pst-anamorphosis : extensions possibles

Jürgen Gilg, Manuel Luque, Jean-Michel Sarlat

21 octobre 2011

Abstract

Il est relativement facile d'adapter les commandes de PStricks à pst-anamorphosis. C'est ainsi que la plupart des macros de pst-anamorphosis-add sont des adaptations de celles de PStricks. Toutes ne l'on pas été pour ne pas alourdir inutilement le package. Nous allons détailler sur un exemple la façon de procéder.

Remarque : toutes les macros ne sont pas adaptables, cela serait le cas si les anamorphoses transformaient une ligne droite en une ligne droite. Or, à part la perspective, ce n'est pas le cas. En conséquence, `\psline`, `\psframe`, `\pspolygon` doivent être réécrits en divisant le segment initial en un grand nombre de petits segments : pour `\pslineA` par exemple chaque segment initial est divisé en 200 parties et ceci quelle que soit sa longueur, dans l'idéal il faudrait tenir compte de la longueur du segment avant de choisir le nombre de segmentations.

1 \psparametricplotA

1.1 Adaptation

À partir d'une copie de \psparametricplot :

1. On remplace parout psparametricplot par psparametricplotA ;
2. dans le premier `\addto@pscode{%`

```
\addto@pscode{%
  \tx@optionsanamorphosis
  \tx@optionsanamorphosisAdd
    #3 %prefix PS code
    \psplot@init
```

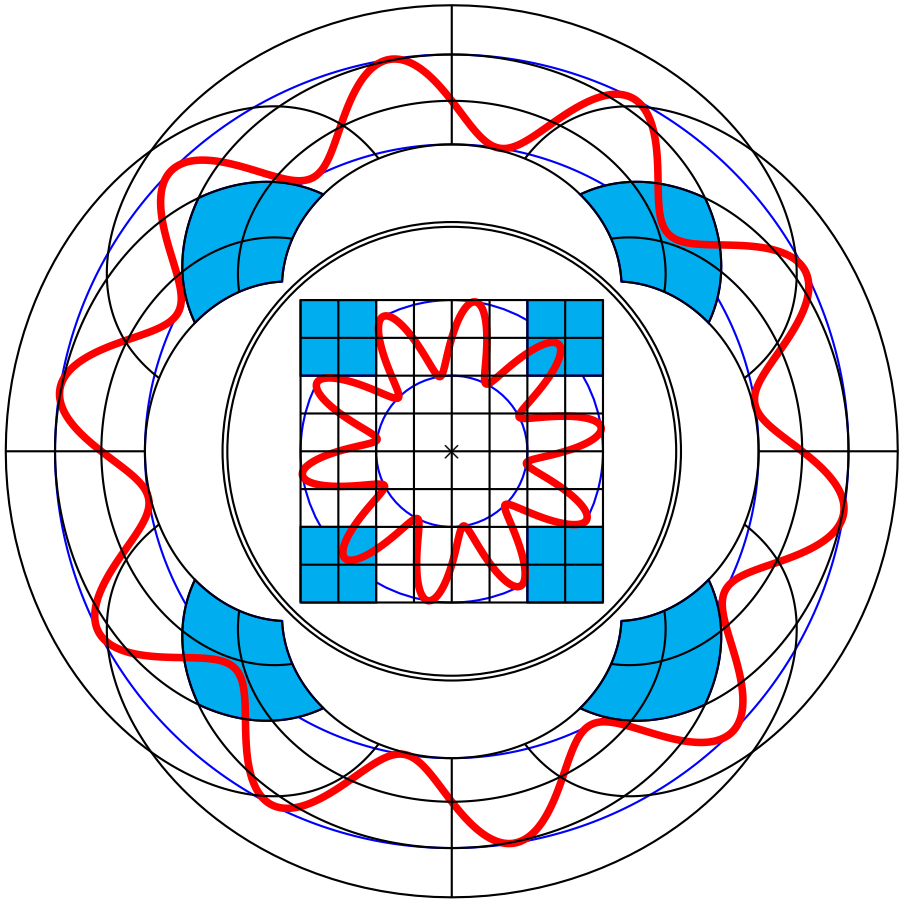
on introduit juste au début les variables de pst-anamorphosis ;

