Des roulades de polygones

Christophe Poulain

Table des matières

Si l'on conçoit aisément qu'un cercle « roule », il est nettement plus difficile d'imaginer qu'autre chose puisse rouler. Et pourtant, ne peut-on chercher la manière dont roule un polygone, de préférence régulier ?
Il faut bien reconnaître qu'il y a bien longtemps que cette recherche a commencé : nos ancètres n'ont-ils pas construits, dans un premier temps, des *roues carrées* pour faire avancer leurs charettes ? Ce n'est donc pas un problème nouveau.

L'auteur aborde ce problème d'un point de vue mathématique mais également informatique. Si la pensée permet d'imaginer et de comprendre les différentes trajectoires, encore faut-il les représenter ensuite ! Et dans ce cas, l'ordinateur est d'un grand secours, surtout si l'on dispose d'un outil comme MetaPost.
Ce logiciel convient très bien pour *programmer*. A la base, MetaPost est un logiciel, interfacé avec LaTeX, qui permet de produire des images d'une qualité supérieure. Mais, pour obtenir ces figures, il faut programmer les différents actions pour que MetaPost puisse les réaliser. Il dispose pour cela de toutes les fonctions de tests, de boucles,. . .

On trouvera au fil des pages les différentes étapes de la réflexion pour aboutir au résultat final. On remarquera aussi l'amélioration du code de programmation au fur et à mesure des questions. Enfin, on s'émerveillera devant la qualité des figures produites par MetaPost.

Ce petit fasicule n'a pour seule prétention que de faire comprendre le plaisir qu'a éprouvé l'auteur à réflechir et à mettre au point toutes ces macros MetaPost afin qu'elles soient, bien entendu, réutilisées, voire améliorées, par d'autres.

Chapitre 1

Vers la cycloïde

1.1 Présentation

Tout ce travail a débuté par une question toute simple : « suis-je capable, avec MetaPost, de reproduire cette figure intéressante du livre[1] ? ». La figure en question était celle-ci.

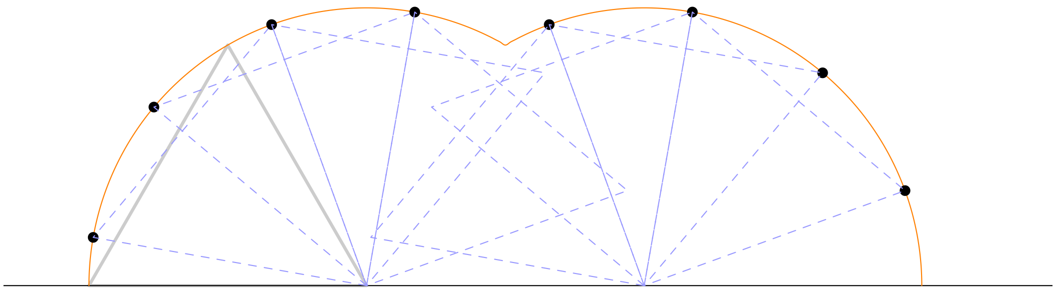


FIG. 1.1 – Un triangle équilatéral roule sur une droite.

Elle permet à l'élève de 6^e d'observer la roulade d'un triangle équilatéral sur une droite. Elle est donnée comme ci-dessus avec les triangles équilatéraux *intermédiaires* tracés afin que l'élève puisse avoir une représentation visuelle puis mentale du processus de roulade. Suit alors une figure sur laquelle on observe un carré placé sur une droite ; figure que l'élève doit compléter afin d'obtenir la roulade du carré sur la droite. La réponse est donnée par la figure 1.2.

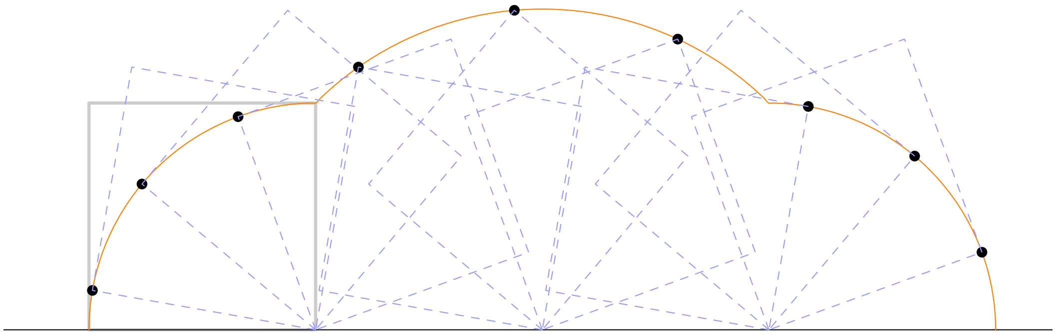


FIG. 1.2 – Un carré roule sur une droite.

Satisfait par l'obtention de la figure, pouvant l'inclure dans n'importe quel de mes documents, seule la curiosité me pousse à me demander ce qui se passe si on augmente le nombre de côtés du polygone régulier ;

[1] Géométrie 6^e – Travaux dirigés – *Bernard Tissot* – Hatier

quelle trajectoire obtient-on, comment la construire et la tracer ?
Il a donc fallu que je généralise mon programme MetaPost. En effet, pour obtenir les 2 figures ci-dessus, j'ai crée deux programmes différents, ne pensant pas devoir aller plus loin. Malgré tout, cette différentiation a eu le mérite de m'obliger à décortiquer tout le mouvement et à trouver les *invariants* dans les deux cas : on utilise des rotations d'angle $\frac{2\pi}{n}$ où n est le nombre de côtés du polygone régulier considéré.

Ayant ces invariants en main, la généralisation[2] des deux programmes fut « facilitée », le plus intéressant, mais également le plus difficile, étant de construire une macro-commande qui me permette d'un simple appel (ici *roulades(n,t)*[3]) de construire la roulade souhaitée : la réflexion étant faite, il s'agissait de *faire faire* le travail par MetaPost. Voici quelques exemples obtenus avec cette macro-commande.

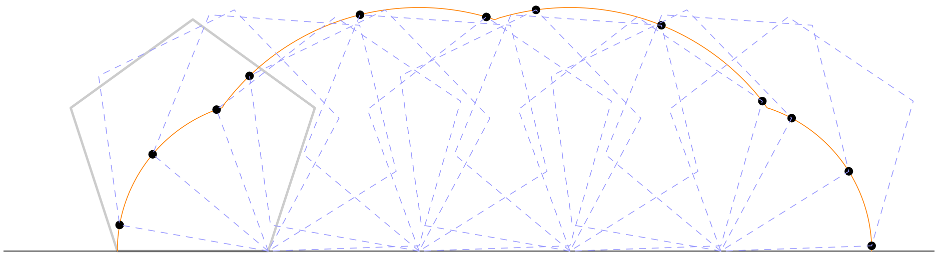


Fig. 1.3 – Un pentagone régulier roule sur une droite.

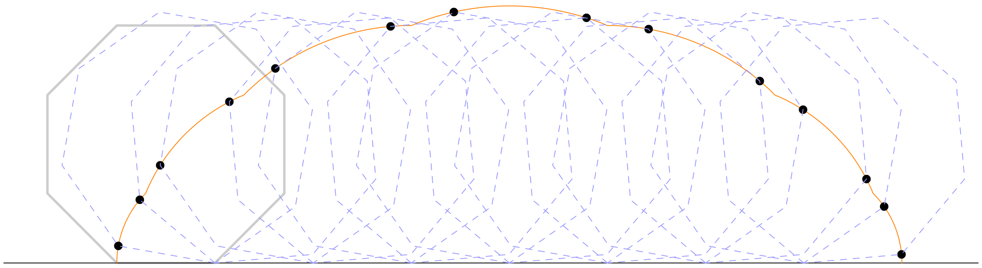


Fig. 1.4 – Un octogone régulier roule sur une droite.

