Formation OpenCV

Philippe FOUBERT

Interactivité (1/3)

➔ Ligne de commande

Nom | Valeur par défaut | Description

```
static const char* keys = "{help h usage ? |      | print this message }"
                          "{@image1        |      | image1 for compare }"
                          "{@image2        |      | image2 for compare }"
                          "{@repeat        |      | number }"
                          "{path           |.     | path to file }"
                          "{fps            | -1.0 | fps for output video }"
                          "{N count        |100   | count of objects }"
                          "{ts timestamp   |      | use time stamp }";
```

Préfixe '@' pour un paramètre obligatoire

```
cv::CommandLineParser parser(argc, argv, keys);
parser.about("Application name v1.0.0");
if(parser.has("help")) { parser.printMessage(); return 0; }
int    N          = parser.get<int>("N");
double fps        = parser.get<double>("fps");
std::string path  = parser.get<std::string>("path");
use_time_stamp    = parser.has("timestamp");
std::string img1  = parser.get<std::string>(0);
std::string img2  = parser.get<std::string>(1);
int        repeat = parser.get<int>(2);
if(!parser.check()) { parser.printErrors(); return -1; }
```

Index ou nom

Utilisation : `./app -N=200 1.png 2.jpg 19 -ts`

Interactivité (2/3)

Clavier

```
char key = cv::waitkey(0)
```

Trackbar

```
cv::createTrackbar(
    const cv::String &trackbarName,  // Name of the created trackbar
    const cv::String &windowName,    // Attach to this window
    int *value,                      // Current position of the slider
    int  maxvalue,                   // Maximal position
    cv::TrackbarCallback onTrackbar = 0, // Pointer to the function to be called
                                     // when the slider position changes
    void *userdata = nullptr);       // Data to be passed to the callback
```

Avec une callback ayant pour prototype :

```
void onTrackbar(
    int   value,     // The current trackbar position
    void *userdata); // The custom data passed from createTrackbar()
```


Interactivité (3/3)

Souris

```
cv::setMouseCallback(
  const cv::String  &windowName, // The window name
  cv::MouseCallback *onMouse,    // The callback to be called
  void              *param = nullptr);  // Data to be passed to the callback
```

Avec une callback ayant pour prototype :

```
void onMouse(
  int event,    // EVENT_LBUTTONDOWN   EVENT_RBUTTONDOWN   EVENT_MBUTTONDOWN
                // EVENT_LBUTTONUP     EVENT_RBUTTONUP     EVENT_MBUTTONUP
                // EVENT_LBUTTONDBLCLK EVENT_RBUTTONDBLCLK EVENT_MBUTTONDBLCLK
                // EVENT_MOUSEMOVE
  int x,
  int y,
  int flags,    // EVENT_FLAG_CTRLKEY  EVENT_FLAG_SHIFTKEY EVENT_FLAG_ALTKEY
                // EVENT_FLAG_LBUTTON  EVENT_FLAG_RBUTTON  EVENT_FLAG_MBUTTON
  void *param);
```


A vous de jouer !

Interaction clavier - Ecrire une application qui :

a) charge une image passée en argument au programme
b) sur appui sur la touche ‘l’, fait pivoter l’image sur la gauche
c) sur appui sur la touche ‘r’, fait pivoter l’image sur la droite
d) sur appui sur la touche ‘s’, enregistre l’image obtenue