

Conception, implémentation et analyse d'un framework d'édition interactive d'images gigapixel.

Frédéric van der Essen
Promoteurs: Philip Dutré, Marc Lobelle.

5 septembre 2010

Qu'est ce qu'une image Giga-Pixel ?

Qu'est ce qu'une image Giga-Pixel ?

- Une image de plus d'un milliard de pixels indépendants

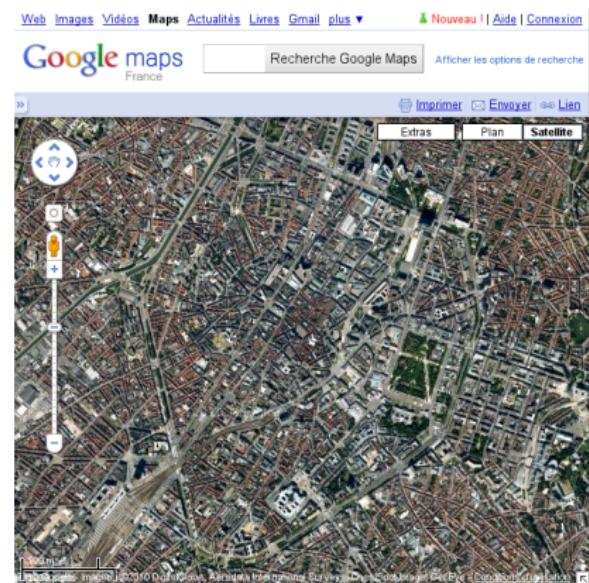
Qu'est ce qu'une image Giga-Pixel ?

- Une image de plus d'un milliard de pixels indépendants
- Une image trop grande pour tenir entièrement dans la mémoire vive.

Applications des images Giga-Pixel

Applications des images Giga-Pixel

■ La cartographie



Applications des images Giga-Pixel

- La cartographie
- La numérisation d'oeuvres d'art



Applications des images Giga-Pixel

- La cartographie
- La numérisation d'oeuvres d'art
- La 3D photoréaliste



Applications des images Giga-Pixel

- La cartographie
- La numérisation d'oeuvres d'art
- La 3D photoréaliste
- Les jeux vidéos.



Images Giga-Pixel : L'état de l'art

Images Giga-Pixel : L'état de l'art

- On sait acquérir des images giga-pixel

Images Giga-Pixel : L'état de l'art

- On sait acquérir des images giga-pixel
- On sait afficher des images giga-pixel

Images Giga-Pixel : L'état de l'art

- On sait acquérir des images giga-pixel
- On sait afficher des images giga-pixel
- On sait stocker des images giga-pixel

Images Giga-Pixel : L'état de l'art

- On sait acquérir des images giga-pixel
- On sait afficher des images giga-pixel
- On sait stocker des images giga-pixel
- Les possibilités d'éditions sont très limitées.

Composition d'un framework

Composition d'un framework

- Une structure de représentation de l'image.

Composition d'un framework

- Une structure de représentation de l'image.
- Des algorithmes d'édition de cette structure.

Composition d'un framework

- Une structure de représentation de l'image.
- Des algorithmes d'édition de cette structure.
- Un algorithme de rasterisation.

Frameworks d'édition d'images

Il existe trois types de frameworks d'éditions d'images :

- Frameworks Bitmaps.
- Frameworks Vectoriels.
- Frameworks Nodaux.

Frameworks Bitmaps

- Représentation de l'image sous forme rasterisée.
- La rasterisation est faite à chaque modification de l'image.

Frameworks Bitmaps

- Représentation de l'image sous forme rasterisée.
- La rasterisation est faite à chaque modification de l'image.
 - $O(S)$, S est le nombre de pixels affectés par l'opération.

Frameworks Bitmaps

- Représentation de l'image sous forme rasterisée.
- La rasterisation est faite à chaque modification de l'image.
 - $O(S)$, S est le nombre de pixels affectés par l'opération.
- Les modifications ne sont pas retenues.

Avantages et Inconvénients

■ Avantages :

- Grand nombre d'opérations.
- Édition pixel par pixel.
- Bonne interactivité à faible résolution.

Avantages et Inconvénients

- Avantages :

- Grand nombre d'opérations.
- Édition pixel par pixel.
- Bonne interactivité à faible résolution.

- Inconvénients :

- Résolution limitée : \leq 100 Méga-Pixels
- Pas d'édition non destructive.

└ Frameworks d'édition d'images

 └ Frameworks Bitmaps

Usages

- Peinture
- Retouche Photo

Usages

- Peinture
- Retouche Photo
- *Adobe Photoshop, The Gimp, etc.*

Frameworks Vectoriels

- Le dessin est composé de primitives
- Ces primitives sont représentées par un scene graph.
- La rasterisation est faite à chaque visualisation de l'image

Frameworks Vectoriels

- Le dessin est composé de primitives
- Ces primitives sont représentées par un scene graph.
- La rasterisation est faite à chaque visualisation de l'image
 - $O(s * n)$, s la taille en pixel de la région à visualiser, n le nombre de primitives affectant la région.