Il faut remarquer dans la construction du programme qu'une autre trajectoire est calculée : il s'agit de celle du centre du polygone régulier considéré. Elle a été rendue nécessaire pour la construction du polygone régulier à n'importe quelle étape de la roulade.
Une autre remarque essentielle : la courbe est construite point par point, ce qui permet, en reprenant une partie du programme, d'en faire « une animation[4] » : un assemblage de plusieurs images fixes qui permettent d'avoir l'illusion du mouvement. C'est aussi une des forces de MetaPost !

En poursuivant le raisonnement, par curiosité, on peut se dire que *plus le nombre de côtés augmente, plus le polygone régulier se rapproche du cercle.* A priori, la trajectoire décrite par ces polygones réguliers tendant vers le cercle devrait se rapprocher d'une *cycloïde*. L'intérêt de la macro est ici très important : juste une et une seule valeur numérique à modifier pour que MetaPost calcule la nouvelle trajectoire. Regardons alors de plus près ce que cela donne ; la figure suivante est construite à partie d'un icosagone régulier

et voici une cycloïde.

[2]On trouvera le code source de ce programme à l'annexe A

[3]n est le nombre de côtés du polygone régulier considéré et t un paramètre permettant d'afficher ou non certaines étapes de la roulade.

[4]A ce sujet, on pourra en voir aux adresses www.melusine.eu.org/syracuse/poulecl/albums/roulades6/ ou www.melusine.eu.org/syracuse/poulecl/albums/roulades12/

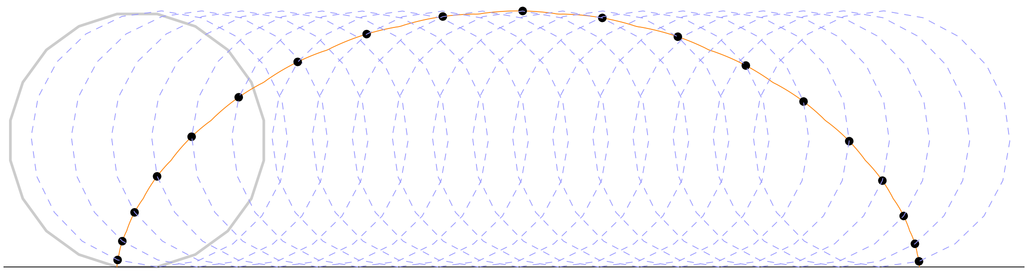


FIG. 1.5 – Un icosagone régulier roule sur une droite.

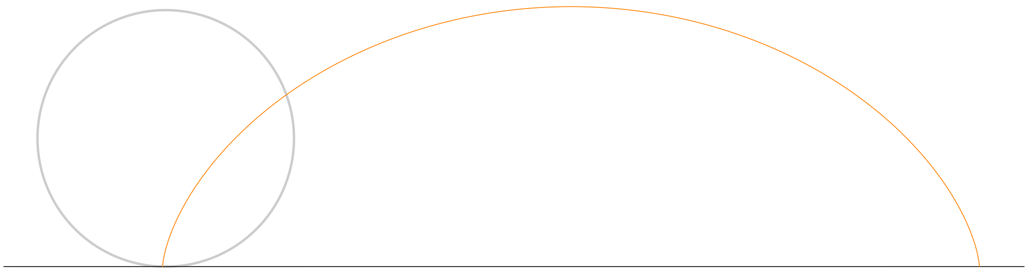


FIG. 1.6 – Une cycloïde.

Etonnant, non ?

1.2 Un peu de mathématiques

Longueur d'une arche de trajectoire. Il est intéressant de calculer la longueur de chacune des arches des trajectoires. Etudions quelques cas particuliers en considérant que les différents polygones réguliers ont des côtés de longueur p et sont inscrits dans un cercle de rayon r.

Triangle équilatéral $2 \times p \times \dfrac{2\pi}{3} = \dfrac{4\pi p}{3}$

Carré $2 \times p \times \dfrac{2\pi}{4} + p\sqrt{2} \times \dfrac{2\pi}{4} = p(2+\sqrt{2})\dfrac{2\pi}{4}$

Pentagone régulier $2 \times p \times \dfrac{2\pi}{5} + 2 \times \sqrt{p^2 - \dfrac{r^2}{4}} \times \dfrac{2\pi}{5}$

Dans le cas d'un polygone régulier de côtés n, la trajectoire est composée de $n-1$ arcs de cercles d'angles $\dfrac{2\pi}{n}$, leur rayon étant les longueurs des segments A_1A_k où $2 \leqslant k \leqslant n$ (voir figure 1.7).

Si, L_n est la longueur de la trajectoire considérée, alors on a

$$L_n = \frac{2\pi}{n} \sum_{2 \leqslant k \leqslant n} A_1A_k$$

$$L_n = \frac{2\pi}{n} \sum_{2 \leqslant k \leqslant n} 2r\sin(k-1)\frac{\pi}{n}$$

$$L_n = \frac{4r\pi}{n} \sum_{2 \leqslant k \leqslant n} \sin(k-1)\frac{\pi}{n}$$

$$L_n = \frac{4r\pi}{n} \sum_{1 \leqslant k \leqslant n-1} \sin k\frac{\pi}{n}$$

$$L_n = \frac{4r\pi}{n} \frac{\sin\frac{\pi}{n}}{1-\cos\frac{\pi}{n}}$$

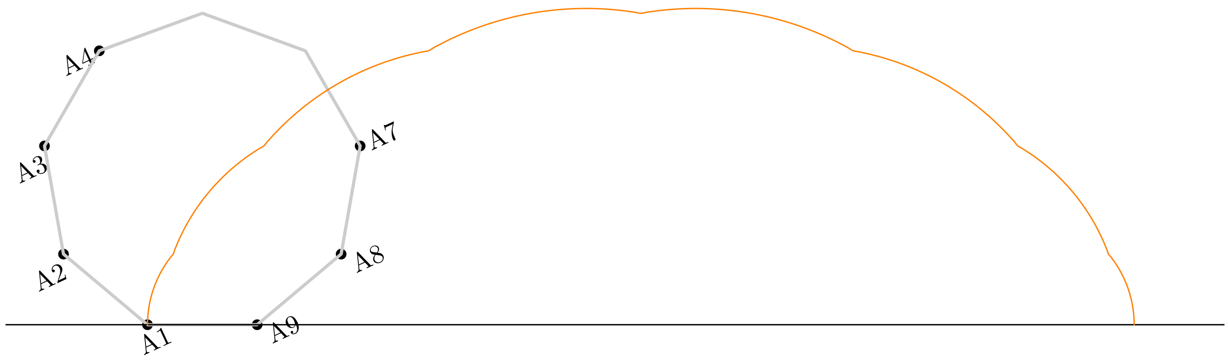


FIG. 1.7 – Position des différents sommets : cas du nonagone ($n = 9$)

FIG. 1.8 – Paramétrage d'une cycloïde

Lorsque n tend vers l'infini, le polygone régulier tend vers le cercle. La longueur L_n a donc pour limite la longueur d'une arche de cycloïde.
La limite de L_n est $8r$, longueur d'une arche de cyloïde.

Une autre méthode pour calculer la longueur d'une arche de cycloïde consiste à utiliser l'abscisse curviligne. Une paramétrisation possible (voir figure 1.8) de la cycloïde est, en posant $\theta = (\overrightarrow{OM}, \overrightarrow{OI})$ avec I le point de tangence de la cycloïde et de la droite sur laquelle elle roule :

$$x = r(\theta - \sin\theta) \qquad y = r(1 - \cos\theta)$$

La longueur d'une arche Γ de cycloïde est donc

$$\begin{aligned}
L &= \int_\Gamma ds \\
L &= \int_0^{2\pi} r\sqrt{(1-\cos\theta)^2 + \sin^2\theta d\theta} \\
L &= \int_0^{2\pi} 2r\sin\frac{\theta}{2}d\theta \\
L &= 2r\int_0^{2\pi} \sin\frac{\theta}{2}d\theta \\
L &= 8r
\end{aligned}$$

Aire sous une arche. Il s'agit de calculer l'aire de la surface hachurée ci-dessous.

