Bref guide d'utilisation du logiciel SPAF (Surfaces Par Agencement de Faces) Version Java / Processing

SPAF est un petit programme conçu pour créer et pour manipuler des figures 3D constituées de faces polygonales. L'entrée de ces figures se fait sous forme d'une suite d'instructions, où l'on donne une liste de sommets (via leurs coordonnées) et de faces (via une liste ordonnée des sommets qui les définissent). Par la suite, cependant, on accède à une plus grande liberté de visualisation et de manipulation de ces figures.

Installation

Précisons tout d'abord que tous les logiciels nécessaires à l'installation sont gratuits.

Il faut d'abord vous assurer que ***Java*** est installé sur votre ordinateur. Si ce n'est pas le cas, vous pouvez aller à l'adresse

https://www.java.com/fr/download/

pour télécharger le programme d'installation.

Lorsque ***Java*** est présent sur votre ordinateur, vous pouvez installer l'environnement de programmation ***Processing***, à l'aide du logiciel disponible à l'adresse

https://processing.org/download/

Jusqu'à nouvel ordre, vous devrez installer la version 2.2.1 de ***Processing***. En effet, la version 3.0 est encore en version beta, et SPAF devra être adapté pour fonctionner correctement avec cette version.

Il faut ensuite installer dans ***Processing*** la bibliothèque d'interface ***ControlP5***, que vous trouverez à l'adresse

http://www.sojamo.de/libraries/controlP5/

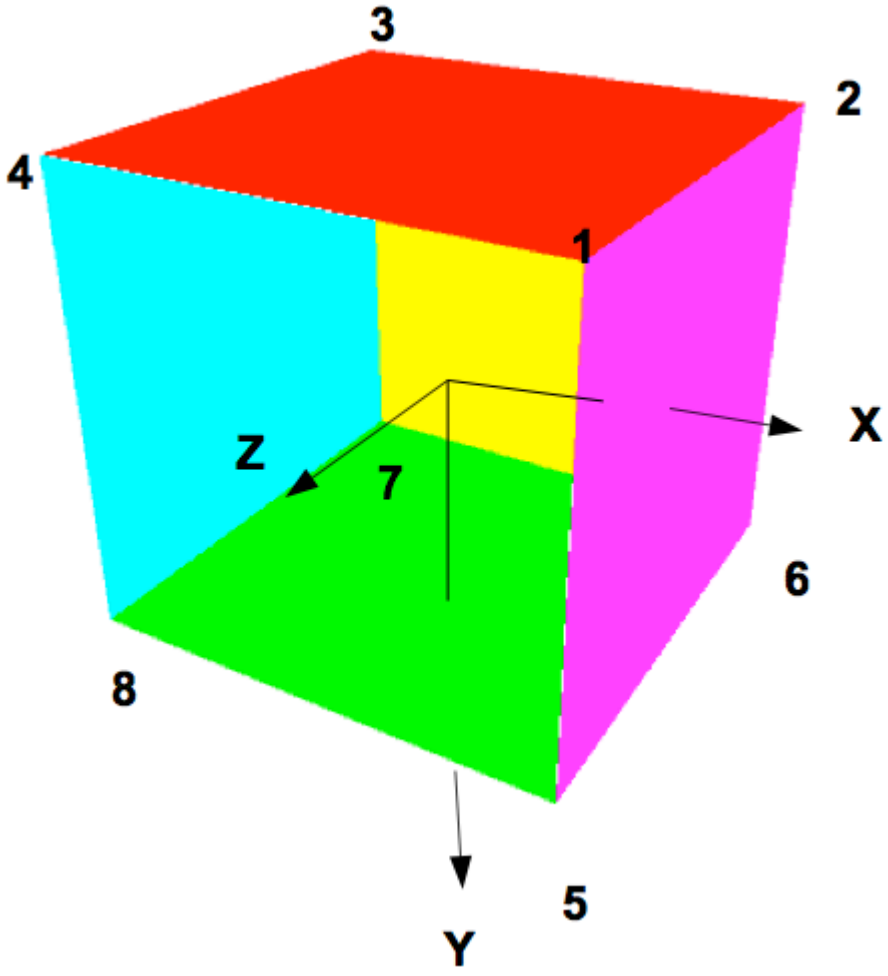
Il ne vous restera plus qu'à télécharger le logiciel ***SPAF*** , à l'adresse

http://www.gruteam.uqam.ca/SPAF/index.htm

et à l'ouvrir dans ***Processing***.

