情報デザイン専攻

画像情報処理論及び演習I

-画像合成-
Image Analogy

第10回講義
水曜日１限
教室6218

吉澤　信
shin@riken.jp, 非常勤講師
大妻女子大学　社会情報学部
独立行政法人 理化学研究所

今日の授業内容

www.riken.jp/brict/Yoshizawa/Lectures/index.html
www.riken.jp/brict/Yoshizawa/Lectures/Lec07.pdf
ごめんなさい.m(_ _)m、講義ページからリンク張り忘れました. 直接URL↑をfirefoxに打ち込んでください.

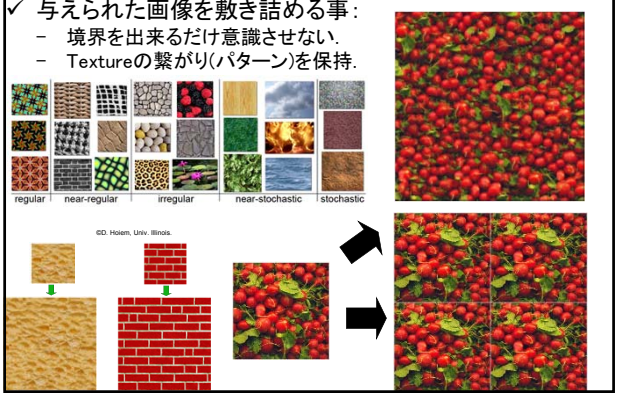
今日は演習メインでやります.

① 前回の復習.
② 演習：Image Analogy
＋レポート第3回の質問.

今日の演習は第4回レポートの内容なので
頑張ってくださいp(^^)q

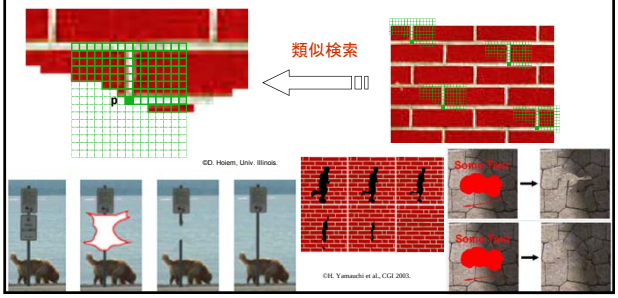
前回の復習：Texture合成

✓ 与えられた画像を敷き詰める事：
- 境界を出来るだけ意識させない.
- Textureの繋がり(パターン)を保持.



前回の復習：Pixel TransferによるInpainting

✓ 画像から似ている画素・Textureを持ってくる.
- 局所Windowで類似パターンを検索：Windowサイズに依存.
- 低周波画像は補間で生成しておくと影等の効果を反映出来る.
- 穴(マスク)を埋める順番が重要！

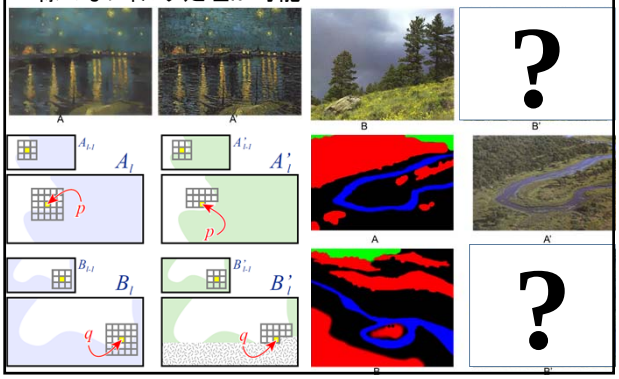


前回の復習：Image Analogy / Example-based

✓ 例題や類推による画像編集・合成:原理は同じ類似検索.

前回の復習：Image Analogy

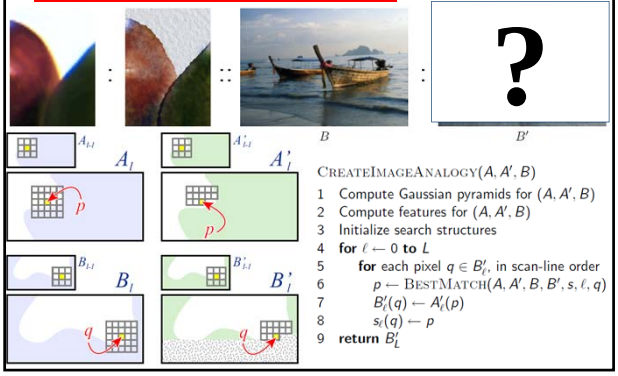
✓ 様々なフィルタ処理が可能！



Shin Yoshizawa: shin@riken.jp

Image Analogyアルゴリズム

パラメータ(Windowサイズ): r>=2.



