# jps2ps : version 0.14

*par Jean-Paul Vignault*
*Groupe des Utilisateurs de Linux Poitevins (GULP)*
(**jpv@melusine.eu.org**)
2 Juillet 2006

Ce document présente les différences entre la version 0.14 et la précédente.

## 1. Divers

Réécriture des commandes **sarc**, **sarcn**. Reprise de la commande **tripointarc** pour rendre plus naturel le sens de parcours : **A B C r tripointarc** va tracer l'arc de cercle de centre $B$ de rayon $r$, dans le sens trigonométrique, de $A$ vers $B$.

## 2. Chemins paramétrés

### 2.1 - L'objet *chemin paramétré*

Un objet *chemin paramétré* est une structure complexe représentant une application du type :

$$\begin{aligned} f\ : I &\to \mathbb{R}^2 \\ t &\mapsto (x, y) \end{aligned}$$

Cet objet possède 2 composantes : le tableau des valeurs du paramètre, et le tableau des points $(x, y)$ correspondants. Dans la pratique, les coordonnées des points sont exprimées soit dans le repère jps, soit dans le repère postscript sous-jacent. Les commandes **ppathtopath** et **pathtoppath** permettent de passer d'un type de chemin à l'autre, et les commandes **pathpointstable** et **pathparamtable** permettent d'accéder en lecture à chacune des composantes de l'objet.

La commande **drawpath** permet de dessiner un chemin paramétré passé en argument.

$pathobj_1$ **ppathtopath** $pathobj_2$ $\longrightarrow$ transforme le chemin paramétré $pathobj_1$ exprimé dans le repère postscript en le chemin paramétré $pathobj_2$ exprimé dans le repère jps

$pathobj_1$ **pathtoppath** $pathobj_2$ $\longrightarrow$ transforme le chemin paramétré $pathobj_1$ exprimé dans le repère jps en le chemin paramétré $pathobj_2$ exprimé dans le repère postscript

$pathobj$ **pathpointstable** *array* $\longrightarrow$ *array* est le tableau de points du chemin paramétré $pathobj$

$pathobj$ **pathparamtable** *array* $\longrightarrow$ *array* est le tableau des paramètres du chemin paramétré $pathobj$

$pathobj$ $string$ **drawpath** $-$ $\longrightarrow$ Dessine le chemin représenté par $pathobj$. $string$ est un paramètre optionnel indiquant le type de terminaison de ligne

### 2.2 - Création d'un chemin paramétré

Chaque fois que l'une des commandes de tracé du format jps est exécutée (**frame**, **droite**, **cercle**, **ellipse**, **courbe**, **courbeparam**, **sarc**, **sarcn**, **tripointarc**, **draw**, **bezier_curve**), le chemin correspondant est sauvegardé sous forme de chemin paramétré, le paramètre appartenant à l'intervalle $[0; 100]$. Les commandes **lastpath** et **lastppath** permettent alors d'accéder à ce chemin, exprimé dans le repère jps ou dans le repère postscript sous-jacent.

On peut également créer un chemin paramétré à partir du chemin courant : **stockcurrentpath** transforme le chemin courant en chemin paramétré et réaffecte les commandes **lastpath** et **lastppath**. On dispose également de 4 autres commandes permettant d'accéder, sans modifier les variables **lastpath**, au chemin paramétré défini par le chemin courant.