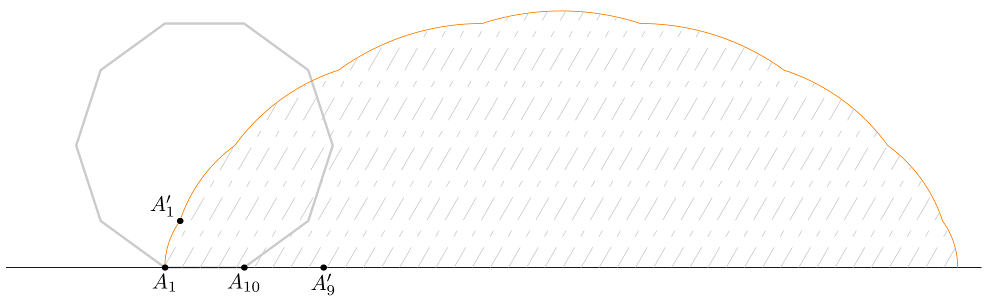


FIG. 1.9 – Surface déterminée par une arche d'un roulade. Cas d'un décagone ($n = 10$).

L'aire S_n peut se décomposer de la façon suivante :

– des triangles $A'_1A_{10}A'_9$ dont la somme des aires est égale à celle du polygone régulier considéré à savoir

$$n\frac{r^2}{2}\sin\frac{2\pi}{n}$$

– des secteurs angulaires $A_1A_{10}A'_1$ qui ont tous le même angle $\dfrac{2\pi}{n}$; la somme de leurs aires étant

$$\frac{\pi}{n}\sum_{k=2}^{n}A_1A_k^2$$

ou[5]

$$2\pi r^2$$

On obtient alors

$$S_n = 2\pi r^2 + n\frac{r^2}{2}\sin\frac{2\pi}{n}$$

qui a pour limite, lorsque n tend vers $+\infty$,

$$S = 3\pi r^2$$

Comme pour la longueur d'une arche, on peut utiliser l'abscisse curviligne pour obtenir l'aire sous une arche.

$$\begin{aligned}
\mathcal{A} &= \int_0^{2\pi r} y\,dx\\
\mathcal{A} &= r^2\int_0^{2\pi}(1-\cos\theta)^2\,d\theta\\
\mathcal{A} &= r^2\int_0^{2\pi}\left(\frac{3}{2}-2\cos\theta+\frac{\cos 2\theta}{2}\right)d\theta\\
\mathcal{A} &= 3\pi r^2
\end{aligned}$$

[5]En appelant O le centre du polygone régulier considéré et en utilisant la formule de Leibniz.

Chapitre 2

Un polygone régulier sur un même autre polygone régulier : Cas du triangle équilatéral

Le cas « rouler sur une droite » étant traité, envisageons maintenant le cas où le polygone régulier, en l'occurence un triangle équilatéral ici, roule sur un autre triangle équilatéral. Quelle type de trajectoire cela peut-il donner ?

2.1 Rapport de longueur $1/n$ où $n > 0$

Pour débuter, j'ai étudié le cas où le rapport de longueur entre les côtés des deux triangles équilatéraux était de la forme $\frac{1}{n}$ où $n > 0$. Que se passe-il alors ?
Il me fallait d'abord définir mes différents centres de rotations : avec une boucle de programmation, MetaPost gère le travail et me donne le `cen[k]` où `k` est l'indice du centre considéré ; d'ailleurs `k` varie de 0 à $3 \times n - 1$.

2.1.1 Rouler à l'extérieur

Tant que le triangle équilatéral « roulant » (noté $\mathcal{E}_r$) n'arrive pas à un des sommets du triangle équilatéral « fixe » (noté $\mathcal{E}_f$), il n'y a pas de problème : cela se passe comme sur la droite.

Là où le problème devient un peu plus délicat, c'est lorsque la roulade arrive à un des sommets de $\mathcal{E}_f$; $\mathcal{E}_r$ ne tourne plus simplement d'un angle de $2\pi/3$ mais de $4\pi/3$.
Il fallait donc programmer un test permettant à MetaPost de définir si la rotation se faisait par rapport à un des sommets de $\mathcal{E}_f$ ou non. La division euclidienne fut d'un grand secours : si le reste entre le coefficient `k` du centre de rotation considéré et n est nul, alors je suis à un sommet et je tourne de $4\pi/3$ sinon je tourne de $2\pi/3$.

Il ne reste plus qu'à poursuivre la construction jusqu'à revenir au point de départ.

D'un point de vue mathématique. Soit a la longueur d'un côté de $\mathcal{E}_r$ et α la longueur d'un des côtés de $\mathcal{E}_f$. Considérons que le rapport de longueur des côtés des deux triangles équilatéraux soit $1/n$, avec $n > 0$. Remarquons alors que $a \times n = \alpha$.

La trajectoire décrite est composée de :

- $3 \times 2 \times \dfrac{n}{3} = 2n$ arcs de cercles de rayon a et d'angles $2\pi/3$ si $n = 3s$ où s est un nombre entier non nul ;
- $3 \times 2 \times \dfrac{n-1}{3} = 2(n-1)$ arcs de cercles de rayon a et d'angles $2\pi/3$ et 2 arcs de cercles de rayon a et d'angle $4\pi/3$ si $n = 3s + 1$ où s est un nombre entier ;
- $3 \times 2 \times \dfrac{n-2}{3} + 2 = 2(n-1)$ arcs de cercles de rayon a et d'angles $2\pi/3$ et 2 arcs de cercles de rayon a et d'angle $4\pi/3$ si $n = 3s + 2$ où s est un nombre entier.

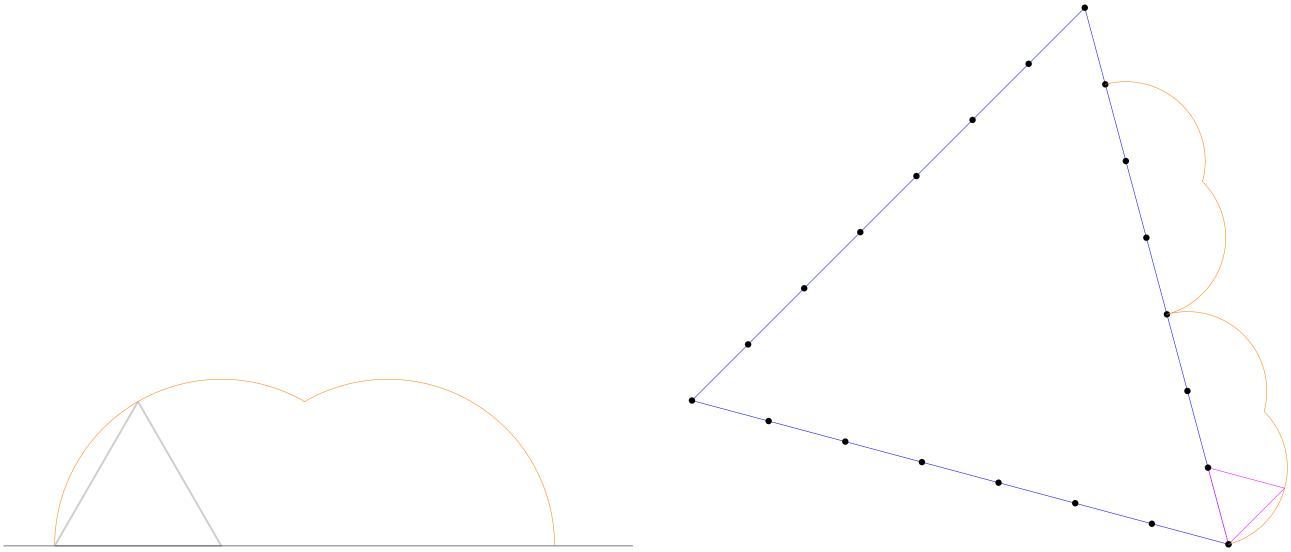


Fig. 2.1 – Triangle équilatéral roulant sur une droite et sur un autre triangle équilatéral.

