Axe median.

Laurent Cancé Francis

06/09/2017

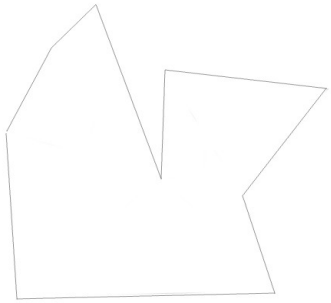
14/09/2017

I. Procédé

Dans l'élaboration de structures de maillages, l'axe médian est utile à de nombreuses applications.

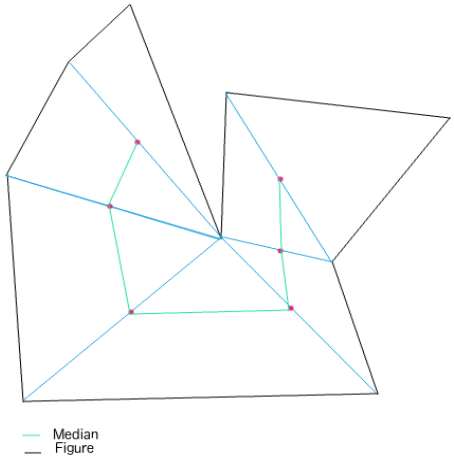
II. Présentation du problème

Il s'agit de définir la structure médiane à une figure 2D, dans le cas présent :



III. Solution

Le calcul de la surface par triangularisation donne l'axe médian par le simple formatage des longueurs de la figure la triangularisation de la figure par le sens de marche de définition pouvant suivre n'importe quel algorithme de triangularisation, pour calculer l'axe médian, il suffit de ne considérer que les segments « ajoutés » à la figure pour en définir la surface, par une méthode itérative définissant des triangles adjacents d'un signe d'orientation à la normale du plan.



IV. Dimensions

Le calcul de l'axe médian pour un maillage 3d revient à définir des tétraèdres concaves ou convexes ; l'algorithme suivant donne une triangularisation pour une figure

quelconque en 2D dans un plan de l'espace de dimension 3.

```
n=0;
count=0;
dec=Figure->Length;
precedent=0;
el=Figure->First;

while ((dec>1)&&(count<2*Figure->Length))
{
	i1=el->data;                                          // i1,i2 segment de la figure en cours de traitement
	if (el->next) i2=el->next->data; else i2=Figure->First->data;
	u=vertices[i2]-vertices[i1];
	U=N^u;
	U.Normalise();
	d=-(U|vertices[i1]);                                  // U,d plan orthogonal à la figure contenant [i1,i2]
	bool sec=true;
	if (precedent>=0) sec=PlanSecant(vertices,Figure,U.x,U.y,U.z,d,i1,i2);

	if (sec)
	{
		if (el->prev) i0=el->prev->data; else i0=Figure->Last->data;
		ss=(U|vertices[i0])+d;                            // sommet précédent suit une concavité évidente
		if (ss<=0.0f) valid=(!Intersectionnant(vertices,Figure,N,i0,i2,false));
		else valid=false;                                 // si génère des intersections

		if (valid) valid=(!PointsInterieurTriangle(vertices,Figure,N,i0,i1,i2));
                                                          // si le triangle recouvre d'autres segments
		if (valid)
		{
			AjouteTriangle(i0,i1,i2);
			precedent=0;
			el=el->prev;
			if (!el) el=Figure->Last;
			SupprimeIndexActuel(Figure);
			dec=Figure->Length;
			if (n-1>=0) n=n-1; else n=Figure->Length-1;
			count=0;
		}
		else
		{
			n++;
			dec=Figure->Length;
			precedent=-1;
			el=el->next;
			if (!el) el=Figure->First;
		}
	}
	else
	{
		precedent=0;
		n++;
		dec--;
		el=el->next;
		if (!el) el=Figure->First;
	}

	if (n>=Figure->Length) n=0;
	count++;
}

if (Figure->Length>2) AjouteFigureConvexeTriangles(Figure) ;
```

avec :

N : Normale du plan
vertices : Liste de sommets indéxés
Figure : Liste d'index de sommets définissant la figure