$-$ **lastpath** $pathobj$ $\longrightarrow$ l'objet path, coordonnées dans le repère jps, associé au dernier chemin dessiné

$-$ **lastppath** $pathobj$ $\longrightarrow$ l'objet path, coordonnées dans le repère postscript, associé au dernier chemin dessiné

$-$ **stockcurrentpath** $-$ $\longrightarrow$ sauvegarde le chemin courant sous forme de chemin paramétré, et réaffecte **lastpath** et **lastppath** en conséquence

$-$ **currentpathpointstable** *array* $\longrightarrow$ tableau des points définissant le chemin courant dans le repère jps

$-$ **currentppathpointstable** *array* $\longrightarrow$ tableau des points définissant le chemin courant dans le repère postscript

$-$ **currentpathobj** $pathobj$ $\longrightarrow$ chemin paramétré courant (dans le repère jps)

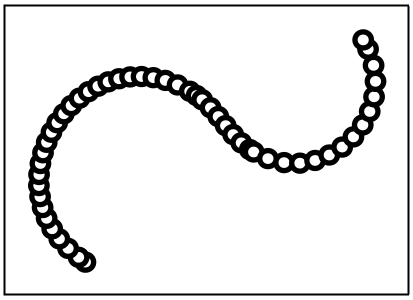
$-$ **currentppathobj** $pathobj$ $\longrightarrow$ chemin paramétré courant (dans le repère postscript)

### 2.3 - Points d'un chemin paramétré

$pathobj$ **`pathstartpoint`** $A \longrightarrow A$ est le premier point du chemin paramétré $pathobj$

$pathobj$ **`pathendpoint`** $B \longrightarrow B$ est le dernier point du chemin paramétré $pathobj$

$t$ $pathobj$ **`pathpoint`** $M \longrightarrow M$ est le point de paramètre $t$ du chemin paramétré $pathobj$



*source jps*

```
autocrop
[-3 -3 .. -1 0 .. 0 -1 .. 2 1] draw
/path1 lastpath def
[0 2 100 {} for] {path1 pathpoint circ2} apply
```

### 2.4 - Opérations sur les chemins paramétrés

$pathobj_1$ **`reversepathobj`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin obtenu à partir de $pathobj_1$ en inversant le sens de parcours

$pathobj$ **`pathlongueur`** $\ell \longrightarrow$ longueur de chemin parmétré $pathobj$

$t$ $pathobj$ **`pathlongueurs`** $\ell_1$ $\ell_2 \longrightarrow \ell_1$ et $\ell_2$ sont les longueurs respectives des sous-chemins $\widehat{AM}$ et $\widehat{MB}$, où $M$ est le point de paramètre $t$ du chemin paramétré $pathobj$, et où $A$ et $B$ sont respectivement les premier et dernier points de $pathobj$

$t$ $pathobj$ **`pathslongueur`** $\ell \longrightarrow$ longueur du sous-chemin $\widehat{AM}$, où $M$ est le point de paramètre $t$ du chemin paramétré $pathobj$, et $A$ est le premier point de $pathobj$

$t$ $pathobj$ **`pathelongueur`** $\ell \longrightarrow$ longueur du sous-chemin $\widehat{MB}$, où $M$ est le point de paramètre $t$ du chemin paramétré $pathobj$, et $B$ est le dernier point de $pathobj$

$t$ $pathobj$ **`splitpath`** $pathobj_1$ $pathobj_2 \longrightarrow$ sépare, à partir du point de paramètre $t$, le chemin paramétré $pathobj$ en 2 sous-chemins paramétrés

$pathobj_1$ **`normalizepath`** $pathobj_2 \longrightarrow$ Les chemins décrits par $pathobj_1$ et $pathobj_2$ sont confondus, mais le tableau des paramètres de $pathobj_2$ a été modifié, de telle facon que ceux-ci soient dans l'intervalle $[0;100]$, et qu'ils soient répartis de façon proportionnelle à la longueur du chemin

$pathobj_1$ $u$ **`translatepath`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin image de $pathobj_1$ par la translation de vecteur $u$

$pathobj_1$ **`projxpath`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin projeté orthogonal de $pathobj_1$ sur l'axe $Ox$

$pathobj_1$ **`projypath`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin projeté orthogonal de $pathobj_1$ sur l'axe $Oy$

$pathobj_1$ $D$ **`orthoprojpath`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin projeté orthogonal de $pathobj_1$ sur la droite $D$