Notre première figure

Comme première figure, nous allons tracer un cube. Comme nous le savons tous, un cube a 8 sommets (numérotés de 1 à 8 sur le document 1 ci-dessous) et 6 faces (dont 5 sont colorées dans ladite figure). Nous avons aussi indiqué le système de coordonnées utilisé par SPAF, avec l'axe des x pointant vers la droite, l'axe des y dirigé vers le bas, et l'axe des z sortant de l'écran. Nous centrerons le cube à l'origine de notre système d'axes, et nous choisirons l'échelle des axes de telle sorte que les coordonnées du sommet 5 soient (1,1,1).



Document 1. La figure représentée dans le système de coordonnées de SPAF

Voyons maintenant comment communiquer ces informations à SPAF. SPAF se présente sous la forme d'un dossier (nommé SPAF) contenant 3 fichiers : *SPAF.pde*, *Face.pde* et *Figure.pde*. Pour ouvrir SPAF, il nous suffit de double-cliquer sur *SPAF.pde*. Nous obtenons alors une fenêtre semblable à celle du document 2, à gauche.

Avant de faire quelque modification que ce soit, nous allons renommer notre projet : nous choisissons l'item « Save as… » du menu « File » de Processing, et nous indiquons le nom

sous lequel nous voulons enregistrer notre Figure (*Cube*, par exemple). L'effet de tout ceci sera la création d'un nouveau dossier (nommé *Cube*) contenant 3 fichiers : *Cube.pde* (obtenu en renommant le fichier *SPAF.pde*), *Face.pde* et *Figure.pde*. Comme nous travaillerons uniquement dans le fichier *Figure.pde*, nous cliquons sur l'onglet correspondant (Figure) et nous sommes prêts à commencer.

Document 2. SPAF tel qu'il apparaît à son ouverture (à gauche)
et après avoir enregistré sous le nom de Cube (à droite)

La définition de notre figure se fera au sein de la procédure « definirFigure », au moyen de déclarations

definirSommet(numéroSommet, x, y, z);

et

definirFace(numéroFace, NoSommet1, NoSommet2, NoSommet3, NoSommet4);

On pourra colorer chaque face au moyen de l'invocation

couleurFace(numéroFace, rouge, vert, bleu)

où chacune des composantes colorées (rouge, vert et bleu) peut prendre des valeurs entre 0 et 255.

Notez que, malheureusement, notre environnement est très sensible aux erreurs syntaxiques : l'oubli d'un seul caractère, que ce soit une parenthèse ou un point-virgule, pourrait avoir des conséquences étranges et catastrophiques.

