moreover, r_1 will be the earliest already computed, or any point in the list if all are known! Now, we take q_{p+1} as the origin of our quaternionic algebra, and we compute $q'_1 = q_{r_1} - q_{p+1}$, ..., $q'_d = q_{r_d} - q_{p+1}$ as in the first case. We should just pay attention to the quaternions already computed, that is: if q_{r_j} is known, then we go to the next and so on. As the internal angles, external angles, and lengths will be calculated with these quaternions, we have:

Theorem 2. *The filling algorithm satisfies the conditions of Theorem 1.*

5. Numerical examples

Let us write more precisely the former algorithm. We have not detailed the functions used in the Python program, we prefer instead illustrating the ideas on some polyhedron $\mathcal{P}_7$ with 8 vertices.

5.1. Choice of the degree sequence

We choose some degree sequence $C = [d_0, \ldots, d_n]$ from the `TABLE degree_sequence_Pn` of the `DATABASE polyhedra`. Here, we have chosen $C = [6, 4, 4, 4, 5, 3, 5, 5]$.

5.2. Corresponding triangulation

This degree sequence belongs to some class, namely a combinatorial polyhedron, whose one reprensentative triangulation T can be found in the `TABLE triangulation_Pn`. These triangulation should be eventually reordered, and we will name $0, 1, \ldots, n$ the ordered points corresponding to the degree sequence C. Here we have

$$T = [[0,1,2],[0,2,3],[0,3,4],[0,4,5],[0,5,6],[0,6,1],[1,6,7],[1,7,2],[2,7,3],[3,7,4],[4,7,6],[4,6,5]].$$

We can even represent this polyhedron in a *Schlegel* diagram, that is the triangulation viewed through one of its triangles, here $[0, 1, 2]$.

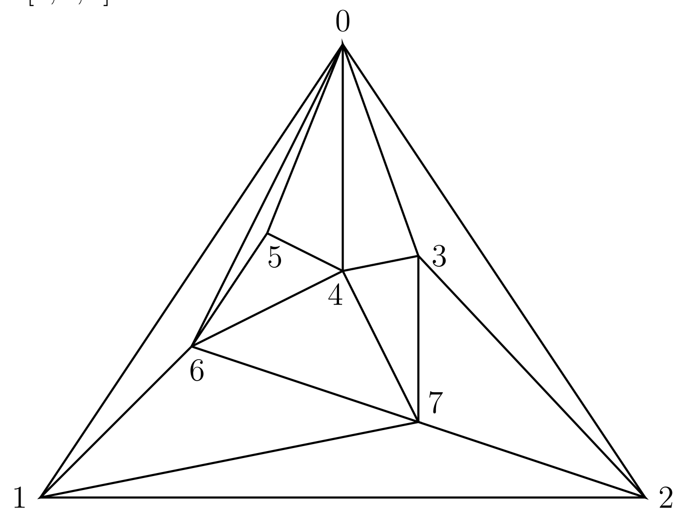

Figure 3: Schlegel diagram of the polyhedron $\mathcal{P}_7$

Pay attention to the fact that only the first triangle $[0, 1, 2]$ is positively oriented on this diagram.

5.3. Computation of the stars

The ordered stars $S(0), \ldots, S(n)$ must be known in view of their treatement in the next subsection. The first star is `S[0]=[[0,1,2],[0,2,3],...,[0,d,1]]` ($d = d_0$).

We construct now the ordered stars $S(p)$ for $p = 1, \ldots, n$. More precisely, we need to compute the stars $S'(0), \ldots, S'(n)$ of the triangles not yet computed. This will be done with the function `star(T)` which returns the list `[S,S']` from the triangulation `T`. In our example, we have:

```
S[0]=[[0,1,2],[0,2,3],[0,3,4],[0,4,5],[0,5,6],[0,6,1]]
S[1]=[[[1, 2, 0], [1, 0, 6], [1, 6, 7], [1, 7, 2]]]
S[2]=[[2, 3, 0], [2, 0, 1], [2, 1, 7], [2, 7, 3]]
S[3]=[[3, 4, 0], [3, 0, 2], [3, 2, 7], [3, 7, 4]]
S[4]=[[4, 5, 0], [4, 0, 3], [4, 3, 7], [4, 7, 6], [4, 6, 5]]
S[5]=[[5, 0, 4], [5, 4, 6], [5, 6, 0]]
S[6]=[[6, 0, 5], [6, 5, 4], [6, 4, 7], [6, 7, 1], [6, 1, 0]]
S[7]=[[7, 1, 6], [7, 6, 4], [7, 4, 3], [7, 3, 2], [7, 2, 1]]
```

