MINES Paris
http://moodle.psl.eu *2024-2025*

Ce mini-projet, à effectuer en binôme, fera l'objet d'un rapport incluant notamment équations et graphiques obtenus par des simulations sous Python. La forme de ce rapport est laissée libre (pdf, notebook, version papier...). Des fichiers python avec le prototype des différentes fonctions à implémenter sont disponibles pour le mini-projet. Le rendu s'effectuera sur Moodle. La date limite de rendu est le 25 avril 2025, aucun rendu ne sera possible après cela.

Amplification de mouvements dans des vidéos

L'objectif de ce mini-projet est d'étudier et d'implémenter un algorithme qui permet de visualiser des changements et des mouvements subtils dans des vidéos [1]. La méthode amplifie ces mouvements afin de les rendre plus perceptibles, et permet par exemple de mesurer le pouls d'une personne à partir d'une simple vidéo.

1 Filtrage temporel

Dans cette section, nous considérons qu'une vidéo peut être représentée par un signal continu $x, t \to I(x,t)$ associant à chaque position spatiale $x \in \mathbb{R}^2$ et chaque temps $t \in \mathbb{R}_+$ une couleur $I(x,t) \in \mathbb{R}^3$. Faisons l'hypothèse que la séquence vidéo $I(x,t)$ ne contient que des objets qui se translatent à vitesses identiques au cours de la vidéo. D'un point de vue mathématique, cela implique qu'il existe une fonction de déplacement $\delta : \mathbb{R} \to \mathbb{R}^2$ telle que

$$\forall x \in \mathbb{R}^2, \forall t \in \mathbb{R}, \quad I(x,t) = f(x + \delta(t)), \tag{1}$$

où $f(x) = I(x,0)$ pour tout $x \in \mathbb{R}^2$. Notre objectif est de synthétiser un signal qui amplifie les déplacements sur une plage de fréquences donnée. En pratique, on cherche donc à synthétiser le signal vidéo

$$S(x,t) = f(x + \delta(t) + \alpha h \star \delta(t)), \tag{2}$$

où l'opération de filtrage $h \star \delta$ permet d'isoler les déplacements correspondant à la plage de fréquences spécifiée. En effectuant un développement de Taylor de l'expression (2), on vérifie que

$$S(x,t) \simeq f(x) + \big(\delta(t) + \alpha h \star \delta(t)\big)\frac{\partial f}{\partial x}(x). \tag{3}$$

Par ailleurs, on vérifie de même que :

$$I(x,t) \simeq f(x) + \delta(t)\frac{\partial f}{\partial x}(x),$$

de sorte que

$$h \star I(x,t) \simeq h \star \delta(t) \frac{\partial f}{\partial x}(x).$$

En pratique, on applique donc les étapes suivantes pour synthétiser le signal :

(i) *Filtrage temporel* : la vidéo est filtrée dans le temps pour isoler les bandes de fréquence d'intérêt.

(ii) *Amplification* : Le signal filtré est amplifié pour augmenter la visibilité des changements.

(iii) *Reconstruction* : Le signal amplifié est réintégré à la vidéo originale pour produire le résultat final, où les changements subtils sont plus prononcés.

Question 1: Implémenter l'opération de filtrage temporel dans la fonction correspondante du fichier *algorithms*. On utilisera pour le filtrage un filtre passe-bande idéal pour lequel on spécifiera les fréquences de coupure $f_{\min}$ et $f_{\max}$. Appliquer l'approche à la vidéo proposée en utilisant la fonction *evm* du fichier *script.py*. On cherchera sur cet exemple à amplifier le signal de pouls de la personne filmée. Commenter le résultat obtenu. □

Question 2: Exprimer les relations (2) et 3) dans le domaine de Fourier pour la variable spatiale x. En déduire que le facteur d'amplification α doit vérifier

$$(1+\alpha)\delta(t) < \frac{\lambda}{8}, \tag{4}$$

où λ est la longueur d'onde spatiale déduite de la pulsation ω :

$$\lambda = \frac{2\pi}{\omega}.$$

Pour arriver à ce résultat, on supposera que l'approximation des fonctions trigonométriques aux petits angles est valide pour des angles inférieurs à $\frac{\pi}{4}$. □