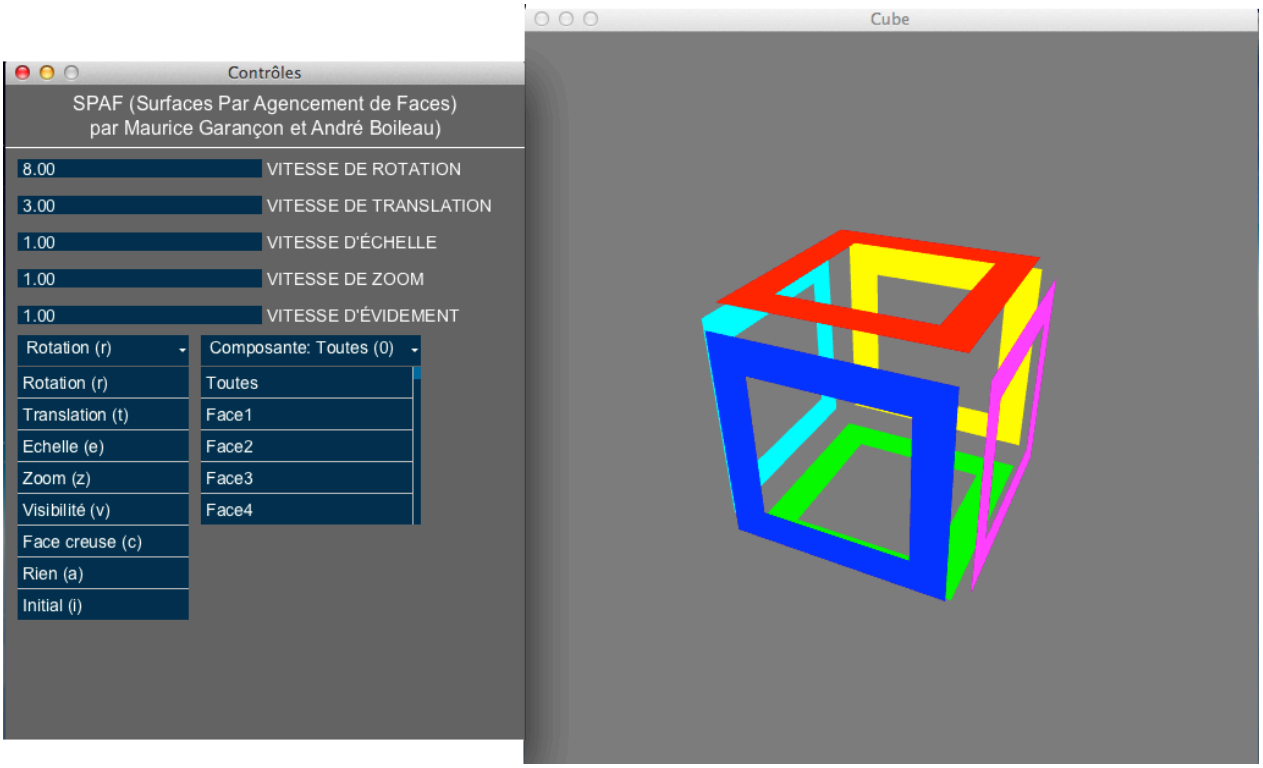
Le document 3 montre la définition finale de notre cube, et nous devons expliciter certaines choses. Tout d'abord la commande « *pixelsParUnite*(100) », qui sert à établir qu'une de nos unités (rappelons que chaque arète de notre cube mesure 2 unités) correspondra à 100 pixels à l'écran. Et enfin on remarque que la notation « // » indique que tout ce qui suit sur la ligne sera considéré comme un simple commentaire, dans ce cas pour nommer la couleur définie pas ses trois composantes.

```
void definirFigure(){
  pixelsParUnite(100);

  definirSommet(1, 1, -1, 1);
  definirSommet(2, 1, -1, -1);
  definirSommet(3, -1, -1, -1);
  definirSommet(4, -1, -1, 1);
  definirSommet(5, 1, 1, 1);
  definirSommet(6, 1, 1, -1);
  definirSommet(7, -1, 1, -1);
  definirSommet(8, -1, 1, 1);

  definirFace(1, 1, 2, 3, 4);
  definirFace(2, 5, 6, 7, 8);
  definirFace(3, 1, 4, 8, 5);
  definirFace(4, 3, 4, 8, 7);
  definirFace(5, 1, 5, 6, 2);
  definirFace(6, 2, 3, 7, 6);


  couleurFace(1, 255, 0, 0);       // rouge
  couleurFace (2, 0, 255, 0);      // vert
  couleurFace (3, 0, 0, 255);      // bleu
  couleurFace (4, 0, 255, 255);    // cyan
  couleurFace (5, 255, 0, 255);    // magenta
  couleurFace (6, 255, 255, 0);    // jaune
}
```

Document 3. Description de notre cube dans SPAF

Pour faire apparaître la figure définie au document 3, il suffit de cliquer sur le petit triangle pointant vers la droite, situé en haut et à droite de la fenêtre. Deux nouvelles fenêtres (voir document 4) apparaissent alors : l'une où apparaît une représentation de notre figure, tandis que l'autre montre l'ensemble des commandes mises à notre disposition.

Au départ, notre figure ne paie pas de mine : elle ressemble à un simple carré bleu centré dans sa fenêtre. Nous savons qu'un cube se cache derrière cette face bleue, et nous allons le révéler à l'aide de la commande « Rotation (r) » du menu des actions (à gauche), appliqué à « Composantes Toutes (0) » du menu de droite. Les déplacements de notre souris avec son bouton enfoncé dans la fenêtre de la figure produisent alors les rotations désirées. Soulignons que nous pouvons avoir recours à la glissière « VITESSE DE ROTATION » pour modifier la réactivité de nos rotations.