Posons L_n la longueur de la trajectoire. On a donc

$$L_n = \begin{cases} \dfrac{4an\pi}{3} = \dfrac{4\alpha\pi}{3} \text{ si } n = 3s,\ s > 0. \\[2ex] \dfrac{4a\pi(n-1)}{3} + \dfrac{8a\pi}{3} = \dfrac{4\alpha\pi}{3} + \dfrac{4a\pi}{3} \text{ si } n = 3s+1,\ s \geqslant 0. \\[2ex] \dfrac{4a\pi(n-2)}{3} + \dfrac{4a\pi}{3} + \dfrac{8a\pi}{3} = \dfrac{4\alpha\pi}{3} + \dfrac{4a\pi}{3} \text{ si } n = 3s+2,\ s \geqslant 0. \end{cases}$$

Il est à remarquer que, quelque soit le multiple non nul de 3 choisi, la longueur de la trajectoire est constante.

Posons maintenant S_n l'aire de la surface délimitée par la trajectoire autour du triangle équilatéral $\mathcal{E}_f$. C'est la surface hachurée ci-dessous. On obtient pour expressions de S_n :

$$S_n = \begin{cases} 2na^2\dfrac{\pi}{3} = \dfrac{2\pi a\alpha}{3} \text{ si } n = 3s,\ s > 0. \\[2ex] 2(n-1)a^2\dfrac{\pi}{3} + 2a^2\dfrac{2\pi}{3} = \dfrac{2\pi a\alpha}{3} + \dfrac{2a^2\pi}{3} \text{ si } n = 3s+1,\ s \geqslant 0. \\[2ex] \dfrac{2\pi a\alpha}{3} + \dfrac{2a^2\pi}{3} \text{ si } n = 3s+2,\ s \geqslant 0. \end{cases}$$

2.2 Rapport de longueur rationnel

Considérons maintenant le cas où le rapport de longueur entre les deux côtés des triangles équilatéraux soit de la forme $\dfrac{num}{deno}$ avec $deno \neq 0$. Comment construire la courbe ?
La section précédente me permis d'appréhender les différents problèmes qui allaient se présenter : le passage des sommets, le nombre de tour pour revenir au point initial,...
Etudions les dans l'ordre.

Le passage des sommets reste, à mon avis le point le plus délicat : comment le triangle roulant $\mathcal{E}_r$ arrive en un sommet ? Eh bien, la plupart du temps, il se trouve même que les centres de rotation soient parfaitement différents des 3 sommets ! Si tel était le cas, le triangle équilatéral $\mathcal{E}_r$ tourne autour d'un des centres `cen[k]` *puis* autour du sommet pour venir se positionner sur l'autre côté du triangle équilatéral $\mathcal{E}_f$. Il fallait donc programmer un moyen de distinguer les centres qui obligent $\mathcal{E}_r$ autour de ces centres puis autour du sommet le

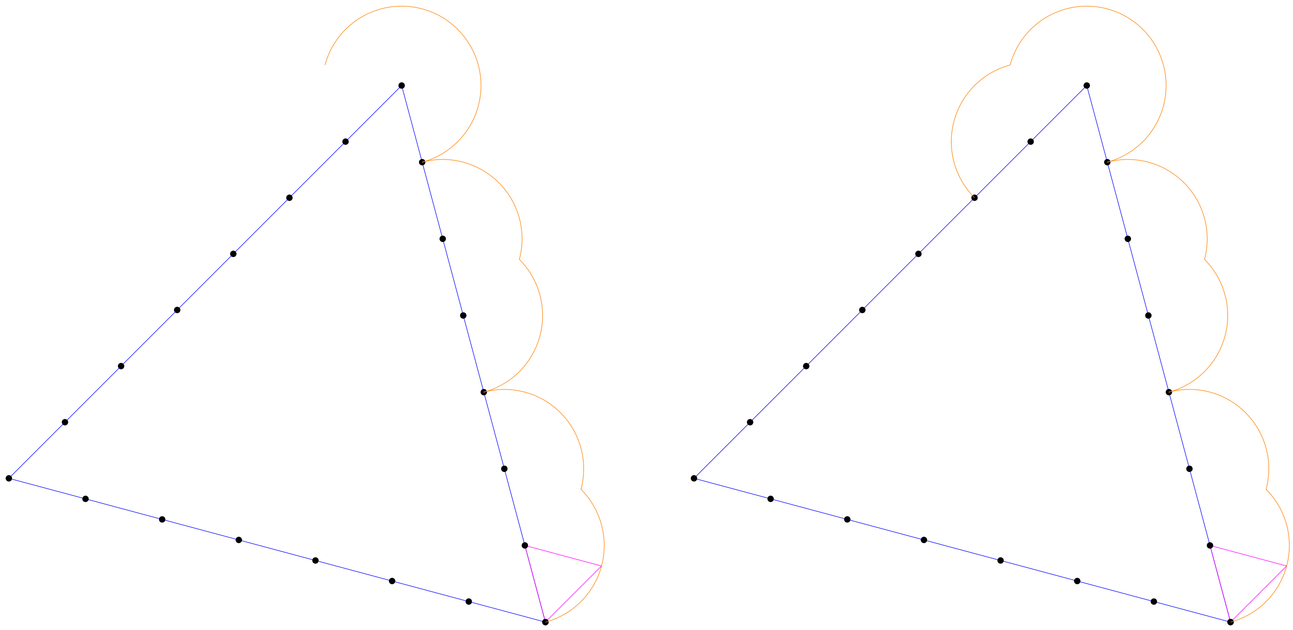


FIG. 2.2 – $\mathcal{E}_r$ est ou n'est pas en un sommet de $\mathcal{E}_f$.

plus proche.
Le test aurait pu se faire en terme de longueur : si le centre est à une distance inférieure à a (qui est la longueur du côté de $\mathcal{E}_f$) alors je fais deux rotations : une de centre `cen[k]` suivie d'une rotation de centre le sommet considéré, toutes deux d'angle $\dfrac{2\pi}{r}$. Malheureusement, MetaPost est meilleur sur les entiers que sur le calcul de longueur (qu'il fait très bien cependant).
Je me suis donc rabattu sur un test sur les entiers : si la différence entre la numérotation du sommet (qui est automatiquement compté dans les `cen[k]`) et le paramètre `k` du centre est inférieur au numérateur *num* du rapport de longueur alors je fais deux rotations : une de centre `cen[k]` suivie d'une rotation de centre le sommet considéré, toutes deux d'angle $\dfrac{2\pi}{r}$.

Il ne reste plus qu'à calculer le nombre de tours ou le nombre de rotations nécessaires pour *boucler*.
En choisissant de compter le nombre de rotations, il y a $3 \times$ *deno* centres de rotations sur le périmètre du triangle $\mathcal{E}_f$ et à chaque rotation, il y a *num* centres qui sont utilisés. Il faut donc prendre :

- le Plus Petit Commun Multiple de $3 \times$ *deno* et de *num* si ces deux nombres sont premiers entre eux. Par exemple, pour le rapport 2/7, on doit prendre le ppcm de 21 et 2, à savoir 42 rotations.
- le produit de $3 \times$ *deno* et de *num* si ces deux nombres ne sont pas premiers entre eux.