2 Approche multi-échelle

L'expression (4) fixe une borne supérieure à la valeur que peut prendre α en fonction des longueurs d'onde présentes dans l'image, c'est à dire des échelles de variation spatiales caractéristiques de l'image. En particulier, pour que l'inégalité (4) soit satisfaite, on remarque qu'il est préférable d'appliquer l'opération d'amplification à une version de l'image filtrée spatialement i.e où on a éliminé les principales discontinuités spatiales. Pour filtrer l'image, une solution consiste à construire une approximation de l'image à partir d'une pyramide gaussienne. Le principe des pyramides gaussiennes repose sur l'application itérative des deux opérations suivantes :

(i) *Filtrage* : L'image est d'abord lissée à l'aide d'un filtre gaussien. Ce filtre atténue les détails fins et le bruit en appliquant une moyenne pondérée autour de chaque pixel. Mathématiquement, cela s'exprime par un produit de convolution entre l'image et une fonction gaussienne :

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}},$$

où σ contrôle l'intensité du lissage.

(ii) *Sous-échantillonnage* : l'image est ensuite sous-échantillonnée en supprimant des lignes et des colonnes. Par exemple, on peut diviser la résolution par deux en ne gardant qu'un pixel sur deux. Cela crée une version plus petite et moins détaillée de l'image originale.

En répétant ces deux étapes (lissage gaussien et sous-échantillonnage) plusieurs fois, on obtient une séquence d'approximations de l'image d'origine avec des résolutions de plus en plus faibles.

FIGURE 1 – Exemple de pyramide gaussienne.

La reconstruction d'une image haute résolution à partir d'une image de plus basse résolution se fait en deux étapes principales :

(i) *Sur-échantillonnage* : Le sur-échantillonnage consiste à augmenter la taille de l'image en ajoutant des pixels. Par exemple, si l'image est réduite de moitié à chaque niveau de la pyramide, on double sa taille en insérant des lignes et des colonnes de pixels dont la valeur est fixée à 0.

(ii) *Interpolation* : Après le sur-échantillonnage, les nouveaux pixels doivent être remplis avec des valeurs appropriées. Pour ce faire, on applique un filtre gaussien à l'image afin d'estimer les valeurs des pixels manquants en se basant sur les pixels voisins.

Pour reconstruire une image de même taille que l'image d'origine à partir de l'image de plus basse résolution de la pyramide, on itère les deux étapes précédentes jusqu'à remonter l'intégralité de la pyramide.

Question 3: Implémenter les opérations d'approximation et de reconstruction des images de la séquence vidéo dans les fonctions correspondantes du fichier *algorithms*. □

Pour synthétiser le signal à partir d'une pyramide gaussienne, on procède de la manière suivante :

(i) *Filtrage spatial* : on utilise une pyramide gaussienne pour obtenir une approximation à basse résolution de chaque frame de la vidéo d'origine.

(ii) *Filtrage temporel* : la vidéo à basse résolution spatiale est filtrée dans le temps pour isoler les bandes de fréquence d'intérêt.

(iii) *Reconstruction* : on reconstruit une version haute résolution de la vidéo filtrée.

(iv) *Amplification* : le signal filtré est amplifié pour augmenter la visibilité des changements.

(v) *Intégration* : Le signal amplifié est réintégré à la vidéo originale pour produire le résultat final, où les changements subtils sont plus prononcés.

Question 4: Implémenter l'algorithme d'amplification eulérienne basé sur les pyramides gaussiennes dans la fonction. Empiriquement, à quelle résolution faut-il réduire la vidéo pour garantir une bonne qualité visuelle ? □

Références

[1] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM transactions on graphics (TOG)*, 31(4) :1–8, 2012.