Document 4. Une représentation de notre cube, obtenue via les commandes à gauche

Notons en passant les caractères entre parenthèses dans les menus, qui agissent comme des équivalents clavier des items correspondants des menus. Soulignons aussi que certains items du menu des actions peuvent s'appliquer à chacune des composantes (y compris à toutes celles-ci), tandis que d'autres ne s'appliquent qu'à la totalité de celles-ci : à vous de découvrir lesquelles…

Pour obtenir le document 1, il faut faire disparaître la face bleue, qui a été la troisième face à être définie. Il suffit alors de choisir « Visibilité (v) » appliquée à « Face3 », puis à faire un clic dans la fenêtre de la figure. Un autre clic la fera réapparaître.

En choisissant « Echelle (e) » appliqué à toutes les composantes, on peut faire varier simultanément la taille de toutes les faces, toujours en déplaçant notre souris avec son bouton enfoncé dans la fenêtre de la figure. On pourra de même utiliser la commande « Face creuse (c) » pour obtenir la figure représentée à droite du document 4.

Avant de poursuivre la lecture de ce guide, nous vous suggérons de continuer à expérimenter les diverses commandes disponibles dans SPAF, que ce soit à partir du *Cube* qu'avec les autres exemples fournis avec SPAF. Amusez-vous bien !

La description des figures dans SPAF

La description des figures SPAF se fait dans le fichier « Figure », qui comporte au départ trois procédures : **definirFigure**, **initialiserFigure** et **redefinirCertainsSommets**. Pour décrire lesdites figures, on pourra utiliser non seulement les primitives SPAF, mais aussi toute la puissance de **Processing** (qui est lui-même une extension du langage de programmation **Java**). Mais ceci ne devrait pas effrayer le débutant : on peut définir des figures très intéressantes sans grande connaissance de la programmation. Cependant, il faut minimalement se rappeler que nos instructions devront être séparées par des point-virgules. Par contre, des connaissances mathématiques seront, en général, nécessaires…

La description des figures SPAF se fait donc en **trois étapes** :

1. Dans un premier temps, on décrit les sommets, les faces (et leur couleur) et les barres de défilement de notre figure dans la procédure **definirFigure**.
2. Par la suite, on décrit l'état initial de notre figure dans la procédure **initialiserFigure**. On peut voir cette étape comme une utilisation par programmation du menu des actions pour décrire l'état initial voulu de notre figure. Notons que cette procédure peut être vide (c.-à.-d. sans aucune commande) quand on ne désire pas modifier l'état initial de notre figure.
3. Enfin, dans la procédure **redefinirCertainsSommets**, on décrira l'effet de nos barres de défilement sur nos sommets. Cette procédure restera vide si nous n'avons défini aucune barre de défilement.

Primitives de SPAF pouvant être utilisées à l'étape 1, dans la procédure « definirFigure ».

pixelsParUnité(x)

Fixe le nombre de pixels pour représenter une unité (où x est un nombre réel).

definirSommet(numéro du sommet, x, y, z)

x, y, z sont les coordonnées du sommet.
Si on veut n sommets, ils doivent absolument être numérotés de 1 à n, même s'ils ne sont pas définis dans l'ordre

definirFace(numéro de la face, numéro du sommet 1, … , numéro du sommet k)

Les numéros des sommets définissant la face doivent être énumérés dans l'ordre d'un « parcours de la frontière » de la face.

Notons que le nombre de sommets doit être compris entre 3 et 20. Nous verrons plus loin comment définir une face comportant plus de 20 sommets.

couleurFace(numéro de la face, r, v, b)

Les composantes (rouge, vert, bleu) de la couleur sont des réels entre 0 et 255.

creerGlissiereG1(nom de la glissière, a, b, c) et **creerGlissiereG1(a, b, c)**

Crée une glissière dans le tableau de contrôle. **a** est la borne gauche, **b** la borne droite et **c** la valeur initiale à la création.

La valeur courante est en permanence disponible dans la variable **valG1**.

Si le nom de la glissière est spécifié, il sera utilisé pour l'identifier dans le tableau de contrôle. Dans le cas contraire, le nom « G1 » sera utilisé.

On peut créer deux autres glissières **creerGlissiereG2** et **creerGlissiereG3** selon le même principe, et leurs valeurs seront **valG2** et **valG3**

Primitives de SPAF pouvant être utilisées aux étapes 2 et 3, dans les procédures « initialiserFigure » et « redefinirCertainsSommets ».