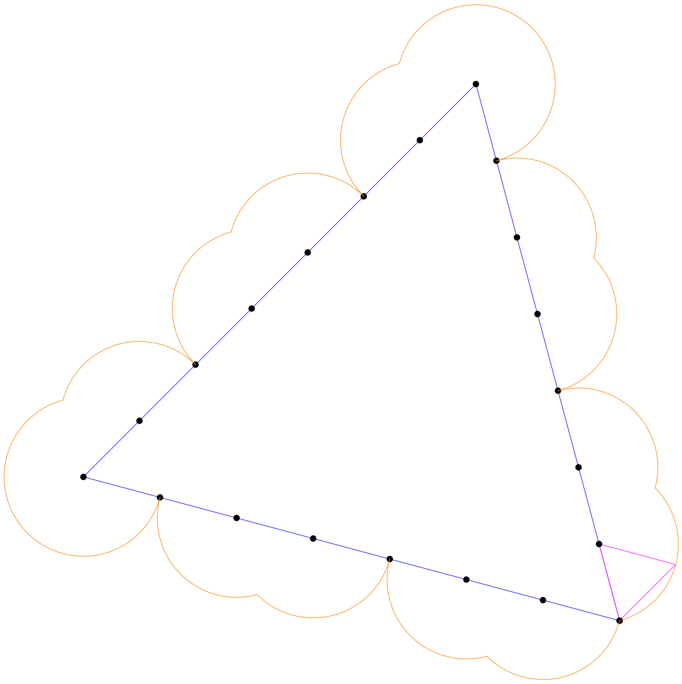


Fig. 2.3 – Trajectoire décrite par un triangle équilatéral roulant sur un triangle équilatéral.

Quelques exemples supplémentaires

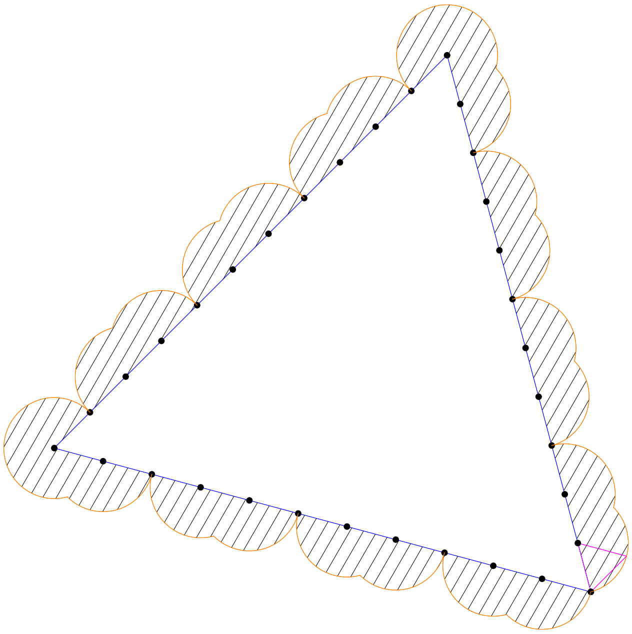
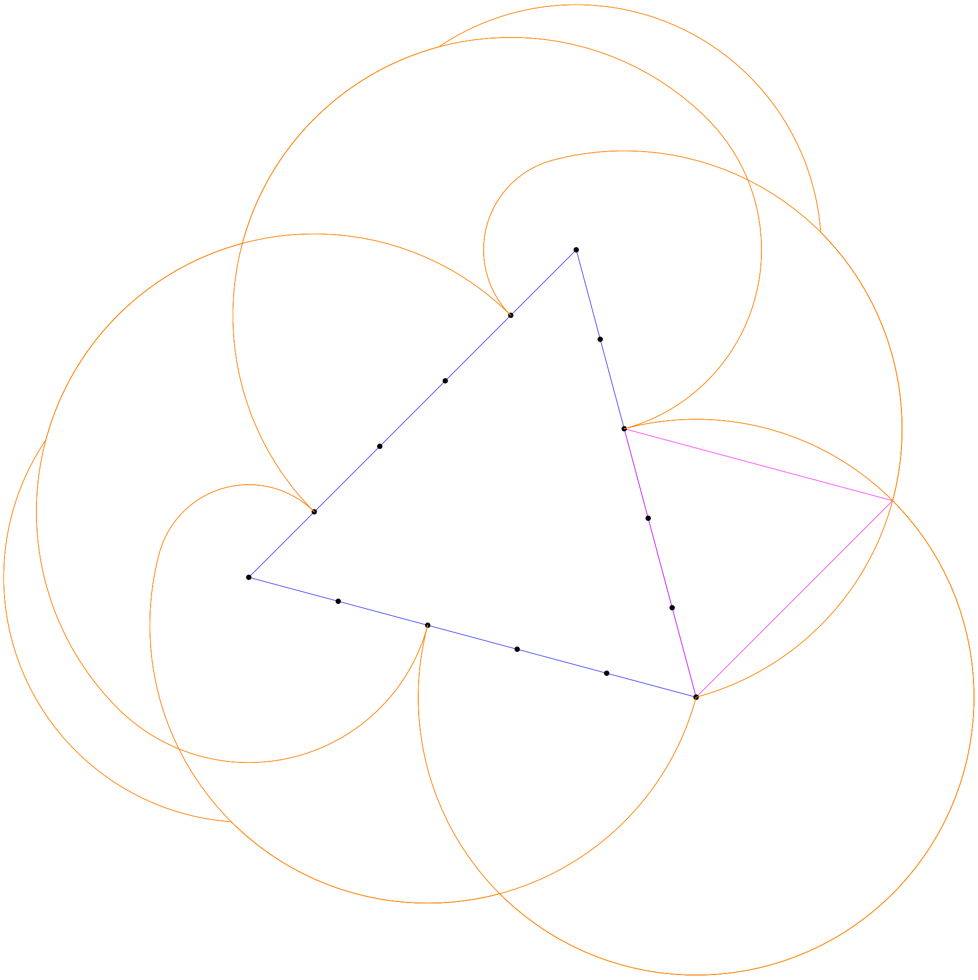


FIG. 2.4 – Surface délimitée par la trajectoire autour de $\mathcal{E}_f$ (cas $n = 11$).

Cas 3/5 : dans ce cas, il y a 45 rotations à effectuer.



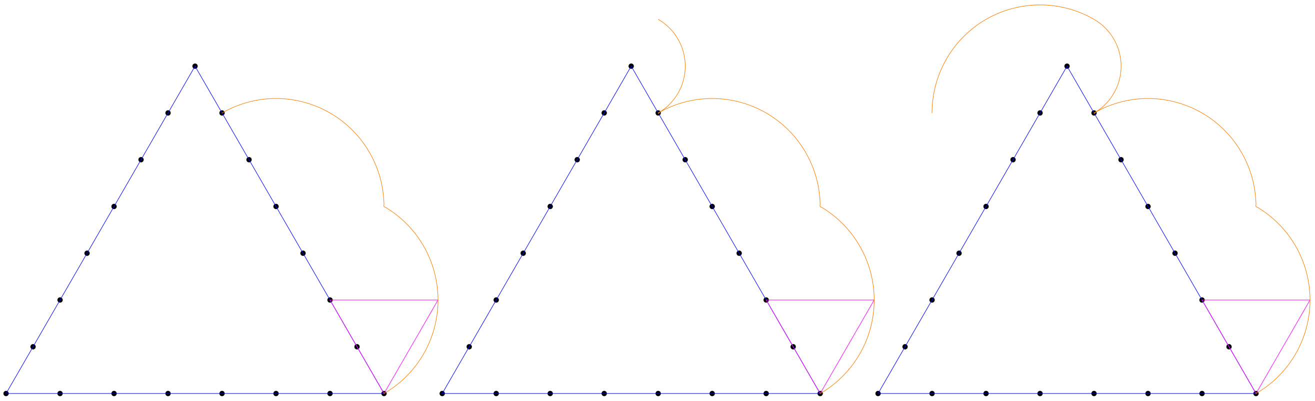


FIG. 2.5 – $\mathcal{E}_r$ est ou n'est pas en un sommet de $\mathcal{E}_f$ (rapport 2/7).

