Interface Mathematica-OpenCV

1. Généralités

Le module CV est chargé par : `Needs["CV`"]`. L'ouverture de l'interface se fait avec `CVOpen[]`.

Les structures de données utilisées par OpenCV (IplImage, CvMat, CvCapture, etc) sont manipulées dans le noyau Mathematica par leur référence. Celle-ci est un entier supérieur ou égal à 1. La référence 0 est aussi utilisée pour représenter un argument nul (équivalent de null en C).

Les données peuvent être récupérées dans Mathematica par différentes fonctions.

L'interface doit être fermée avec `CVClose[]`. Cette fonction libère la mémoire associée aux structures de données puis ferme la liaison MathLink.

Les fonctions se classent en quatre catégories :

- ▷ Fonctions d'interface avec les fonctions d'OpenCV.
- ▷ Fonctions complémentaires de traitement d'image.
- ▷ Fonctions d'interface graphique évoluée (implémentées en Java).
- ▷ Fonctions d'échange de données avec Mathematica.

La syntaxe des fonctions d'interface est très proche de celle des fonctions C de la bibliothèque OpenCV. La principale différence réside dans la manipulation de références (sous forme d'entiers) au lieu des pointeurs. Les structures de données simples, comme les coordonnées de points, sont directement transmises sous forme de liste.

Les fonctions d'échange de données permettent de récupérer les gros volumes de données associés aux images ou à d'autres structures comme les contours, afin de les traiter directement dans Mathematica, par exemple pour obtenir des représentations graphiques.

2. Fonctions de base

2.a. Structures de données

Les images sont représentées par deux structures de données différentes : IplImage et CvMat. La plupart des fonctions agissent indifféremment sur l'une ou l'autre. On désigne par CvArr un argument qui peut être soit IplImage, soit CvMat, soit CvSeq.

```
array=CVGetArray[arr,channel]
```

Obtenir la matrice des valeurs d'une couche d'une matrice ou d'une image.

- ▷ `arr (Integer)` : référence d'un CvArr (CvMat ou IplImage).
- ▷ `channel (Integer)` : couche à lire (de 1 à 4).
- ▷ `array (Real List)` : matrice des valeurs.

```
array=CVGetImageArray[img,norm,order]
```

Obtenir un tableau de valeurs correspondant à une image.

▷ `img (Integer)` : référence d'une image IplImage

▷ `norm (Integer)` : 1 pour obtenir des valeurs entre 0 et 1, 0 sinon

▷ `order (Integer)` : 1 pour laisser les couches dans l'ordre original, -1 pour inverser l'ordre des couches

▷ `array (Real Array)` : matrice de n-uplets, où n est le nombre de couches de l'image.

```
CVReleaseImage[img]
```

Libération de la mémoire utilisée par une IplImage devenue inutile. Cette fonction doit être utilisée systématiquement dans les applications qui manipulent un grand nombre d'images différentes (par exemple traitement d'une vidéo).

▷ `img (Integer)` : référence d'une IplImage.

```
CVSetArray[arr,channel,array]
```

Affecter des valeurs à un CvArr (matrice CvMat ou image IplImage). La fonction opère sur des matrices de dimension quelconque.

▷ `arr (Integer)` : référence d'un CvArr (CvMat ou IplImage).

▷ `channel (Integer)` : couche à remplir (de 1 à 4).

▷ `array (Real List)` : matrice contenant les valeurs, sous forme de listes imbriquées.

2.b. Opérations sur les tableaux

```
CVConvertScale[src,dst,scale,shift]
```

Conversion de src vers dst avec modification des valeurs : dst=src*scale+shift. La transformation s'applique aux images multicouches.

▷ `src (Integer)` : référence d'un CvArr. Image source.

▷ `dst (Integer)` : référence d'un CvArr. Image de destination.

▷ `scale (Real)` : facteur multiplicatif.

▷ `shift (Real)` : décalage.

```
ref=CVCreateImage[{width,height},depth,channels]
```

Créer une image IplImage.