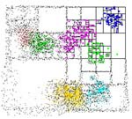
CreateImageAnalogy(A, A', B)
1 Compute Gaussian pyramids for (A, A', B)
2 Compute features for (A, A', B)
3 Initialize search structures
4 for ℓ ← 0 to L
5 for each pixel q ∈ B'ℓ, in scan-line order
6 p ← BestMatch(A, A', B, B', s, ℓ, q)
7 B'ℓ(q) ← A'ℓ(p)
8 sℓ(q) ← p
9 return B'L

Shin Yoshizawa: shin@riken.jp

Image Analogyアルゴリズム2

- ✓ 検索はANN (Approximate Nearest Neighbor)ライブラリを使う.
- ✓ ANNはエラー(誤差)を許して高速にn次元空間の近傍をサーチ.

パラメータ(ANNError): E>=1.0.



CreateImageAnalogy(A, A', B)
1 Compute Gaussian pyramids for (A, A', B)
2 Compute features for (A, A', B)
3 Initialize search structures
4 for ℓ ← 0 to L
5 for each pixel q ∈ B'ℓ, in scan-line order
6 p ← BestMatch(A, A', B, B', s, ℓ, q)
7 B'ℓ(q) ← A'ℓ(p)
8 sℓ(q) ← p
9 return B'L

Shin Yoshizawa: shin@riken.jp

Image Analogyアルゴリズム3

- ✓ Best Approximate Matchは Windowの半径2のとき55次元ベクトルのガウス相関.
  - ガウス相関: 中心の画素からガウス関数で重みを付けて対応する画素を要素とするベクトルの距離.
- ✓ Best Coherence Matchは Textureの整合性を加味して既に合成された画素の対応する画素でサーチ.
  - Textureの整合性を重視する場合はパラメータkを大きくする.
  - 大きくしすぎるとAとA'だけしか結果に反映されないので注意.

パラメータ(Texture度): k>=0.

BestMatch(A, A', B, B', s, ℓ, q)
1 $p_{app} \leftarrow$ BestApproxMatch(A, A', B, N', ℓ, q)
2 $p_{coh} \leftarrow$ BestCoherenceMatch(A, A', B, N', s, ℓ, q)
3 $d_{app} \leftarrow \|F_\ell(p_{app}) - F_\ell(q)\|^2$
4 $d_{coh} \leftarrow \|F_\ell(p_{coh}) - F_\ell(q)\|^2$
5 if $d_{coh} \le d_{app}(1 + 2^{\ell - L}k)$
6 then return p_{coh}
7 else return p_{app}

Shin Yoshizawa: shin@riken.jp

演習:Image Analogyを使ってみよう!

www.riken.jp/brict/Yoshizawa/Lectures/Ex05.zip
www.riken.jp/brict/Yoshizawa/Lectures/Lec07.pdf

ごめんなさい.m(_ _)m、講義ページからリンク張り忘れました.
直接URL↑をfirefoxに打ち込んでください.

Image Analogyでフィルタリング:

1. Ex05内に用意されたプログラム群を動かしてみる.
2. Ex05内の画像を用いてImage Analogyによる色々なフィルタリング処理をしてみる.
3. 新しいフィルタリングを考えてみよう!

今日の演習は第4回レポートの内容なので
頑張ってくださいねーp(^^)q

Shin Yoshizawa: shin@riken.jp

演習:ANNのコンパイル

www.riken.jp/brict/Yoshizawa/Lectures/Ex05.zip
www.riken.jp/brict/Yoshizawa/Lectures/Lec07.pdf

まずはじめに、ANNをコンパイルする.

1. Ex05.zipを展開する.
2. Ex05内にann_1.1.2.zipがあるのでEx05内で展開する.
3. 端末でEx05/ann_1.1.2に入る、もしもデスクトップに展開していたら、「cd ~/Desktop/Ex05/ann_1.1.2」.
4. コンフィギュレーションを行う4.の後に端末で「sh Make-config」でエンターキー.
5. コンパイルする5.の後に端末で「make linux-g++」と打ち込みエンターキーを押す.Ex05/ann_1.1.2/libの下にlibANN.aが出来れば成功.