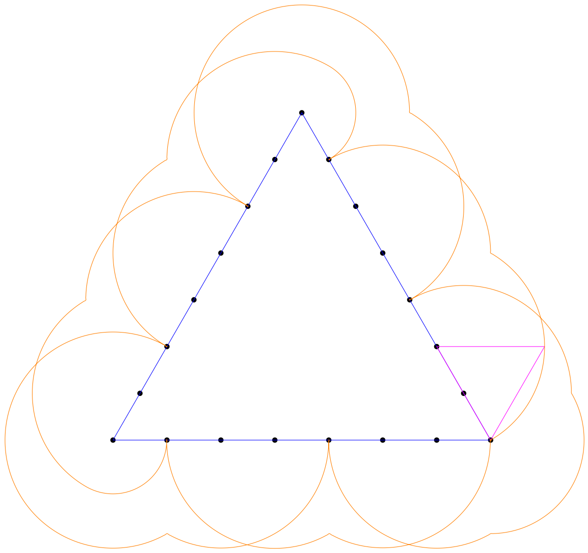
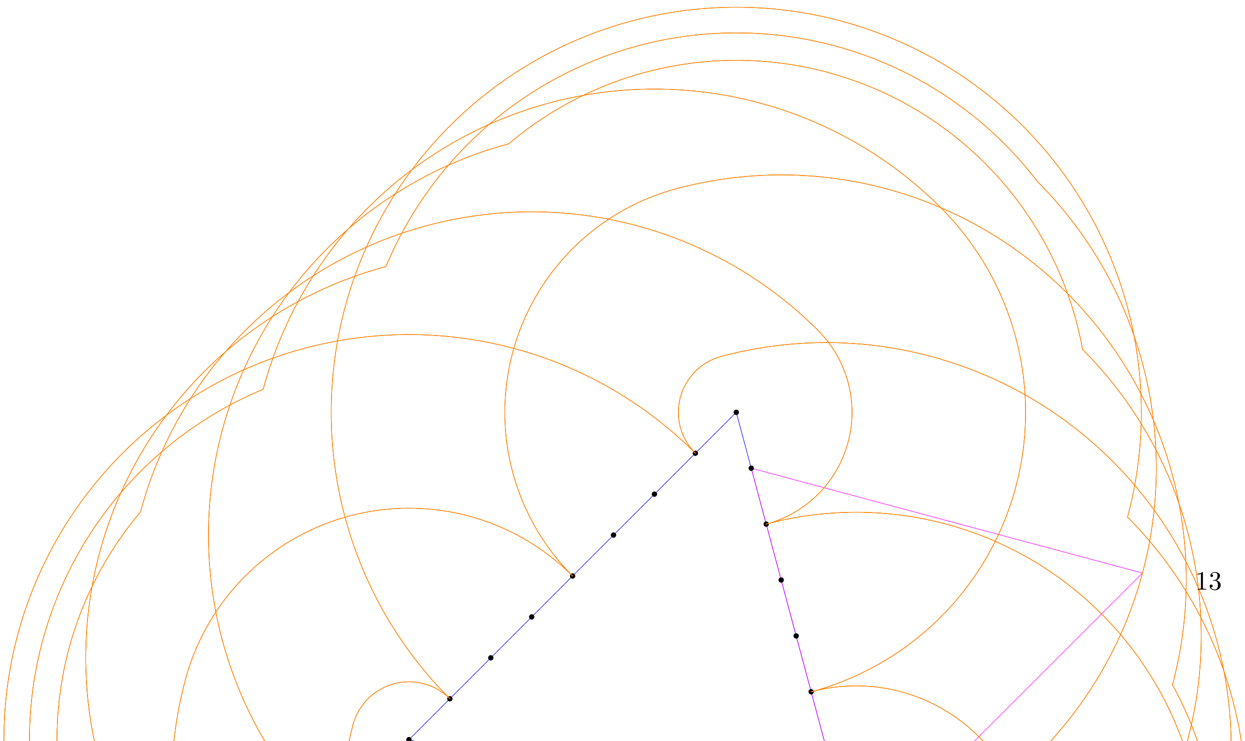


FIG. 2.6 – Trajectoire totale dans le cas d'un rapport égal à 2/7.

Cas 7/8 : dans ce cas, il y a 168 rotations à effectuer



Chapitre 3

Où on mélange les polygones réguliers

Doit-on se limiter à prendre deux triangles équilatéraux ? Non, bien sur ! Pourquoi ne pas envisager un carré roulant sur un pentagone régulier ? C'est la question que m'a posée un ami qui avait pu observer mes précédentes *expériences informatiques*.
Le cahier des charges était donc le suivant : *créer un macro-commande en MetaPost qui permette à l'utilisateur de construire la trajectoire décrite par la roulade d'un polygone régulier à côtés m sur un polygone régulier à n côtés et dont le rapport des longueurs des côtés est choisi par l'utilisateur.* Cela fait beaucoup de contraintes !

Construire des polygones réguliers avec MetaPost n'est pas un problème en soit : pour un polygone régulier à n côtés, on fixe le premier sommet A_1 et un point O qui est le centre du polygone ; ensuite, on effectue une boucle de $n-1$ pas dans laquelle on construit les images de A_2 par des rotations de centre O et d'angles $\frac{2k\pi}{n}$ avec $2 \leqslant k \leqslant n$.
Une fois ceci fait, il faut construire les différents centres de rotations : là encore, 2 boucles bien définies permettent à MetaPost de déterminer ces points.

Arrive le problème de la trajectoire et se pose encore le problème des sommets. Mais cette fois, les contraintes sont différentes : les angles de rotation ne sont pas identiques.
J'utilise néanmoins la même méthode que pour le triangle équilatéral mais dans une version sensiblement *différente* : il faut cette fois-ci que le polygone roulant $\mathcal{E}_r$