▷ `width (Integer)` : largeur de l'image

- ▷ `height (Integer)` : hauteur de l'image
- ▷ `depth (Integer)` : profondeur des couches. Utiliser les constantes IplDepth8U, IplDepth16U, etc.
- ▷ `channels (Integer)` : nombre de couches
- ▷ `ref (Integer)` : référence de la structure IplImage créée

```
mat=CVCreateMat[rows,cols,type,channels]
```

Créer une matrice CvMat. Les matrices servent à manipuler des parties d'image ou à effectuer différents calculs.

- ▷ `rows (Integer)` : nombre de lignes.
- ▷ `cols (Integer)` : nombre de colonnes.
- ▷ `type (Integer)` : type de l'image. Utiliser les constantes CV8U, CV8S, CV16U, CV16S, CV32S, CV32F ou CV64F.
- ▷ `channels (Integer)` : nombre de couches (entre 1 et 4).
- ▷ `mat (Integer)` : référence de la structure CvMat créée.

```
CVMinMaxLoc[arr]
```

Obtenir les valeurs mimimales et maximales, ainsi que leur position, d'un CvArr à une couche et à deux dimensions.

- ▷ `arr (Integer)` : référence d'un CvArr (CvMat ou IplImage).
- ▷ `m (Real List)` : valeurs min/max et positions sous la forme min,max,minX,minY,maxX,maxY

```
s=CVGetSize[img]
```

Obtenir la taille d'une image.

- ▷ `img (Integer)` : référence d'un CvArr.
- ▷ `s (Integer List)` : taille sous la forme largeur,hauteur.

```
CVResetImageROI[img]
```

Désactiver la région d'intérêt.

- ▷ `img (Integer)` : référence de IplImage.

```
CVSetImageROI[img,rect]
```

Définir une région d'intérêt (ROI) pour une image. La plupart des fonctions d'OpenCV opèrent sur la région d'intérêt lorsque'elle est définie.

- ▷ `img (Integer)` : référence de IplImage.
- ▷ `rect (Integer List)` : rectangle sous la forme x,y,width,height.

```
CVSplit[src,dst0,dst1,dst2,dst3]
```

Recopier les couches d'une image dans des images monochromes.

▷ `src (Integer)` : référence d'un CvArr. Image source.

▷ `dst0, dst1, dst2, dst3 (Integer)` : références de CvArr. Images de destination.

2.c. Structures dynamiques

```
mem=CVCreateMemStorage[blocksize]
```

Créer une zone de mémoire pour les structures de données dynamiques.

▷ `blocksize (Integer)` : taille des blocs en octets. Si la valeur est 0, une taille par défaut est adoptée.

▷ `mem (Integer)` : référence d'un CvMemStorage.

```
next=CVGetSeqHNext[seq]
```

Obtenir l'élément suivant disposé horizontalement dans une structure de données. Chaque élément est une séquence.

▷ `seq (Integer)` : référence de CvSeq. Élément d'une structure de données.

▷ `next (Integer)` : référence de CvSeq. Élément suivant.

```
prev=CVGetSeqHPrev[seq]
```

Obtenir l'élément précédent disposé horizontalement dans une structure de données. Chaque élément est une séquence.

▷ `seq (Integer)` : référence de CvSeq. Élément d'une structure de données.

▷ `prev (Integer)` : référence de CvSeq. Élément précédent.

```
next=CVGetSeqVNext[seq]
```

Obtenir l'élément suivant disposé verticalement dans une structure de données. Chaque élément est une séquence.

▷ `seq (Integer)` : référence de CvSeq. Élément d'une structure de données.

▷ `next (Integer)` : référence de CvSeq. Élément suivant.

```
prev=CVGetSeqVPrev[seq]
```

Obtenir l'élément précédent disposé verticalement dans une structure de données. Chaque élément est une séquence.

▷ `seq (Integer)` : référence de CvSeq. Élément d'une structure de données.

▷ `prev (Integer)` : référence de CvSeq. Élément précédent.

3. Traitement d'image et vision informatique

3.a. Filtrage

```
CVFilter2D[src,dst,kernel]
```

Filtrer une image par un noyau de convolution. Celui-ci doit être une matrice de taille impaire, par exemple 3x3.

