TP9 – Retouche d'images

La « retouche d'images » (*image editing*) regroupe un certain nombre d'applicaztions, dont l'*inpainting* que vous avez programmé dans le TP8. D'autres applications de retouche d'images sont illustrées dans l'article :

`https://www.cs.jhu.edu/~misha/Fall07/Papers/Perez03.pdf`

La première partie de ce TP vise à réaliser une application de *photomontage*. La seconde partie, qui est facultative, consiste à coder une ou plusieurs autres applications tirées de l'article ci-dessus.

Algorithme naïf de photomontage

Le script `exercice_0.m` demande à l'utilisateur de sélectionner un polygone P en cliquant dans une « image source » S (un double-clic sur le sommet cliqué en premier permet de terminer la saisie), et de le coller dans une deuxième image C, dite « image cible ». L'utilisateur doit ensuite sélectionner une « imagette » rectangulaire I dans C, en cliquant sur deux sommets opposés. La fonction `imresize` de Matlab permet de définir une transformation affine $T : S \to C$, telle que $T(E) = I$, où E désigne le rectangle englobant de P. L'image résultat R est égale à C, sauf pour les pixels $\mathbf{p} \in C$ tels que $T^{-1}(\mathbf{p}) \in P$, auquel cas $R(\mathbf{p}) = S(T^{-1}(\mathbf{p}))$.

Cet algorithme de photomontage est très naïf. On peut donc s'attendre à des résultats peu convaincants : sur l'image résultat R, on voit bien la trace du « collage ». Or, au lieu d'insérer une partie de l'image source S dans l'image cible C, comme le fait `exercice_0.m`, mieux vaut insérer une partie du gradient G_S de S dans le gradient G_C de C, et considérer qu'il s'agit du gradient G_R de l'image résultat R. Mais, s'il est facile de calculer de tels gradients (cf. TP8), le calcul de R à partir de G_R se fait par *intégration*.

Exercice 1 : intégration du gradient G_R (condition de Neumann)

Le calcul de l'image résultat R à partir du gradient G_R revient à résoudre un problème d'intégration. Un champ vectoriel $G_R : \mathbb{R}^2 \to \mathbb{R}^2$, tel que $G_R(x, y) = [G_x(x, y), G_y(x, y)]^\top$, n'est généralement pas intégrable, à moins que la *condition d'intégrabilité* $\frac{\partial G_x}{\partial y} = \frac{\partial G_y}{\partial x}$ soit vérifiée. La résolution de ce problème d'intégration peut néanmoins être effectuée, au sens des moindres carrés, en minimisant la fonctionnelle :

$$E(R) = \iint_{(x,y)\in I} \|\nabla R(x, y) - G_R(x, y)\|^2 \, dx \, dy \tag{1}$$

Dans (1), G_R constitue la donnée et R l'inconnue. L'équation d'Euler-Lagrange associée à ce problème d'optimisation est une *équation de Poisson* (cf. cours) :

$$\Delta R(x, y) = \nabla \cdot G_R(x, y) \tag{2}$$

Comme l'équation (2) est linéaire, sa discrétisation peut s'écrire sous forme matricielle :

$$\mathbf{A}\,\mathbf{u}_c = \mathbf{b}_c \tag{3}$$

où :

- La matrice $\mathbf{A}$ est une version discrète de l'opérateur laplacien.
- Le vecteur $\mathbf{u}_c$ contient les niveaux de couleur de l'image résultat R, pour les pixels situés à l'intérieur de l'imagette I, dans le canal c (rouge, vert ou bleu).
- Le vecteur $\mathbf{b}_c$ contient la divergence du gradient G_R dans le canal c.

Complétez le script `exercice_1.m` afin de mettre en œuvre ce nouvel algorithme de photomontage. Comme le rang de la matrice $\mathbf{A}$ n'est pas maximal : vous constatez que la résolution du système (3), par exemple à l'aide de l'opérateur \, provoque l'affichage d'un message d'avertissement ; libre à vous de choisir une contrainte permettant de définir la constante d'intégration (cf. cours).

Exercice 2 : intégration du gradient G_R (condition de Dirichlet)

L'algorithme de photomontage du script `exercice_1.m` est plus performant que celui de `exercice_0.m`, mais le « collage » reste visible. Pour gommer ce défaut résiduel, il convient de résoudre l'équation (3), dans chaque canal, en imposant une *condition de Dirichlet*. Cela signifie que, pour chaque pixel $\mathbf{p}$ situé sur le bord de l'imagette I, la condition au bord « de type Dirichlet » $R(\mathbf{p}) = C(\mathbf{p})$ doit être imposée. Une façon très simple d'imposer cette condition au bord consiste à modifier les lignes de la matrice $\mathbf{A}$ et du vecteur $\mathbf{b}_c$ qui correspondent aux pixels $\mathbf{p}$ du bord de l'imagette I (cf. cours).

Faites une copie du script `exercice_1.m`, de nom `exercice_2.m`, que vous modifierez de façon à imposer cette condition au bord. La modification de la matrice $\mathbf{A}$ rend son rang maximal. Cela a deux conséquences : les messages d'avertissement doivent disparaître ; le calcul de la constante d'intégration n'est plus nécessaire.

Une fois le script `exercice_2.m` mis au point, vous pourrez le relancer en inversant les rôles de l'image source et de l'image cible, puis en le testant sur n'importe quelle paire d'images de votre choix.

Exercice 3 : autres applications de retouche d'images (facultatif)

Vous êtes libres de coder une ou plusieurs applications de retouche d'images, parmi celles qui sont décrites dans l'article cité plus haut.