Si on le désire, on peut redéfinir la position de certains sommets, ce qui aura pour effet de modifier les faces correspondantes, et donc la figure associée. Pour ce faire, on peut faire appel aux fonctions *xSommet*, *ySommet* et *zSommet* donnant les coordonnées des divers sommets.

redefinirSommet(i, x, y, z)

Change les coordonées du sommet numéro **i** en x, y, z. À l'étape 3, les valeurs utilisées pour **x**, **y** et/ou **z** seront des expressions faisant appel aux variables **valG1**, **valG2** et/ou **valG3** des glissières. À l'étape 2, ces variables auront encore leurs valeurs initiales.

xSommet(i), ySommet(i), zSommet(i)

Fonctions retournant respectivement les coordonnées (redéfinies ou non) en x, y, z du sommet **i**.

Poursuivons en décrivant les commandes pouvant s'appliquer tant aux faces individuelles qu'à toute la figure. Quand on désire qu'une de ces commandes s'applique à toutes les faces, il suffit de l'employer avec un numéro de face égal à 0 (zéro).

fixeEchelle(i, valeur)

Fixe l'échelle de la face numéro **i** à la valeur spécifiée, un réel.

fixeVisible(i, valeur)

Fixe la visibilité de la face numéro **i** à la valeur spécifiée, un booléen (*true* ou *false*).

fixeTailleCreux(i, valeur)

Lorsqu'on évide une face, l'évidemment est homothétique à la face elle-même. La valeur (entre 0 et 1) est en fait le rapport d'homothétie utilisé pour la face numéro **i**.

integrerDansFigure(i, valeur)

Indique si on veut que la face numéro **i** fasse partie (valeur = *true*) ou non (valeur = *false*) de la figure. Cette commande sert dans les cas où notre figure comporte un nombre variable de faces (dépendant, par exemple, selon l'état de certaines glissières). Dans la définition initiale, on est contraint de définir le nombre maximal de faces possibles, car on ne peut créer par la suite de nouvelles faces.

changerCouleur(i, rouge, vert, bleu)

Pour déterminer, après la création initiale, la couleur de la face numéro **i**, en spécifiant ses composantes rouge, verte et bleue. Notez ici que **i** ne peut prendre la valeur 0, et donc qu'on ne peut appliquer ce changement de couleur à toutes les faces. Devinez pourquoi!

Les commandes précédentes servaient à donner des valeurs à certains paramètres définissant les faces de notre figure. Mais on peut aussi vouloir connaître la valeur de ces paramètres sans vouloir modifier celle-ci : c'est précisément le rôle des fonctions ci-dessous.

donneEchelle(i)

Cette *fonction* retourne l'échelle de la face numéro **i**.

donneVisible(i)

Cette *fonction* retourne *true* si la face numéro **i** est visible, et *false* sinon.

donneTailleCreux(i)

Cette *fonction* retourne le rapport d'homothétie entre le creux et de la face numéro **i**.

donneDansFigure(i)

Cette *fonction* retourne *true* si la face numéro **i** fait partie de la figure, et *false* sinon.

donneRouge(i) ***donneVert(i)*** ***donneBleu(i)***

Ces *fonctions* retournent les composantes de couleur de la face numéro **i**.

Passons maintenant à la description de commandes représentant des transformations globales s'appliquant à toute la figure.

fixeTranslationX(valeur) et **fixeTranslationY(valeur)**
ajouteTranslationX(valeur) et **ajouteTranslationY(valeur)**

Translate toute la figure (selon l'axe des X ou des Y) d'une distance spécéfiée par la valeur. Dans le cas de **fixeTranslation**, la translation s'effectue depuis la position initiale de la figure. Dans le cas de **ajouteTranslation**, la translation s'effectue depuis la position précédente de la figure.

fixeRotationX(valeur) et **fixeRotationY(valeur)**
ajouteRotationX(valeur) et **ajouteRotationY(valeur)**

Fait tourner toute la figure autour de l'axe des X ou des Y d'un angle dont la valeur est donnée en degrés. Dans le cas de **fixeRotation**, la rotation s'effectue depuis la position initiale de la figure. Dans le cas de **ajouteRotation**, la rotation spécifiée par la valeur s'ajoute aux rotations précédentes cumulatives de la figure.

Les commandes précédentes servaient à donner des valeurs à certains paramètres globaux. On peut aussi vouloir connaître la valeur de ces paramètres sans vouloir les modifier : c'est précisément le rôle des fonctions ci-dessous.