- ▷ `src (Integer)` : référence d'un CvArr. Image source.
- ▷ `dst (Integer)` : référence d'un CvArr. Image de destination.
- ▷ `kernel (Integer)` : référence d'une CvMat. Noyau de convolution.

```
CVLaplace[src,dst,apertureSize]
```

Calculer le Laplacien d'une image à une couche.

- ▷ `src (Integer)` : référence d'un CvArr. Image source.
- ▷ `dst (Integer)` : référence d'un CvArr. Image de destination.
- ▷ `apertureSize (Integer)` : taille du noyau de convolution (1,3,5 ou 7).

```
CVSobel[src,dst,xorder,yorder,apertureSize]
```

Calculer une dérivée d'une image à une couche par l'opérateur de Sobel.

- ▷ `src (Integer)` : référence d'un CvArr. Image source.
- ▷ `dst (Integer)` : référence d'un CvArr. Image de destination.
- ▷ `xorder (Integer)` : ordre de la dérivée selon x.
- ▷ `yorder (Integer)` : ordre de la dérivée selon y.
- ▷ `apertureSize (Integer)` : taille du noyau de convolution (1,3,5 ou 7).

3.b. Traitements divers

```
CVCvtColor[src,dst,convert]
```

Conversion entre deux espaces colorimétriques.

- ▷ `src (Integer)` : référence d'un CvArr. Image source.
- ▷ `dst (Integer)` : référence d'un CvArr. Image de destination.
- ▷ : code de conversion. Utiliser les constantes CVBGR2HSV, CVHSV2BGR, etc.

3.c. Histogrammes

```
CVCalcBackProject[img,back,hist]
```

Rétroprojection d'un histogramme sur une image.

▷ `img (Integer List)` : liste de références de IplImage, autant que l'histogramme comporte de dimensions.

▷ `back (Integer)` : référence d'un CvArr. Destination de la rétroprojection.

▷ `hist (Integer)` : référence d'un CVHistogram

```
hist=CVCreateHist[dims,type,ranges,uniform]
```

Création d'un histogramme à n dimensions.

▷ `dims (Integer List)` : liste des dimensions.

▷ `type (Integer)` : type d'histogramme (CVHistArray, CVHistSparse ou CVHistUniform).

▷ `uniform (Integer)` : 0 ou 1.

▷ `hist (Integer)` : référence de la structure CVHistogram créée.

```
CVCalcHist[img,hist,accumulate,mask]
```

Calcul d'histogrammes.

▷ `img (Integer List)` : liste de références de IplImage, autant que l'histogramme comporte de dimensions.

▷ `hist (Integer)` : référence d'un CVHistogram

▷ `accumulate (Integer)` : 0 ou 1. Si égal à 1, l'histogramme n'est pas effacé, ce qui permet d'accumuler les histogrammes de plusieurs images.

▷ `mask (Integer)` : référence d'un CvArr ou 0. Masque indiquant les pixels à prendre en compte.

```
CVConvertHistScale[hist,scale,shift]
```

Transformation des valeurs d'un histogramme : newhist=hist*scale+shift.

▷ `hist (Integer)` : référence de l'histogramme (CvHistogram).

▷ `scale (Real)` : facteur multiplicatif.

▷ `shift (Real)` : décalage.

```
img=CVGetHueSatHistImage[hist,pixels]
```

Obtenir une représentation graphique sous forme d'image d'un histogramme 2D Hue-Sat.

▷ `hist (Integer)` : référence de l'histogramme (CvHistogram).

▷ `pixels (Integer)` : taille en pixels de chaque case de l'histogramme.

▷ `img (Integer)` : référence de l'image IplImage créée.

3.d. Détection d'éléments

```
CVCanny[image,edges,threshold1,threshold2,apertureSize]
```

Détecter les bords par la méthode de Canny.

- ▷ `image (Integer)` : référence d'un CvArr. Image source à une couche.
- ▷ `edges (Integer)` : référence d'un CvArr. Image de destination : image binaire comportant les bords.
- ▷ `threshold1 (Real)` : seuil bas.
- ▷ `threshold2 (Real)` : seuil haut.
- ▷ `apertureSize (Integer)` : taille du noyau de convolution (1,3,5 ou 7).

```
seq=CVHoughLines2[image,storage,method,rho,theta,threshold,param1,param2]
```

Détection de points alignés dans une image binaire, par la transformée de Hough.