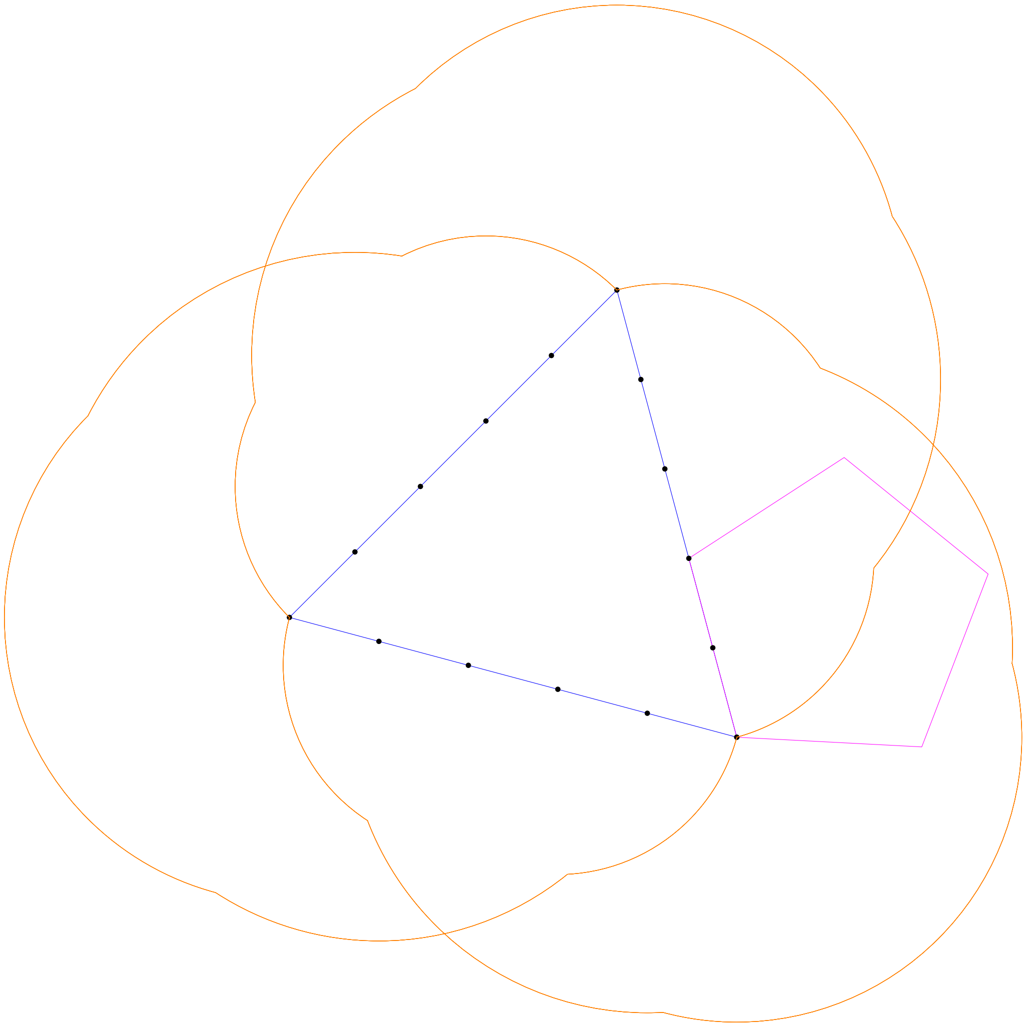
- soit à une distance inférieure à a d'un des sommets de $\mathcal{E}_f$, tout en restant *avant* ce sommet. Dans ce cas,
  - si cette distance est nulle, alors la rotation se fait par rapport au sommet d'un angle égal à $3.5\frac{2\pi}{n}$;
  - sinon, on effectue deux rotations : une autour du sommet d'angle $\frac{2\pi}{n}$ suivie d'une rotation autour du `cen[k]` considéré d'angle $\frac{2\pi}{m}$
- ou soit en un centre `cen[k]` où `k` est un multiple du dénominateur : dans ce cas, on est tombé *directement* sur un des sommets de $\mathcal{E}_f$, et on effectue une rotation autour de ce sommet d'angle $\frac{2\pi}{m} + \frac{2\pi}{n}$.
- sinon on est sur un des côtés de $\mathcal{E}_f$ et on effectue une rotation d'angle $\frac{2\pi}{m}$.

Malgré ceci fait, une nouvelle modification était devenue obligatoire. Jusqu'à présent, la construction des différentes trajectoires se faisait point par point. Or, MetaPost est configuré pour n'avoir qu'un nombre limité[1] de points sur une courbe. Il fallait contourner ce problème : il fallait faire appel à une macro `arccercle`[2] et cela me faisait, non seulement disparaître ce problème de limite mais réduisait considérablement le temps de compilation des fichiers.

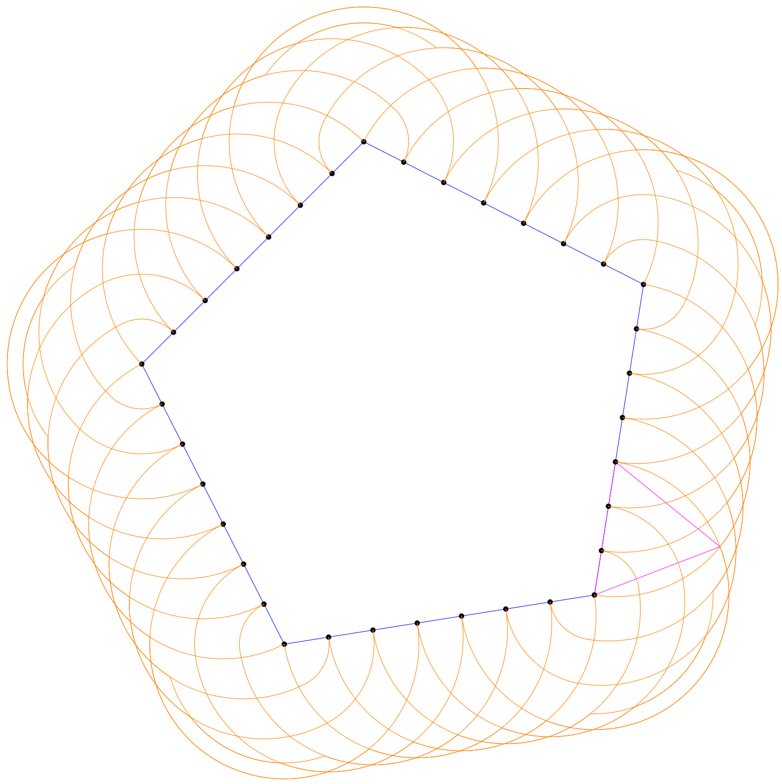
Quelques exemples On touvera la version finale de ce programme à l'Annexe B.

[1] En fait, on peut modifier cette valeur mais cela reste néanmoins une limite
[2] Que j'avais déjà construite

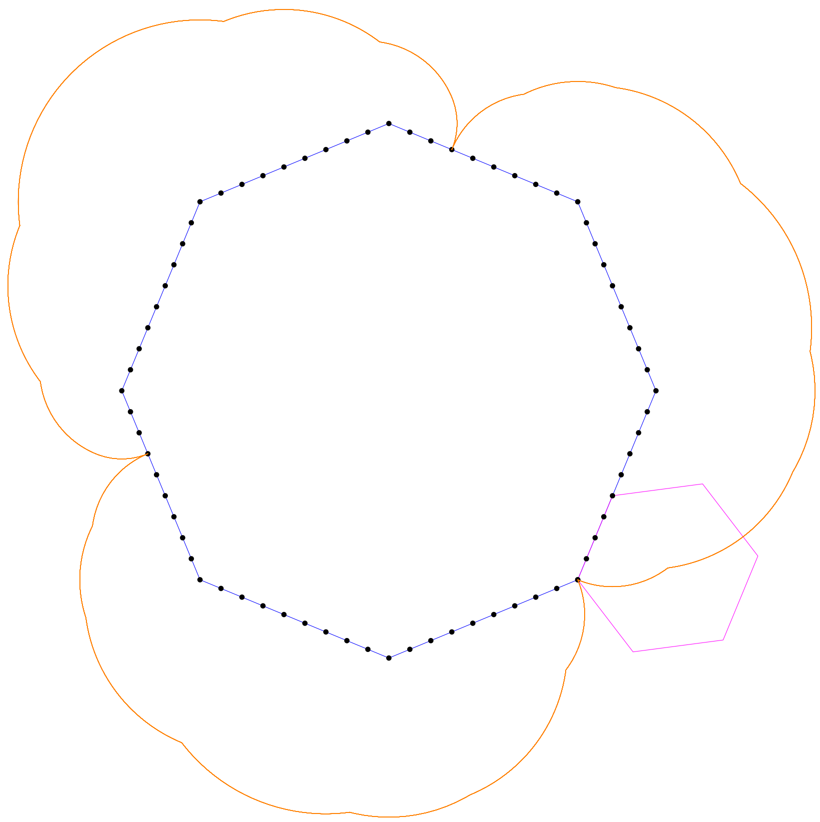
Pentagone régulier roulant sur un triangle équilatéral dans le rapport 2/5