Avantages et Inconvénients

■ Avantages :

- Indépendance à la résolution.
- Édition non destructive.
- Description compacte de l'image.

Avantages et Inconvénients

- Avantages :

- Indépendance à la résolution.
- Édition non destructive.
- Description compacte de l'image.

- Inconvénients :

- Pas d'édition pixel par pixel
- Nombre limité de primitives.

Usages

- Impression
- Documents structurés
- Interfaces graphiques

Usages

- Impression
- Documents structurés
- Interfaces graphiques
- *Adobe Illustrator, Inkscape, Navigateur Webs, Cette présentation !*

Frameworks Nodaux

- Représentation de l'image sous forme de graphe de modifications
- La rasterisation est faite à chaque visualisation de l'image.

Avantages et Inconvénients

■ Avantages :

- Édition non destructive.
- Édition non linéaire.
- Traitement de masse.
- Indépendance à la résolution.
- Combinaison avec d'autres frameworks.

Avantages et Inconvénients

- Avantages :

- Édition non destructive.
- Édition non linéaire.
- Traitement de masse.
- Indépendance à la résolution.
- Combinaison avec d'autres frameworks.

- Inconvénients :

- Nombre limité d'opérations.
- Algorithmes de modifications complexes à gérer.

└ Frameworks d'édition d'images

└ Frameworks Nodaux

Usages

- Post-Production Vidéo
- Retouche Photo

Usages

- Post-Production Vidéo
- Retouche Photo
- *Apple Shake, Blender 3D, The Gimp*

Himalaya

Nouveau type de framework d'édition d'images.

Himalaya

Nouveau type de framework d'édition d'images.

- Inspiration :
 - Framework Nodaux.
 - MegaTexturing

Himalaya

Nouveau type de framework d'édition d'images.

- Inspiration :
 - Framework Nodaux.
 - MegaTexturing
- Avantages :
 - Édition non destructive.
 - Édition non linéaire.
 - Édition pixel par pixel.
 - Grand nombre d'opérations.
 - Bonne interactivité à haute résolution.

Himalaya

Nouveau type de framework d'édition d'images.

- Inspiration :

- Framework Nodaux.
- MegaTexturing

- Avantages :

- Édition non destructive.
- Édition non linéaire.
- Édition pixel par pixel.
- Grand nombre d'opérations.
- Bonne interactivité à haute résolution.

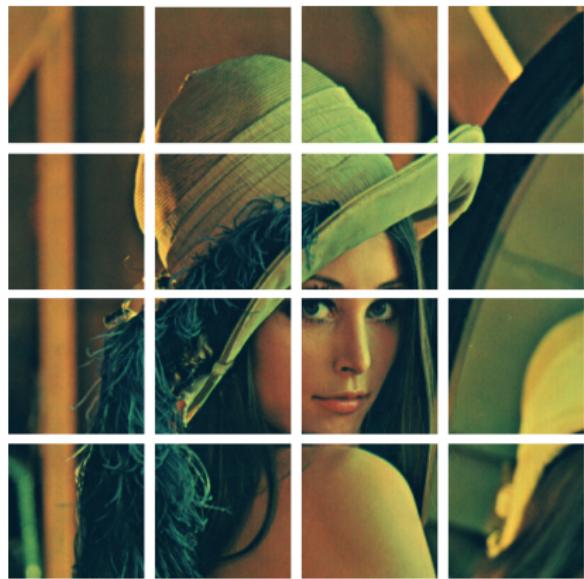
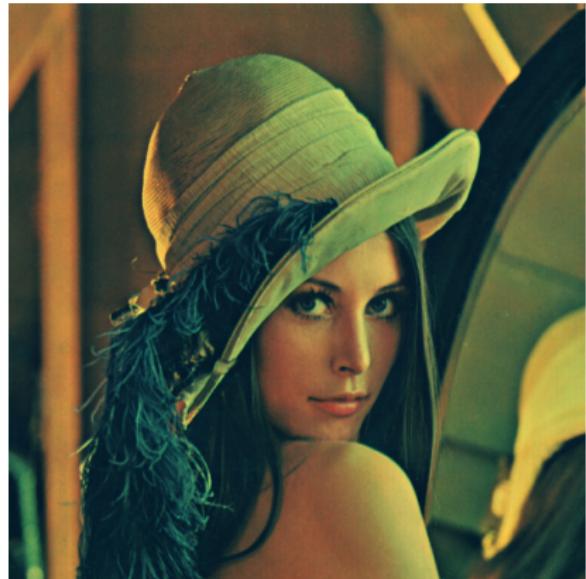
- Inconvénients :

- Filtres spatiaux.

