UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

Travaux dirigés Traitement d'images n°2

Transformations géométriques et colorimétriques

—Master 1—

▸ Exercice 1. *Rotation*

1. *Ouvrez, lisez et exécutez le fichier* `tp02_ex01.cpp` *avec l'image* `image_margin.jpg`.

2. *Créez une fonction permettant de générer une matrice de rotation* 3×3. *Cette fonction aura le prototype suivant :*

```
void get_rotation(cv::Mat &R, const float angle_in_degree)
```

   Il faudra convertir l'ange en radian en utilisant le nombre π *noté* `M_PI`, *n'oubliez pas d'inclure* `cmath`. *Vous pouvez initialiser la matrice* `R` *avec la commande suivante :*

```
R = cv::Mat(3, 3, CV_32FC1, cv::Scalar(0.0f));
```

3. *À l'aide de la fonction définie dans la question précédente, ajoutez une fonction permettant d'effectuer une rotation de* α *degrés autour le l'origine* $(0, 0)$ *de l'image. Pour opérer la matrice de transformation sur votre image, utilisez la commande* `cv::warpPerspective`.

4. *Créez une fonction permettant de générer une matrice de translation* 3×3 *avec le prototype :*

```
void get_translation(cv::Mat &T, const float t_x, const float t_y)
```

5. *Translatez votre image pour vérifier que votre matrice est correcte.*

6. *Créez une fonction permettant d'effectuer une rotation de l'image autour d'un point quelconque passé en paramètre (par exemple le centre de l'image). Votre fonction aura le prototype suivant :*

```
void rotate_image_around_point(cv::Mat &image,
                               const float angle_in_degree,
                               const float center_x,
                               const float center_y)
```

7. *Pour finir, sur l'image initiale, effectuez 20 rotations successives de 10 degrés chacune, puis 20 rotations de -10 degrés. On s'attend à retrouver la même image. Commentez.*

▸ **Exercice 2.** ***Balance des blancs***

1. *Ouvrez, lisez et exécutez le fichier* `tp02_ex02.cpp` *avec l'image* `wb_incandescent.png`*.*

2. *Un objet blanc pris en photo n'apparait pas forcément blanc sur la photo suivant la teinte de l'illuminant de la scène. Il est alors nécessaire d'opérer une balance des blancs qui transforme l'image de telle sorte qu'un objet blanc apparaisse blanc. Le soucis, c'est qu'on ne sait pas vraiment identifier les objets sensés être blancs sur l'image. Il existe plusieurs solutions permettant de contourner le problème. Parmi les plus simple, il y a la méthode de "gray world assumption" qui suppose que la couleur moyenne d'une image est (souvent) gris. Pour effectuer la balance des blancs, il suffit alors de :*

   (a) *calculer la couleur moyenne* (r_m, g_m, b_m) *de l'image.*

   (b) *modifier chaque pixel de l'image avec la matrice :*

$$\begin{pmatrix} r' \\ g' \\ b' \end{pmatrix} = \begin{bmatrix} \frac{g_m}{r_m} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{g_m}{b_m} \end{bmatrix} \begin{pmatrix} r \\ g \\ b \end{pmatrix}$$

   de telle sorte que la couleur moyenne de l'image devienne gris. Vérifier que c'est bien ce que fait cette matrice.

3. *Complétez la fonction* `grayWorld` *de telle façon qu'elle renvoie les trois paramètres situés sur la diagonale de la matrice.*

4. *Testez sur les images dont le nom commence par* `wb_`*.*

▸ **Exercice 3.** ***Fond vert***

Le but de cet exercice est de retirer le fond vert d'une image pour le remplacer par une image de remplacement.

1. *Lisez le fichier* `tp2_pick_color.cpp`*, il sert à afficher les coordonnées en pixel d'un point cliqué sur une image. Ouvrez une image commençant par "*`fond_vert`*" et cliquez sur un point vert moyen (ni trop foncé, ni trop clair). Notez quelque part les coordonnées de ce pixel.*

2. *Ouvrez à présent le fichier* `tp2_exo3.cpp`*. Ce programme commence par convertir l'image à font vert dans l'espace colorimétrique* `Lab`*. Savez-vous pourquoi ? (cherchez sur le net)*

3. *Dans votre programme, identifiez la couleur cible en renseignant la position du pixel vert moyen de la question 1 dans la variable* `target_color_Lab`*.*

4. *À présent, pour chaque pixel de l'image à fond vert, cherchez les pixels dont la distance Euclidienne à la couleur cible est inférieure à 50. Dans ce cas, il s'agit d'un pixel de couleur similaire au vert ciblé qu'il faut effacer. Remplacer le pixel correspondant dans l'image oringinale (BGR) par un pixel blanc. Vérifiez que ce sont bien les pixels verts qui sont remplacés.*

5. *Au lieu de changer les pixel ciblés en blanc, remplacez les par la valeur du pixel associé sur l'image de remplacement. Pensez à utiliser un opérateur modulo sur cette image en cas de dépassement de coordonnées.*

▸ **Exercice 4. *Zoom***

En utilisant à nouveau le template du programme `tp02_ex03.cpp`*, faites une fonction permettant de faire un effet loop sur sur le point cliqué. Il s'agit d'un carré d'une vingtaine de pixels de coté qui vient se superposer à l'image (du moins visuellement) et affichant le contenu de l'image au point cliqué, avec un facteur de zoom* $\beta = 2$*.*