Shin Yoshizawa: shin@riken.jp

演習:Ex05内の説明

www.riken.jp/brict/Yoshizawa/Lectures/Ex05.zip
www.riken.jp/brict/Yoshizawa/Lectures/Lec07.pdf

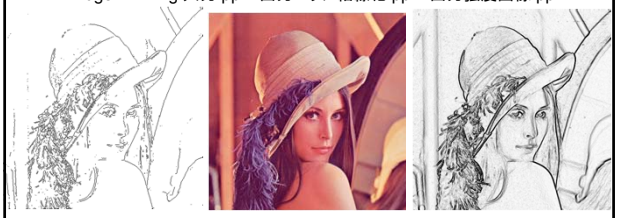
Ex05内の説明:コンパイルは端末で「make」Makefile

- ✓ ImageAnalogyClass.h: Image Analogyの本体.
  - ColorImage.h: カラー画像クラス.
  - GaussianPyramid.h: ガウスピラミッドクラス.
- ✓ Image Analogyとは関係ないファイル:
  - Image Analogyの入力画像を生成するフィルタで使うヘッダーファイル:Gauss.h: ガウス平滑化用、fastgb.h & gaussfgt1D.h:高速エッジ保存フィルター用.
  - 前回までに使ったファイル:SimpleImage.h(画像クラス)、otsu.h(大津の二値化)、ppmio.h(カラー画像入出力)、thinning.h(細線化).

Shin Yoshizawa: shin@riken.jp

演習：Image Analogyとは関係ないファイル

- ✓ まずは、Image Analogyとは直接関係ないプログラムから. ただし、これらのプログラムを使えばImage Analogyに入力させる画像を簡単に作成可能: Run_Smoothing.sh, Run_EdgePreserving.sh, Run_EdgeThinning.sh.
- ✓ EdgeThinning.cxx: エッジ強度画像(勾配強度＝Gradientベクトルの大きさ)とエッジの細線化画像を出力するプログラム:引数3:
  - EdgeThinning 入力.ppm 出力エッジ細線化.ppm 出力強度画像.ppm

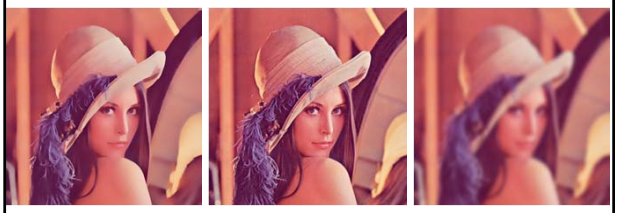


エッジ細線化画像　入力　エッジ強度画像

Shin Yoshizawa: shin@riken.jp

演習：Image Analogyとは関係ないファイル

- ✓ Smoothing.cxx:ガウス平滑化を実行するプログラム:引数3:
  - Smoothing 入力.ppm 出力.ppm 平滑化度合(double)
  - 平滑化度合のパラメータは0より大きな実数2.0～20.0ぐらいが実用的.
- ✓ EdgePreservingFilter.cxx: エッジ保存平滑化を実行: 引数3
  - EdgePreservingFilter 入力.ppm 出力.ppm エッジの大きさ(double)
  - エッジの大きさパラメータは0より大きな実数0.5～2.0ぐらいが実用的.