$pathobj_1$ $I$ $\alpha$ **`rotatepath`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin image de $pathobj_1$ par la rotation de centr(e $I$ et d'angle $\alpha$

$pathobj_1$ $I$ $k$ **`hompath`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin image de $pathobj_1$ par l'homothétie de centre $I$ et de rapport $k$

$pathobj_1$ $I$ **`sympath`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin image de $pathobj_1$ par la symétrie de centre $I$

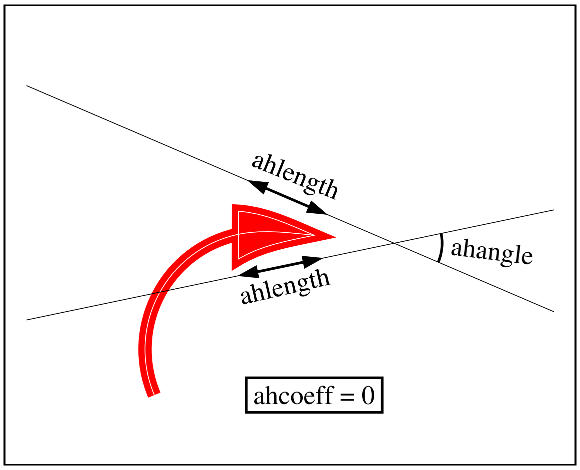
$pathobj_1$ $D$ **`axesympath`** $pathobj_2 \longrightarrow pathobj_2$ est le chemin image de $pathobj_1$ par la symétrie axiale d'axe $D$

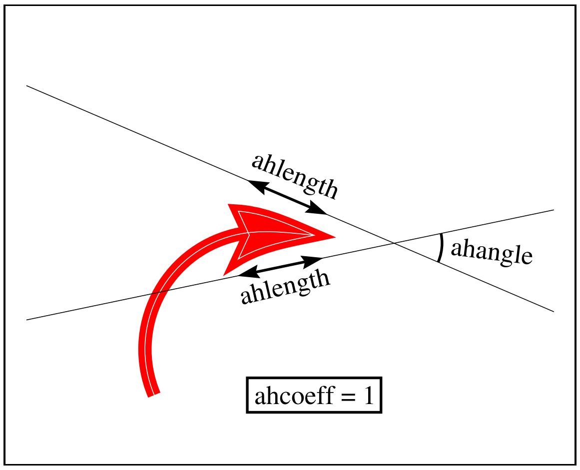
## 3. Les flèches

Le dessin des flèches s'obtient avec les options de tracés de fin de ligne. Trois paramètres permettent de modifier ces flèches :

- *ahlength* : longueur en picas des flèches de fin de ligne. **valeur par défaut :** 6
- *ahangle* : angle au sommet des flèches de fin de ligne. **valeur par défaut :** 30
- *ahcoeff* : coefficient entre 0 et 1 indiquant le niveau de décrochement sur la base de la flèche. **valeur par défaut :** 1

Pour obtenir les dessins attendus, il est nécessaire que la longueur (en picas) du chemin soit supérieur à *ahlength*.




En interne, le format utilise les commandes suivantes :

– **`arrowpath0`** *pathobj* $\longrightarrow$ chemin correspondant à l'axe d'une flèche potentielle au début du chemin courant lors du dernier appel à **`arrowpaths`**

– **`arrowpath1`** *pathobj* $\longrightarrow$ chemin correspondant à l'axe d'une flèche potentielle à la fin du chemin courant lors du dernier appel à **`arrowpaths`**

*pathobj* **`draw_arrow`** – $\longrightarrow$ considère *pathobj* comme l'axe d'une flèche et dessine la flèche correspondante

– **`arrowpaths`** $pathobj_1$ $pathobj_2$ $\longrightarrow$ depose sur la pile les 2 sous chemins en provenance du chemin courant necessaires à **`gere_arrowhead`**