

## readwriteshow <ims> <imd>

Écrire un programme `readwriteshow.cpp`

- ce programme permet de charger une image donnée en paramètre `<ims>`
- l’affiche dans une fenêtre avec le nom du fichier et donne les dimensions de l’image sur la sortie standard
- l’application crée également une image 200 de largeur et 100 de hauteur, de couleur magenta uniforme et l’affiche dans une nouvelle fenêtre
- sauvegarde cette image avec le nom donnée en paramètre `imd`
- le programme vérifie, l’existence du fichier `ims` et des paramètres
- vérifier l’image produite en la visualisant avec `pvisu`

## crop <ims> <i0> <j0> <h> <w>

Écrire le programme `crop.cpp`

- ce programme extrait une sous-image de `ims` la position  $(i_0, j_0)$  et de largeur  $w$  et de hauteur  $h$
- faire le crop à la main : extraction pixels à pixels, sauvegarder l’image dans `crop.png`
- utiliser la classe `Rect` d’`opencv`, sauvegarder l’image dans `crop-ocv.png`
- vérifier que les deux méthodes donnent exactement le même résultat, visuellement en faisant une image différence entre votre méthode et celle d’`opencv`
- visualiser les images résultats via `pvisu`
- tester avec différents paramètres

## video-player <video-name>

Écrire le programme `video-player.cpp`

- ce programme affiche la vidéo `video-name`
- ce programme affiche dans le terminal, les informations suivantes sur la vidéo : `fps` (frames par seconde), `width`, `height`, la largeur et la hauteur des frames, le nombre total de frames de la vidéo
- ce programme affiche, pendant les 1000 premières frames, la frame courante
- créer une `slide bar` permettant de se déplacer dans la vidéo, au déplacement, affiche dans le terminal la frame destination

## pixels-loops <ims>

Écrire le programme `pixels-loops.cpp`

- ce programme mesure le temps d’exécution des différents types d’accès à une matrice de pixels (`Mat`) créée via l’image `ims`
- créer une très grande image via `convert` et l’option `-resize` (par ex.  $\times 1000$ )
- utiliser / comparer les méthodes `at()`, les itérateurs, les opérations matriciels (surcharge des opérateurs arithmétiques, par ex.  $*$ ,  $/$ ,  $+$ ,  $-$ , l’accès via les pointeurs et appliquer la formule suivante

$$I(x, y) = \frac{1}{64}I(x, y) * 64 + 64/2$$

- chercher un moyen de mesurer le temps d’exécution des différents type accès