***donneTranslationX*()** et ***donneTranslationY*()**

Retourne la composante (en X ou en Y) de la translation globale.

***donneRotationX*()** et ***donneRotationY*()**

Retourne l'angle de rotation autour de l'axe des X ou de l'axe des Y.

La commande suivante permet de prendre une « photo » de la figure, et de la sauvegarder dans le format spécifié dans le nom.

prendrePhoto(nomCompletFichier)

Le fichier résultant est enregistré dans le dossier comportant la description de la figure. On peut se servir de cette commande pour enregistrer plusieurs vues de notre figures qui pourront par la suite être assemblées pour former une animation *QuickTime* ou autre. Exemples de **nomCompletFichier** : "photoFigure.jpeg" ou même "figure " + no + ".png" (où « no » est un nom de variable).

Nous avons vu que l'environnement **Processing** manquait d'outils de mise au point. Les commandes suivantes, bien qu'assez primitives, peuvent nous aider à mettre au point les instructions décrivant nos figures.

affiche(valeur) et **afficheLigne(valeur)**

Affiche la valeur fournie dans la console, ce rectangle noir au bas de la fenêtre d'édition **Processing**. La valeur affichée peut être un nombre ou une chaîne de caractères. **afficheLigne** fait suivre l'affichage d'un saut de ligne.

quitterProgramme()

Interrompt l'exécution du programme (traçant la figure), afin de permettre la consultation des informations affichées par les commandes précédentes.

Éléments du langage Java pouvant être utiles pour décrire des figures plus complexes.

conditionnelle
if (condition) {instructions si condition vérifiée} else {instructions si condition non vérifiée}

boucle avec compteur
for (initialisation; condition; mise à jour) {instructions}

boucle avec condition
while (condition) {instructions}

À titre d'exemple, voyons maintenant comment on peut utiliser un vecteur de Java pour définir une face avec plus de 20 sommets, disons avec 100 sommets dont les numéros varient entre 101 et 200. On définit tout d'abord notre vecteur comme suit

```
int [ ] mon_vecteur = [101, 102, … , 199, 200];
```

puis on utilise une forme spéciale de la procédure « definirFace »

```
definirFace(numero, mon_vecteur);
```

On peut même généraliser en supposant que les numéros des sommets sont compris entre **a** et **b**, où **a** et **b** sont des variables entières définies précédemment. On définit alors notre vecteur comme suit

```
int mon_vecteur = new int [b-(a-1)];
for (int k = a; k<=b; k++) {mon_vecteur[k-a]=k;};
```

et on termine de la même façon par

```
definirFace(numero, mon_vecteur);
```

Vous pourrez trouver dans les exemples inclus avec SPAF d'autres utilisations du langage Java pour définir des figures…

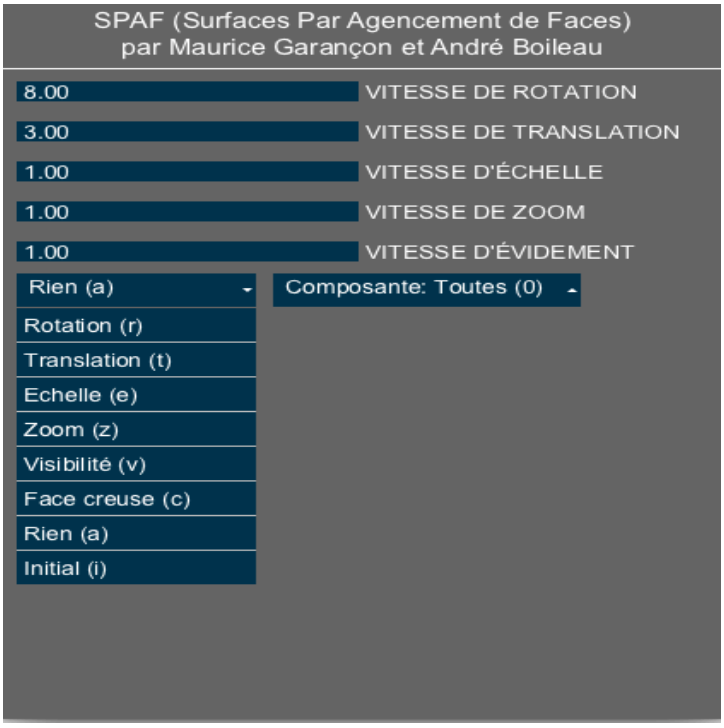
L'exploration des figures dans SPAF

Une fois décrite la figure, on peut en commander l'affichage, que ce soit par un clic sur le petit triangle  de la fenêtre d'édition, par le choix de l'item « Run » du menu « Sketch », ou par son équivalent clavier ⌘R. C'est alors que le plaisir commence…