whereas:

```
S'[0]=[[0,1,2],[0,2,3],[0,3,4],[0,4,5],[0,5,6],[0,6,1]]
S'[1]=[[1, 6, 7], [1, 7, 2]]
S'[2]=[[2, 7, 3]]
S'[3]=[[3, 7, 4]]
S'[4]=[[4, 7, 6], [4, 6, 5]]
S'[5]=[]
S'[6]=[]
S'[7]=[]
```

5.4. Euclidean values

We browse the stars $S'(0)$, …, $S'(n)$ and, in each star, the ordered list of triangles, in order to compute the representative quaternions $q_0, \dots, q_n$ of the vertices $0, \dots, n$. These will be stored in the array `Q=zeros((4,n+1))` initialized to 0, where for $i = 0$ to n: $q_i =$ `Q[:,i]=[Q[0,i],...,Q[3,i]]` has four coordinates. The Euclidean values are put in the array `V=empty((3,6*(n-1)))` where for $j = 0, \dots, 6(n-1)$, `V[:,j]` represents the Euclidean values of the j^{th} triangle of the liste S. In fact, if this triangle is $\widehat{a,b,c}$ with vertex a, then `V[0,j]` is the external angle $\widehat{a,c}$, `V[1,j]` is the internal angle $\widehat{b,a,c}$, and `V[2,j]` is the length $|a,b|$. We start with $S(0)$.

5.4.1. Filling of the Euclidean values of $S[0]$

We will use the following functions.

`intangle(q,q')`: computes the non oriented angle between the vectors q and q'.
`extangle(q,q',q'')`: returns the oriented angle between the faces (q,q') and (q',q'') oriented by q'.
`prod(q,q')`: calculates the product of the quaternions q and q'.

Moreover, the formula (R) (cf. Section 3) is also applied for the local constructibility at the vertex 0. This is achieved by the function
`recur(q1,q2,beta,delta)` which gives the quaternion

$$\texttt{q3} = \frac{\texttt{sin(alpha-beta).q2 + sin(beta).prod(prod(Q,q1),Q*)}}{\texttt{sin(alpha)}},$$

where `alpha = intangle(q1,q2)` and `Q = cos(delta/2) + sin(delta/2).q2`. We have chosen to ask the user the values of the first external and internal angles, but it is also possible, for instance, to take random values.

```
def init(n,C):  ####initialize star(0) for Pn with degree sequence C
I=array([1,0,0,0])
Q=zeros((4,n+1))  #array of the quaternions with Q[:,0]=[0,0,0,0]
Q[:,1]=[0,1,0,0]
V=empty((3,6*(n-1))) #array of Euclidean values
V[0,0]=input("internal angle 1,0,2:")
Q[:,2]=array([0,cos(V[0,0]),sin(V[0,0]),0])
for i in range(2,C[0]):
V[1,i-2]=input("external angle 0,"+str(i)+":")
V[0,i-1]=input("internal angle"+str(i)+",0,"+str(i+1)+":")
q1=Q[:,i-1]
q2=Q[:,i]
beta=V[0,i-1]
delta=V[1,i-2]
Q[:,i+1]=recur(q1,q2,beta,delta)
V[0,C[0]-2]=intangle(Q[:,C[0]],Q[:,1])
```

```
V[1,C[0]-2]=extangle(Q[:,C[0]-1],Q[:,C[0]],Q[:,1])
V[1,C[0]-1]=extangle(Q[:,C[0]],Q[:,1],Q[:,2])
for i in range(C[0]):
V[2,i]=input("length of the edge 0,"+str(i+1)+":")
Q[:,i+1]=V[2,i]*Q[:,i+1]
return [V,Q]
```

In our example, we tried respectively

$$\alpha = \frac{\pi}{6} \simeq 0.5236 \quad \text{and} \quad \delta = 2.\arcsin(\sqrt{3})/\cos(\pi/12) \simeq 2.224$$

for the first internal and external angles, as well as 1 for the lengths of the edges, and the result for the Euclidean values V[:,0], ..., V[:,5] is:

```
[[ 0.5236  0.5236  0.5236  0.5236  0.5236   0.5236  ]
[ 2.224    2.224    2.224    2.224   2.224  2.224  ]
[ 1        1        1          1       1        1]]
```

which can be verified by another technique. In fact, for a regular local polyhedron with n faces, the internal angle α and the external angle δ are bound by the formula $\cos(\pi/n) = cos(\alpha/2).\sin(\delta/2)$.