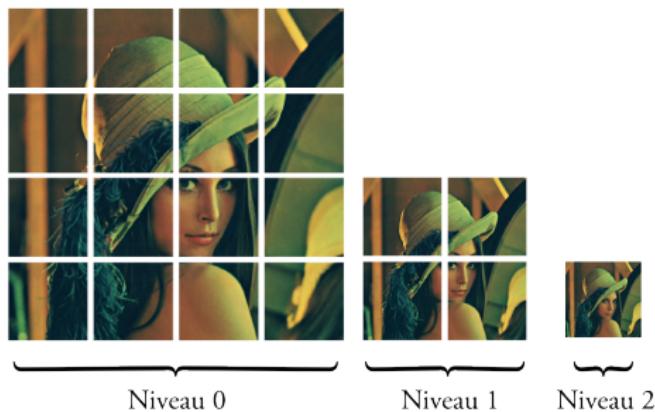
Tiles



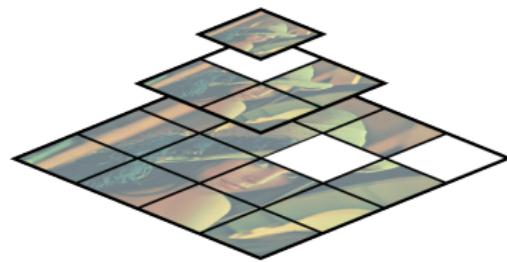
Tiles



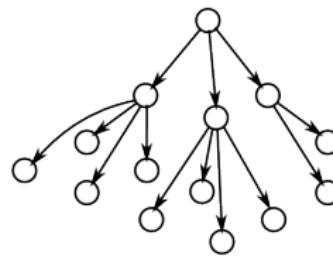
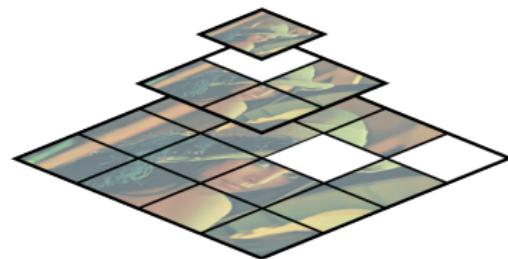
MipMaps



Pyramide de tile creuse



Pyramide de tile creuse



hlFrame : Propriétés

hlFrame : Propriétés

- Accès, insertion et suppression de tiles en $O(1)$

hlFrame : Propriétés

- Accès, insertion et suppression de tiles en $O(1)$
- La profondeur de l'arbre reste optimale.

hlFrame : Propriétés

- Accès, insertion et suppression de tiles en $O(1)$
- La profondeur de l'arbre reste optimale.
- Pas de taille maximale.

hlFrame : Propriétés

- Accès, insertion et suppression de tiles en $O(1)$
- La profondeur de l'arbre reste optimale.
- Pas de taille maximale.
- Les régions de couleur unie ne consomment pas de mémoire.

hlFrame : Utilité

hlFrame : Utilité

- Stocker des bitmaps

hlFrame : Utilité

- Stocker des bitmaps
- Servir de cache aux opérations.

Images

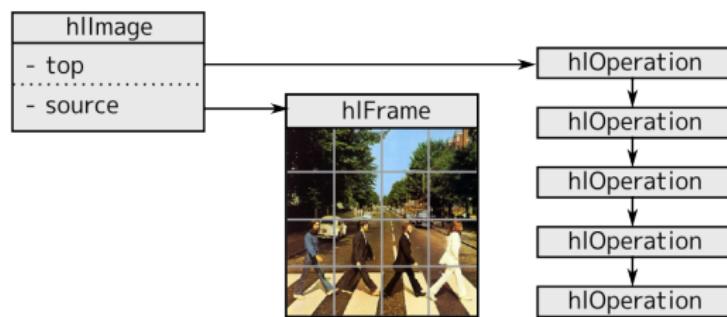
Une image dans Himalaya c'est :

- Un bitmap
Source
- Une pile
d'opérations
qui vont
modifier ce
bitmap.

Images

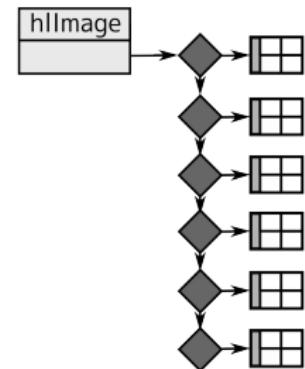
Une image dans Himalaya c'est :

- Un bitmap
Source
- Une pile
d'opérations
qui vont
modifier ce
bitmap.