En plus de la fenêtre graphique (où la figure est représentée) apparaît la fenêtre des commandes (voir le document 5), qui va nous permettre d'interagir dynamiquement avec notre figure. Cette fenêtre comporte trois zones :

- Les glissières (en haut)
  Dans le cas illustré, cette zone ne comporte que des glissières affectant la vitesse d'exécution de certaines commandes. Quand une commande donnée est sélectionnée, la glissière correspondante peut être modifiée avec la souris ou avec le clavier : « - » pour en diminuer la valeur, et « + » ou « = » pour l'augmenter.
  De plus, comme on l'a vu précédemment, cette zone peut accueillir jusqu'à trois glissières supplémentaires définies par l'utilisateur.
- Le menu déroulant des commandes (en bas, à gauche)
  Chacune de ces commandes peut être choisie avec la souris ou avec le clavier, en tapant la lettre entre parenthèses. L'effet de ce choix peut être immédiat (Initial, Rien), nécessiter un clic de souris (Visibilité), un appui prolongé sur un bouton de la souris(Zoom), ou un glisser de souris (Rotation, Translation, Echelle, Face creuse).

- Le menu déroulant des composantes (en bas, à droite)
  À l'exception de certaines commandes (Rotation, Initial), qui s'appliquent seulement à l'ensemble des composantes, ce menu permet de spécifier si les commandes s'appliqueront à toutes les composantes/faces (c'est-à-dire à toute la figure) ou bien seulement à une composante/face particulière. Les composantes/faces peut être choisies avec la souris, et les composantes numérotées de 0 à 9 peuvent même être choisies avec le clavier, en tapant le chiffre correspondant.



Document 5. La fenêtre des commandes

Décrivons maintenant brièvement l'effet de chacune de ces commandes.

Rotation

Fait tourner toute la figure autour de l'axe des X ou des Y, en fonction du glisser de la souris. Cette commande ne peut s'appliquer à une composante/face particulière : elle affecte globalement toute la figure.

Translation

Effectue une translation en fonction du glisser de la souris. Cette commande peut s'appliquer à une composante/face particulière ou à toute la figure. Quand elle s'applique à toute la figure, la translation se fait selon un vecteur du plan X-Y, en fonction du glisser de la souris. Quand elle s'applique à une face seulement, le déplacement s'effectue selon une direction perpendiculaire à la face.

Echelle

Effectue une homothétie (dilatation ou contraction) en fonction du glisser de la souris. Le centre d'homothétie est le centre de gravité de tous les sommets (de la figure ou des la composante/face choisie).

Zoom

Effectue un *zoom in*, quand le bouton gauche est enfoncé, ou un *zoom out*, quand le bouton droit est enfoncé. Un mouvement de la souris change le centre du zoom.

Visibilité

Après avoir choisi cette commande et la composante visée, un clic rendra invisible la composante en question si elle est visible, et la rendra visible si elle est invisible.

Face creuse

Après avoir choisi cette commande et la composante visée, un glisser de souris augmentera ou réduira un trou dans la composante/face en question (ou simultanément dans toutes les faces).

Rien

Une fois cette commande choisie, aucune action de la souris dans la fenêtre de la figure n'est prise en compte. Cependant, on peut encore utiliser la souris pour sélectionner des commandes ou modifier des glissières.

Initial

Cette commande fait retracer la figure dans sa position originale, à la fin des étapes 1 (**definirFigure**) et 2 (**initialiserFigure**), comme si l'utilisateur n'avait jamais débuté ses interactions avec celle-ci.

Terminons en mentionnant une commande très utile, mais accessible uniquement à l'aide de la touches « p » du clavier : prendre une photo de la figure affichée, et placer le fichier correspondant dans le dossier contenant la description de la figure.

Un exemple de figure plus complexe

Pour illustrer d'autres facettes de la définition de figures, nous allons décrire une sorte de « toît refermable », avec des glissières pouvant spécifier le nombre de triangles utilisés, et le degré de fermeture du toît (voir le document 6).