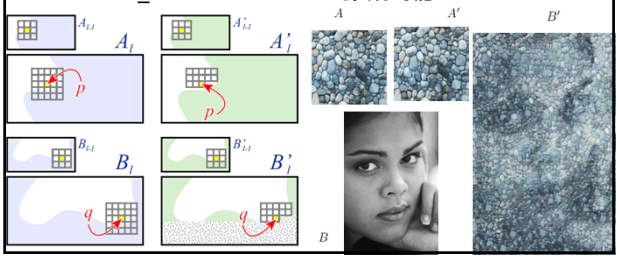


EdgePreservingFilter, 1.0　入力　Smoothing, 5.0

Shin Yoshizawa: shin@riken.jp

演習：Texture Transfer

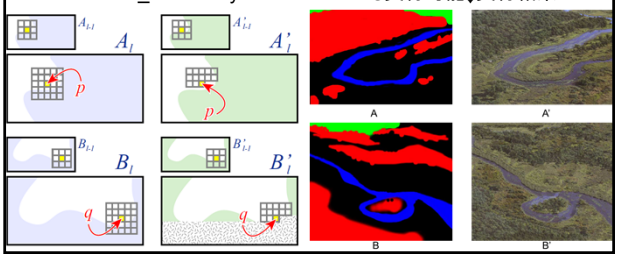
- ✓ Image Analogyを用いてTexture Transferを実行するプログラム: 引数 8
  - TextureTransfer 入力画像A.ppm 入力画像A'.ppm 入力画像B.ppm 出力画像B'.ppm Texture度k(double>=0.0) ANN誤差(double>=1.0) Window半径(int>=2) Blending(1.0>=double>=0.0, 小→元画像強め)
  - sh Run_TextureTransfer.shでも実行可能.



Shin Yoshizawa: shin@riken.jp

演習：Texture by Numbers

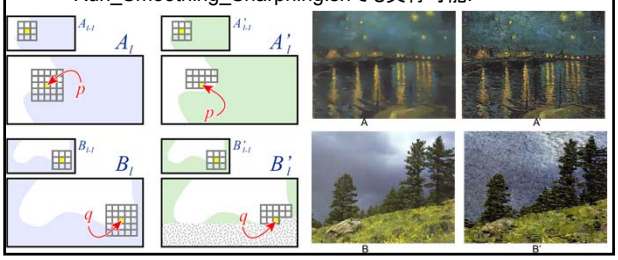
- ✓ Image Analogyを用いてTexture by Numbersを実行するプログラム: 引数 7
  - TextureByNumbers 入力画像A.ppm 入力画像A'.ppm 入力画像B.ppm 出力画像B'.ppm Texture度k(double>=0.0) ANN誤差(double>=1.0) Window半径(int>=2)
  - sh Run_TextureByNumbers.shでも実行可能↓実行結果.



Shin Yoshizawa: shin@riken.jp

演習：Artistic Filters

- ✓ Image Analogyを用いて様々なArtisticフィルタを実行: 引数 7
  - ArtisticFilter 入力画像A.ppm 入力画像A'.ppm 入力画像B.ppm 出力画像B'.ppm Texture度k(double>=0.0) ANN誤差(double>=1.0) Window半径(int>=2)
  - sh Run_ArtisticFilter.sh、sh Run_Etc1～3.sh、sh Run_Smoothing_Sharpning.shでも実行可能.



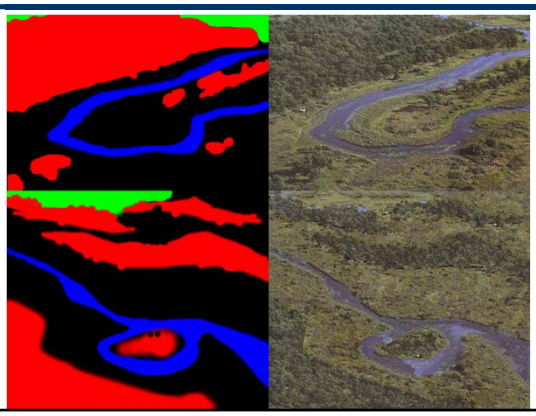
Shin Yoshizawa: shin@riken.jp

演習：シェルの説明

- ✓ 端末にて「sh シェルスクリプト名.sh」で実行、中にコンパイル＋実行＋表示のコマンドが書いてある.
  - Run_TextureTransfer.sh: 5種類のテクスチャーをテクスチャー度を1,3,5,7,9の五種類で実行.