3. et pour terminer dans `/xy {`:

```
  /xy {
\ifPst@algebraic F@pstplot \else #5 \fi
\pst@number\psyunit mul exch
\pst@number\psxunit mul exch
tx@anamorphosisPathDict begin Anamorphose end
} def
```

on insère à la fin les calculs propres à l'anamorphose. C'est terminé, la macro est opérationnelle !

1.2 Exemple

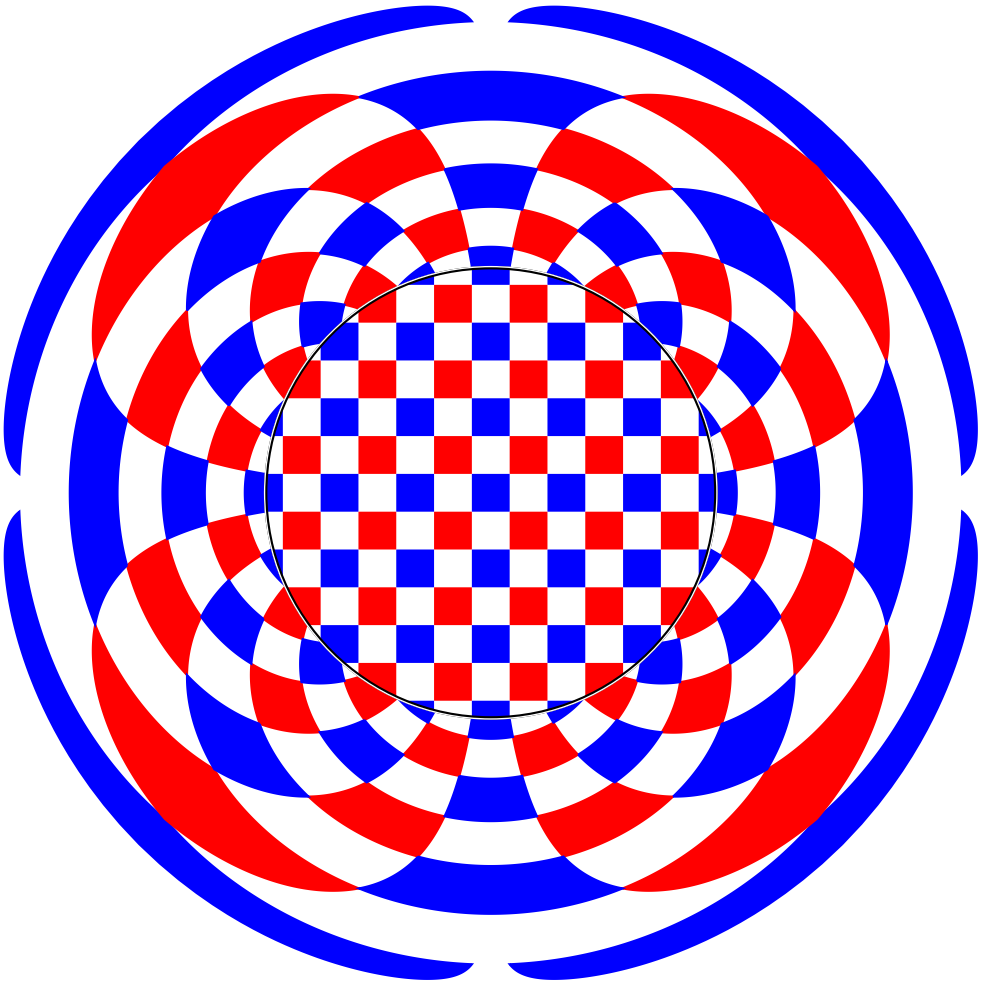


```
\parametricplotA[plotpoints=3600,linecolor=red,linewidth=1mm]{0}{360}{%
```

```
  /Radius 1.5 0.5 t 10 mul sin mul add def
  Radius t cos mul
  Radius t sin mul
    }
\parametricplot[plotpoints=3600,linecolor=red,linewidth=1mm]{0}{360}{%
  /Radius 1.5 0.5 t 10 mul sin mul add def
  Radius t cos mul
  Radius t sin mul
```