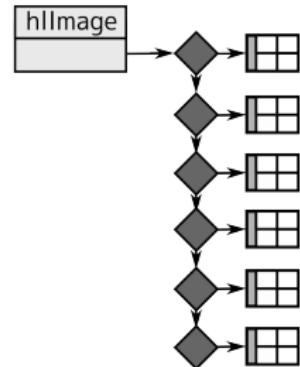
Rasterisation

Rasterisation



Rasterisation

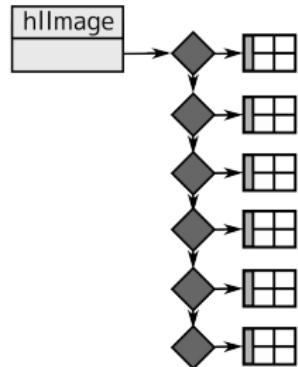
hTile *Rasterise(operation, tile_coord)* :



Rasterisation

hTile *Rasterise(operation, tile_coord)* :

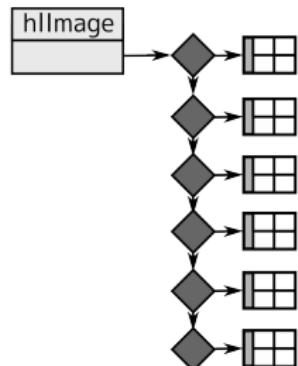
- 1 Tile en Cache ? Renvoyer le tile



Rasterisation

hTile *Rasterise(operation, tile_coord)* :

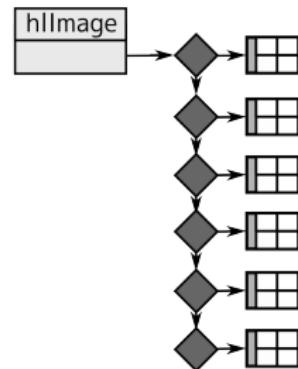
- 1 Tile en Cache ? Renvoyer le tile
- 3 hTile T =
Rasterise(operation.down,tile_coord)



Rasterisation

hTile *Rasterise(operation, tile_coord)* :

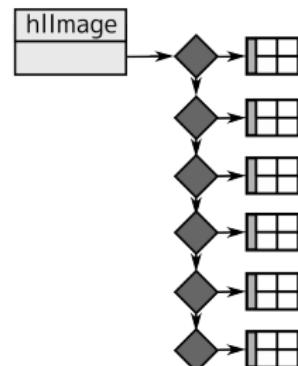
- 1 Tile en Cache ? Renvoyer le tile
- 3 hITile T =
Rasterise(operation.down,tile_coord)
- 4 *Draw(operation,T)*



Rasterisation

hTile *Rasterise(operation, tile_coord)* :

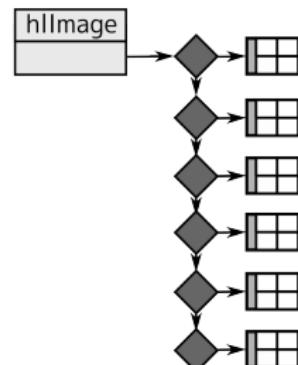
- 1 Tile en Cache ? Renvoyer le tile
- 3 hITile T =
Rasterise(operation.down,tile_coord)
- 4 *Draw(operation,T)*
- 5 On met T en cache



Rasterisation

hTile *Rasterise(operation, tile_coord)* :

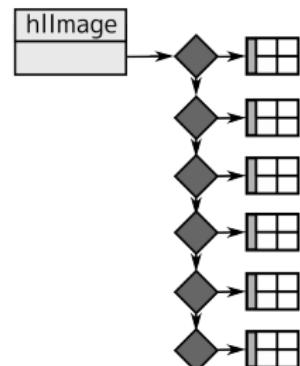
- 1 Tile en Cache ? Renvoyer le tile
- 3 hITile T =
Rasterise(operation.down,tile_coord)
- 4 Draw(operation,T)
- 5 On met T en cache
- 6 On renvoie T



Rasterisation

hTile *Rasterise(operation, tile_coord)* :

- 1 Tile en Cache ? Renvoyer le tile
- 2 Opération opaque ? Aller en 4
- 3 hITile T =
Rasterise(operation.down,tile_coord)
- 4 *Draw(operation,T)*
- 5 On met T en cache
- 6 On renvoie T



Propriété de la Rasterisation

- Complexité temporelle : $O(n)$ où n est le nombre d'opérations ajoutées depuis la dernière rasterisation du tile.
- Rasteriser une région : $O(s * n)$ où s est le nombre de tiles dans la région.

Ne pas mettre en cache à chaque opération

- Ne cacher qu'à certaines opérations.
- Enlever de la cache avant le retour lorsque l'opération ne doit plus cacher.

États

- Permettre à l'utilisateur d'accéder à différentes versions de son image.