5.4.2. Filling of the Euclidean values of $S[1]$, ..., $S[n]$

Once again, n, C, and T are the inputs of the program.

```
####################
### MAIN PROGRAM ###
####################

[V,Q]=init(n,C)
S=star(T,n)
[S,S']=sorted_lists(S,C)   # in the case there are not sorted
counter=C[0]               # number of quaternions already computed
length=C[0]                # length of S[0]
beta=0                     # provisional value of V[0,]
delta=0                    # provisional value of V[1,]
v2=0                       # provisional value of V[2,]
for i in range(1,n+1):     # we are looking for the quaternions
for j in range(C[i]):  # in S[i] not yet found
if (S[i][j] in Sdim[i]) and (S[i][j][2]>compteur):  # new
compteur+=1    # quaternion to compute
########### requested values ##########
delta=float(input('angle externe '+str(i)+','
+str(S[i][j][1])+': '))
beta=float(input('angle interne '+str(S[i][j][1])+','
+str(i)+','+str(S[i][j][2])+': '))
v2=float(input('longueur arete '+str(i)+','
+str(S[i][j][2])+': '))
##### calculus of the quaternion ######
q1=Q[:,S[i][j-1][1]]-Q[:,i]
q1=(1/long_arete(q1))*q1
q2=Q[:,S[i][j][1]]-Q[:,i]
q2=(1/long_arete(q2))*q2
Q[:,compteur]=Q[:,i]+v2*recur(q1,q2,beta,delta)
```

```
### filling of the tabular V of Euclidean values ###
for j in range(C[i]):
q1=Q[:,S[i][j][1]]-Q[:,i]
q2=Q[:,S[i][j][2]]-Q[:,i]
if j ==C[i]-1:
q3=Q[:,S[i][0][2]]-Q[:,i]
else:
q3=Q[:,S[i][j+1][2]]-Q[:,i]
V[0,longueur+j]=intangle(q1,q2)
V[1,longueur+j]=extangle(q1,q2,q3)
V[2,longueur+j]=long_arete(q2)
print(V[0:3,longueur:longueur+C[i]])
longueur+=C[i]

############################
# END OF THE MAIN PROGRAM #
############################
```

In our example, the program just ask the values of the external angle $\widehat{1,6}$, the internal angle $\widehat{6,1,7}$, and the length $|1,7|$ in order to compute the last quaternion q_7. Then all the Euclidean values are computed. As another verification, we have also tested this algorithm on the regular octahedron and obtained the expected values, that is 1.047 rd ($\simeq \pi/3$) for the internal angles, and 1.9106 rd for the external angles.

Our algorithm can also admit no proper realization. For instance, a two-fold covered four-sided pyramid (by putting one star into the other so that two opposite vertices coincide), or even two four-fold covered pair of two regular triangles sharing an edge. The only limits are those of Theorem 1.

References

[1] A.D. Alexandrov: *Convex polyhedra.* Springer, 2010.

[2] A. Bobenko, I. Izmestiev: *Alexandrov's theorem, weighted Delaunay triangulations, and mixed volumes.* Ann. Inst. Fourier (Grenoble) **58**, no. 2, 447–505 (2008).

[3] J.S.L. Devadoss, J. O'Rourke: *Discrete and computational geometry.* Princeton University Press, 2011.

[4] P. Honvault: *Combinatorics of triangulated polyhedra.* J. Geometry Graphics **22**/1, 41–47 (2018).

[5] P. Honvault: *Similarities of the sphere.* Forum Geometricorum **16**, 313–316 (2016).

[6] S. Negami: *Diagonal flips of triangulations on surfaces, a survey.* Yokohama Math. J. **47** (1999).

[7] D. Rorabaugh: *On-Line Encyclopedia of Integer Sequences A253882.* 2015.

[8] W.T. Tutte: *A census of planar triangulations.* Canad. J. Math. **14**, 21–38, 18, 21, 45 (1962).

[9] K. Wagner: *Bemerkungen zum Vierfarbenproblem.* Jahresber. Dtsch. Math.-Ver. **46**, Abt. 1, 26–32 (1936).

Received June 17, 2019; final form October 21, 2019