Document 6. Diverses positions de notre toît

Le document 7 nous révèle la description de la figure que nous avons utilisée. Jetons-y un coup d'œil.

À l'étape 1, dans la procédure **definirFigure**, nous commençons par définir trois glissières : une pour spécifier le nombre de triangles formant notre toît (entre 3 et 20), une autre pour fixer la hauteur des sommets, et une dernière pour choisir le rayon du cercle sur lequel se trouveront les sommets du haut. Nous décrivons ensuite la position des sommets à la base du toît : régulièrement espacés sur un cercle dans le plan X-Z. De même, nous décrivons la position des sommets qui serviront à « refermer le toît » : régulièrement espacés sur un cercle dans un plan parallèle au plan plan X-Z, et décalés par rapport aux sommets de la base. Puis nous utilisons ces sommets pour définir les faces. Notons au passage que nous utilisons les numéros de ces faces pour « calculer » leur couleur.

Soulignons ici que nous devons créer à cette étape tous les sommets et toutes les faces (soit 20, le maximum). Plus tard, nous ne pourrons que redéfinir les coordonnées de certains sommets, et décider quelles faces seront inclues ou exclues de la figure.

À l'étape 2, dans la procédure **initialiserFigure**, nous nous contentons d'incliner la figure vers l'utilisateur.

À l'étape 3, dans la procédure **redefinirCertainsSommets** (qui est appelée périodiquement par SPAF), on lit la valeur des trois glissières et on modifie la figure en conséquence : changement de la position des sommets mobiles, inclusion des faces retenues, et exclusion des autres.

```
void definirFigure(){
  pixelsParUnite(100);

  // ajusterGlissiere si nécessaire
  creerGlissiereG1("nb de  triangles",3, 20, 20);  // nombre de côtés du polygone
  creerGlissiereG2("hauteur",-2, 2, -2);    // hauteur du sommet
  creerGlissiereG3("rayon",0, 2, 2);    // rayon supérieur

  // Définition des sommets
  for(int k=0;k<=20;k++) {
    definirSommet(k+1,2*cos(k*PI/10),0,2*sin(k*PI/10));
    definirSommet(k+22,2*cos(k*PI/10+PI/20),-2,2*sin(k*PI/10+PI/20));
  }

  // Définition des faces
  int coul;
  for(int k=1;k<=20;k++) {
    definirFace(k, k,k+1,k+21);
    if (k % 2 == 0) {coul=round(k*255.0/20);} else {coul=255-round(k*255.0/20);}
    couleurFace(k,255, coul, 255-coul);
  }
}

void initialiserFigure(){
   fixeRotationX(-40);
}

void redefinirCertainsSommets(){
   int nb = round(valG1);
   float h = valG2;
   float r = valG3;

   for(int k=0;k<=nb;k++) {
     redefinirSommet(k+1,2*cos(k*2*PI/nb),0,2*sin(k*2*PI/nb));
     redefinirSommet(k+22,r*cos(k*2*PI/nb+PI/nb),h,r*sin(k*2*PI/nb+PI/nb));
   }
   for(int k=1;k<=nb;k++) {
     integrerDansFigure(k,true);
   }
   for(int k=nb+1;k<=20;k++) {
     integrerDansFigure(k,false);
   }
}
```

Document 7. Description de notre figure

En terminant…

Nous sommes arrivés à la fin de cette brève documentation. Cependant, vous pouvez continuer à vous familiariser avec SPAF en consultant les exemples fournis avec le logiciel. Ceux-ci se retrouvent dans le dossier compressé SPAF.zip, disponible sur le site web de SPAF, qui contient deux dossiers : le dossier *SPAF* lui-même, qui correspond en fait à une figure vierge, et le dossier *Exemples*. Ce dernier contient un ensemble de dossiers, qui correspondent chacun à une figure particulière, à deux exceptions près :

- Le dossier *Graphes* contient lui-même plusieurs exemples/dossiers de surfaces paramétriques, ainsi qu'une documentation sur le sujet.
- Le dossier *Animations* contient, lui aussi, plusieurs exemples/dossiers de figures, accompagnées chacune d'un petit film fabriqué à partir de SPAF. Le processus est expliqué dans un tutoriel vidéo sur le site web de SPAF.