- ▷ `image (Integer)` : référence d'un CvArr. Image source à une couche.
- ▷ `storage (Integer)` : référence du CvMemory utilisé pour stocker les lignes obtenues.
- ▷ `method (Integer)` : les méthodes possibles sont CVHoughStandard, CVHoug-Probabilistic, CVHoughMultiScale.
- ▷ `rho (Real)` : résolution en unité de pixels.
- ▷ `theta (Real)` : résolution en radians.
- ▷ `threshold (Integer)` : seuil définissant la valeur minimale dans le plan accumulateur pour qu'une ligne soit retenue.
- ▷ `param1 (Real)` : dans la méthode CVHougProbabilistic : longueur de segment minimale. Dans la méthode CVHoughMultiScale : diviseur de la résolution en ρ.
- ▷ `param2 (Real)` : dans la méthode CVHougProbabilistic : distance maximale entre deux segments alignés pour qu'ils soient considérés comme un seul segment. Dans la méthode CVHoughMultiScale : diviseur de la résolution en θ.
- ▷ `seq (Integer)` : référence d'une CvSeq. Séquence comportant les lignes détectées.

```
lines=CVGetHoughLines[seq,method]
```

Obtenir les droites détectées par `CVHoughLines2` sous forme d'une liste de doublets.

- ▷ `seq (Integer)` : référence d'une CvSeq. Séquence comportant les lignes détectées, renvoyée par CVHoughLines2.
- ▷ `method (Integer)` : méthode utilisée dans CVHoughLines2.
- ▷ `lines (Array)` : pour la méthode CVHoughStandard : liste de doublets (ρ, θ).

3.e. Analyse de structures et description de formes

```
seq=CVFindContours[img,mem,mode,method,offset]
```

Obtenir les contours présents dans une image. Celle-ci est interprétée comme une image binaire : les valeurs 0 restent 0, les valeurs non nuls sont interprétées comme 1. Les contours des zones connexes formées de 1 sont recherchés. La fonction renvoit le premier contour. Pour accéder aux autres contours, utiliser CVGetSeqHNext et CVGetSeqVNext.

- ▷ `img (Integer)` : référence de l'image source (CvArr, 8 bits, une couche).
- ▷ `mem (Integer)` : référence du CvMemory utilisé pour stocker les données.
- ▷ `mode (Integer)` : CvRetrExternal : contours externes seulement. CVRetrList : tous les contours sous forme d'une liste chaînée. CVRetrCcomp : tous les contours sous forme d'une structure à deux niveaux. CvRetrTree : tous les contours sous forme d'un arbre.
- ▷ `method (Integer)` : méthode d'approximation. CVChainCode, CVChainApproxNone, CVChainApproxSimple, CVChainApproxTC89L1, CVChainApproxTC89KCOS, CVLinkRuns.
- ▷ `offset (Integer List)` : coordonnées x,y du décalage à appliquer à chaque points du contour.
- ▷ `seq (Integer)` : premier contour (référence d'une CvSeq).

```
c=CVGetContour[seq]
```

Obtenir la liste des points d'un contour.

- ▷ `seq (Integer)` : référence de la séquence (CvSeq) contenant le contour.
- ▷ `c (Real List)` : liste de points.

```
p=CVContourPerimeter[seq]
```

Obtenir le contour d'un périmètre.

- ▷ `seq (Integer)` : référence de la séquence (CvSeq) contenant le contour.
- ▷ `p (Real)` : périmètre du contour.

4. Interface graphique et lecture-écriture des fichiers

4.a. Interface graphique

```
edit=CVJImageEditor[name,modal]
```

Création d'une fenêtre pour éditer des images (relevé de points, de zones, etc). Si la fenêtre est modale, le fil d'exécution reprend lorsque la fenêtre est fermée par l'utilisateur.

- ▷ `name (String)` : nom de la fenêtre.

▷ `modal (Boolean)` : fenêtre modale ou pas.