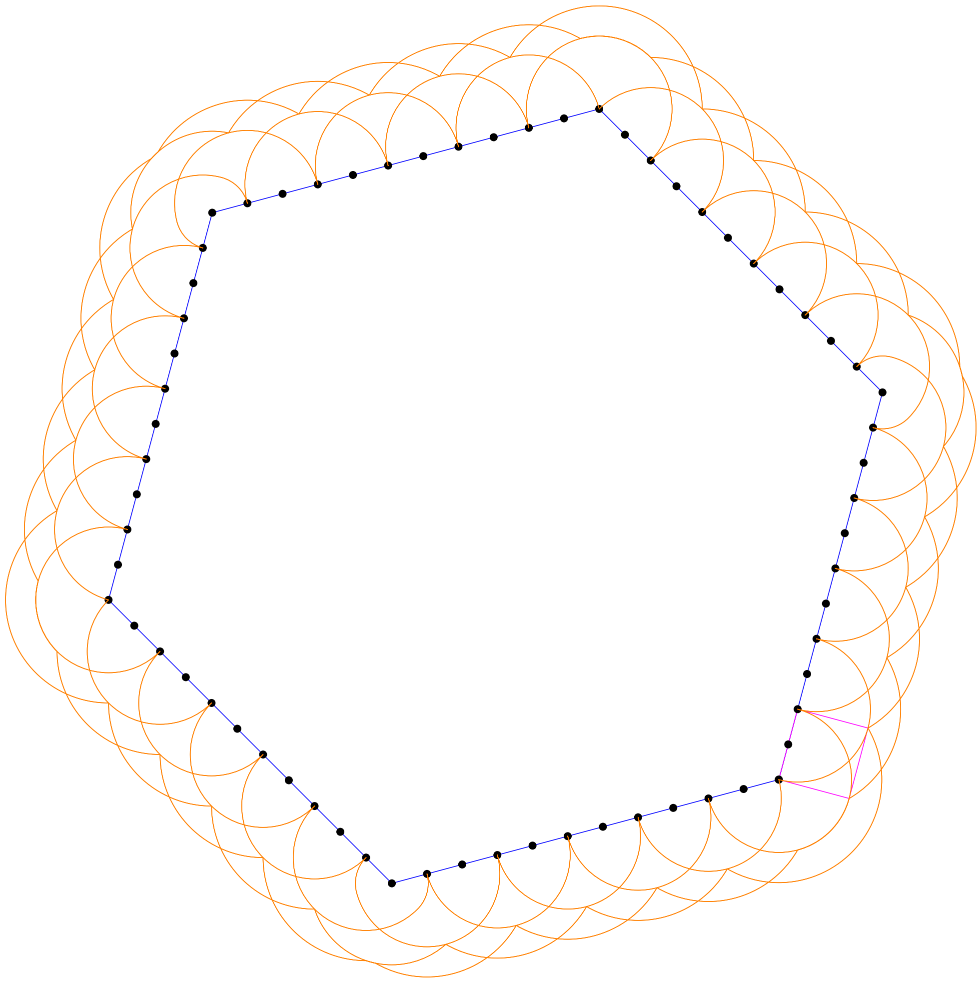
Triangle équilatéral roulant sur un pentagone régulier dans le rapport 3/7



Hexagone régulier roulant sur un octogone régulier dans le rapport 4/9



Carré roulant sur un hexagone régulier dans le rapport 2/11



Annexe A

Programme n°1

```
input geometriesyr11;
u:=0.75cm;
vardef roulades(expr n, t)=
  save $;
  pair A[],I[],O,centre[],J[];
  path cc[],poly;
  picture $;
  $=image(
    O=u*(3,3);
    A1=u*(1,1);
    for j=2 upto n:
      A[j]=rotation(A[j-1],O,-360/n);
    endfor;
    trace A1
    for k=2 upto n:
      --A[k]
    endfor
    --cycle withpen pencircle scaled 1.25bp withcolor gris;
    trace droite(A1,A[n]);
    for k=0 upto n-2:
      centre[k]=A[n] shifted(k*(A[n]-A[1]));
    endfor;
    I0=A1;
    for k=0 upto (n-2):
      for j=1 upto (360/n):
        I[k*(360/n)+j]=rotation(I[k*(360/n)],centre[k],-j);
      endfor;
    endfor;
    J0=O;
    for k=0 upto (n-2):
      for j=1 upto (360/n):
        J[k*(360/n)+j]=rotation(J[k*(360/n)],centre[k],-j);
      endfor;
    endfor;
    cc1=I[0]
    for j=1 upto (360-360/n):
      ..I[j]
    endfor
    ;
    trace cc1 withcolor orange;
```

```
    if t=1:
      marque_p:="plein";
      for k=0 upto n-2:
        for j=10 step 20 until (360/n):
          pointe(I[k*(360/n)+j]);
          trace centre[k]--
          for p=1 upto n-1:
            rotation(centre[k],J[k*(360/n)+j],-p*(360/n))--
          endfor
          cycle dashed evenly withcolor 0.4[blanc,bleu];
        endfor;
      endfor;
    fi;
    );
  $:=rotation($,A1,-angle(A[n]-A[1]));
  $
enddef;
```

Annexe B

Programme n°2

```
input geometriesyr11;

input marith;%à utiliser pour les "grandes valeurs" des côtés.

vardef roulades(expr n,m,num,deno)=
  save $;
  picture $;
  pair O,A[],B[];
  path cc,poly;
  %numeric totalcen,totaltour
  O=(0,0);
  %polygone sur lequel on roule
  B1=u*(5,-5);
  %n=8;
  for k=2 upto n:
    B[k]=rotation(B1,O,(k-1)*360/n);
  endfor;
  B[n+1]=B1;
  trace B1
  for k=2 upto n:
    --B[k]
  endfor
  --cycle withcolor bleu;
  %polygone roulant
  A1=B1;
  A2=(num/deno)[B1,B2];
  for k=3 upto m:
    A[k]=rotation(A[k-2],A[k-1],(180*m-360)/m);
  endfor;
  trace A1
  for k=2 upto m:
    --A[k]
  endfor
  --cycle withcolor violet;
  %définition des centres de rotation
  pair cen[];
  cen[0]=B1;
  totalcen:=0;
  for k=0 upto n-1:
    for j=1 upto deno:
```

```
    cen[deno*k+j]:=(j/deno)[B[k+1],B[k+2]];
    totalcen:=totalcen+1;
  endfor;
endfor;
marque_p:="plein";
for k=0 upto totalcen:
  pointe(cen[k]);
endfor;
totaltour:=num*m;
totalcentre:=totalcen;
for p=1 upto totaltour:
  for k=0 upto totalcen:
    cen[p*totalcen+k]=cen[k];
    totalcentre:=totalcentre+1;
  endfor;
endfor;
%calcul et tracé de la trajectoire
pair I[];
I0=A1;
ang:=360/m;
angn:=360/n;
p:=0;
$=image(
  for k=num step num until (num*m*totalcen):
        %show k;
    if (k-deno*((k div deno))<num) and (k-deno*((k div deno))>0):
      if (deno*((k div deno)+1)-k)=0:
        %message("au revoir");
        I[p+1]=rotation(I[p],cen[deno*((k div deno)+1)],3.5*angn);
        trace arccercle(I[p],I[p+1],cen[deno*((k div deno)+1)]) withcolor orange;
        p:=p+1;
      else:
        %message("bonjour");
        I[p+1]=rotation(I[p],cen[deno*(k div deno)],angn);
        I[p+2]=rotation(I[p+1],cen[k],ang);
        trace arccercle(I[p],I[p+1],cen[deno*(k div deno)])--arccercle(I[p+1],I[p+2],cen[k])
withcolor orange;
        p:=p+2;
      fi;
    elseif (k mod deno)=0:
    %message("bien joue");
    I[p+1]=rotation(I[p],cen[k],ang+angn);
    if I[p+1]<>I[p]:
      trace arccercle(I[p],I[p+1],cen[k]) withcolor orange;
    fi;
    p:=p+1;
    else:
      %message("perdu");
      I[p+1]=rotation(I[p],cen[k],ang);
      trace arccercle(I[p],I[p+1],cen[k]) withcolor orange;
      p:=p+1;
    fi;
  endfor;
  );
$
```

```
enddef;
```

Table des figures