2 \psframeA

`\psframeA(x1,y1)(x2,y2)` a les mêmes propriétés que `\psframe`

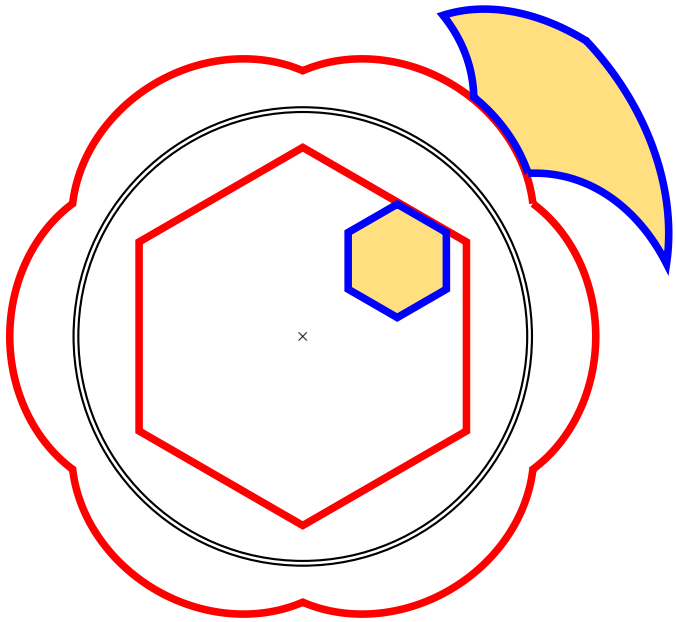


```
    \multido{\ry=-2.75+1,\rY=-2.25+1}{6}{
\multido{\n=-2.75+1.00,\N=-2.25+1.00}{7}{%
```

```
    \psframeA[fillstyle=solid,fillcolor=red,linecolor=red](\n,\ry)(\N,\rY)}
    }
\multido{\ry=-3.25+1,\rY=-2.75+1}{3}{
\multido{\n=-3.25+1.00,\N=-2.75+1.00}{7}{%
    \psframeA[fillstyle=solid,fillcolor=blue,linecolor=blue](\n,\ry)(\N,\rY)}
    }
```

3 \pspolygonA

En réalité la macro pspolygonA n'a pas été écrite. En effet il suffit de servir de \pslineA et de boucler le chemin en revenant au point origine.



```
\begin{pspicture}(-5,-5)(5,5)
\psset{type=conical,Rmirror=3}
\pspolygon[linecolor=red,linewidth=1mm](2.5;30)(2.5;90)(2.5;150)(2.5;210)(2.5;270)
\pslineA[linecolor=red,linewidth=1mm](2.5;30)(2.5;90)(2.5;150)(2.5;210)(2.5;270)(2
\rput(1.25,1){\pspolygon[linecolor=blue,fillstyle=solid,linewidth=1mm,fillcolor={[
\multido{\i=30+60,\I=1+1}{6}{%
    \pnode(!\i\space cos 0.75 mul 1.25 add \i\space sin 0.75 mul 1 add){A\I}
    }
```

```
\pslineA[linecolor=blue,fillstyle=solid,linewidth=1mm,fillcolor={[rgb]{1 0.875 0.5]
\pscircle[doubleline=true]{3}
\psdot[dotstyle=x](0,0)
\end{pspicture}
```

On remarquera que le `\rput(1.25,1)` doit être remplacé par un calcul adapté. La macro `\rputA` n'a pas été définie.