▷ `edit (Java class ShowImage)` : référence de l'éditeur.

```
CVJShowImage[edit,img]
```

Afficher une image dans un éditeur d'image de la classe ShowImage.

▷ `edit (Java class ShowImage)` : référence de l'éditeur.

▷ `img (Integer)` : référence d'une IplImage.

```
CVJShowVideoFrames[edit,capture,skip]
```

Afficher une vidéo image par image dans un éditeur d'image. La flèche verte permet à l'utilisateur de passer à l'image suivante.

▷ `edit (Java class ShowImage)` : référence de l'éditeur.

▷ `capture (Integer)` : référence d'une CvCapture.

▷ `skip (Integer)` : nombre d'images du flux vidéo à sauter entre deux images consécutives affichées.

```
list=CVJGetPixelPointList[edit]
```

Obtenir les coordonnées des points sélectionnés par l'utilisateur dans l'éditeur d'image, en unité de pixel. Ces coordonnées sont de type flottant (Real) car la sélection d'un point peut se faire entre deux pixels sur une image grossie. Convertir en Integer avec Floor si nécessaire.

▷ `edit (Java class ShowImage)` : référence de l'éditeur.

▷ `list (Real List)` : liste de doublets, coordonnées des points sélectionnés.

```
list=CVJGetUserPointList[edit]
```

Obtenir les coordonnées des points sélectionnés par l'utilisateur dans l'éditeur d'image, dans le repère défini par l'ulisateur.

▷ `edit (Java class ShowImage)` : référence de l'éditeur.

▷ `list (Real List)` : liste de doublets, coordonnées des points sélectionnés.

```
list=CVJGetPixelRectangleList[edit]
```

Obtenir les rectangles saisis par l'utilisateur dans un éditeur d'image, en unité de pixel.

▷ `edit (Java class ShowImage)` : référence de l'éditeur.

▷ `list (Real List)` : liste de 4-uplets de la forme x,y,width,height spécifiant les rectangles sélectionnés.

```
list=CVJGetUserRectangleList[edit]
```

Obtenir les rectangles saisis par l'utilisateur dans un éditeur d'image, dans le repère défini par l'utilisateur. Le rectangle a ses côtés parallèles à ceux de l'image, et non aux axes du repère.

- ▷ `edit (Java class ShowImage)` : référence de l'éditeur.
- ▷ `list (Real List)` : liste de 4-uplets de la forme x,y,width,height spécifiant les rectangles sélectionnés.

4.b. Lecture-écriture des images et vidéo

```
img=CVLoadImage[filename,iscolor]
```

Lecture d'une image à partir d'un fichier. Les formats possibles sont JPG, PNG, BMP, TIF, PPM, SR. Le fichier est recherché dans le répertoire courant, lequel peut être modifié avec la fonction `SetDirectory`. Le chemin d'accès complet peut aussi être spécifié.

- ▷ `filename (String)` : nom du fichier ou chemin d'accès complet.
- ▷ `iscolor (Integer)` : une des constantes CVLoadImageUnchanged, CVLoadImageGrayScale, CVLoadImageColor.
- ▷ `img (Integer)` : référence d'une IplImage. Si la lecture a échouée, la valeur 0 est renvoyée.

```
capture=CVCaptureFromFile[filename]
```

Lecture d'une vidéo à partir d'un fichier.

- ▷ `filename (String)` : nom du fichier ou chemin d'accès complet.
- ▷ `capture (Integer)` : référence d'une CvCapture. Si la lecture a échouée, la valeur 0 est renvoyée.

```
img=CVGetFrame[capture,skip]
```

Lecture d'une image dans une vidéo. L'image obtenue est une copie. En cas de lecture séquentielle de toute la vidéo, ne pas oublier de libérer la mémoire occupée par l'image avec `CVReleaseImage`.

- ▷ `capture (Integer)` : référence d'une CvCapture.
- ▷ `skip (Integer)` : nombre d'images à sauter à partir de la position courante.
- ▷ `img (Integer)` : référence d'une IplImage, copie de l'image. Si la fin du flux vidéo est atteinte, la valeur 0 est renvoyée.