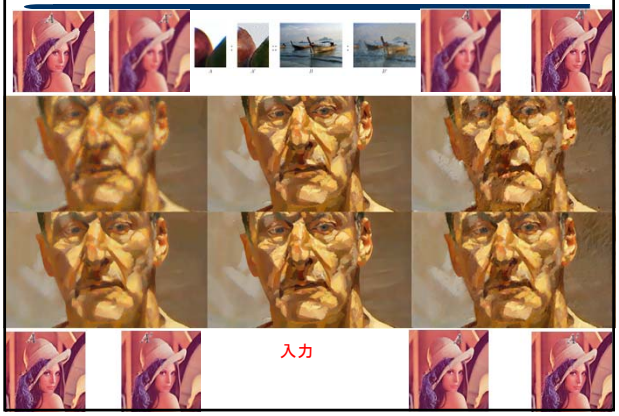
Shin Yoshizawa: shin@riken.jp

演習：Run_TextureByNumbers.sh



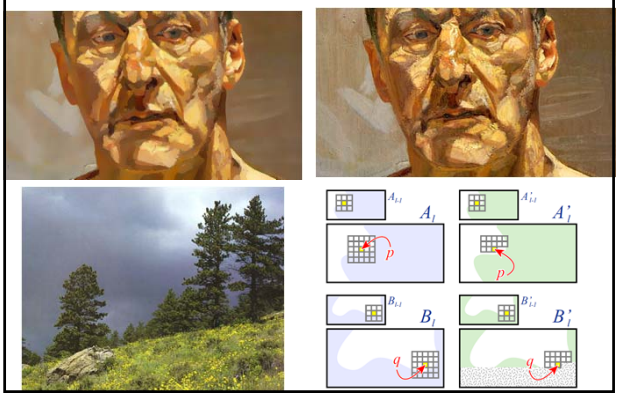
Shin Yoshizawa: shin@riken.jp

演習：Run_Smoothing_Sharpning.sh



Shin Yoshizawa: shin@riken.jp

演習：Run_ArtisticFilter.sh



Shin Yoshizawa: shin@riken.jp

演習：Run_ArtisticFilter.sh

- テクスチャー度の違い：6種類実行.

Shin Yoshizawa: shin@riken.jp

演習：Run_ArtisticFilter.sh

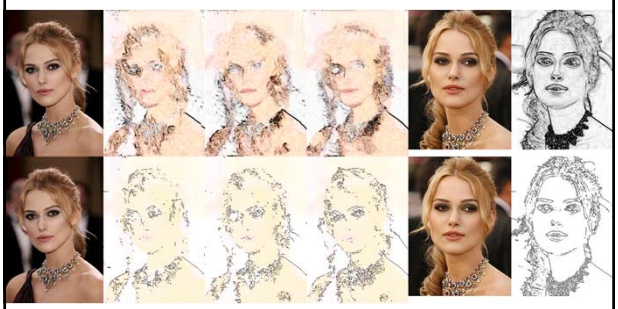
- Window半径の違い：2種類実行.

注意点：ANN誤差は全て1000.0で実行、1.0に近ければ綺麗な結果だが、計算時間が大: 数十分～数十時間かかる可能性あり!

Shin Yoshizawa: shin@riken.jp

演習：Run_Etc1.sh

- ArtisticFilterにEdgeThinningの出力(エッジ強度)を使った結果.



Shin Yoshizawa: shin@riken.jp

演習：Run_Etc2.sh

Shin Yoshizawa: shin@riken.jp

演習：Run_Etc3.sh

- ✓ 油絵的フィルタ効果

入力→

Shin Yoshizawa: shin@riken.jp

演習：Run_Etc3.sh

- ✓ 水彩画的フィルタ効果

入力→

Shin Yoshizawa: shin@riken.jp

演習：シェルスクリプトを動かしてみよう！

端末にて「sh シェルスクリプト名.sh」

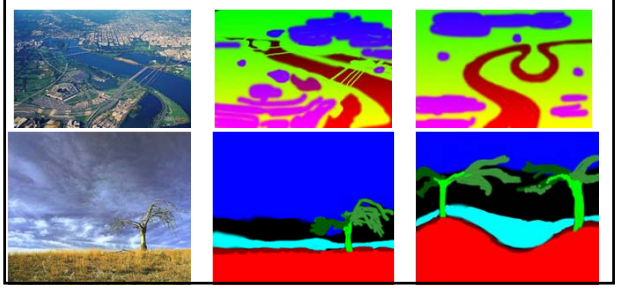
- ✓ Run_ArtisticFilter.sh
- ✓ Run_EdgePreserving.sh
- ✓ Run_EdgeThinning.sh
- ✓ Run_Smoothing.sh
- ✓ Run_Smoothing_Sharpning.sh
- ✓ Run_TextureTransfer.sh
- ✓ Run_TextureByNumbers.sh
- ✓ Run_Etc1.sh
- ✓ Run_Etc2.sh
- ✓ Run_Etc3.sh

Shin Yoshizawa: shin@riken.jp

演習：シェルスクリプトを変えてみよう！

Run_TextureByNumbers.shを使って以下の画像に対して処理してみよう!

A　　A’　　B

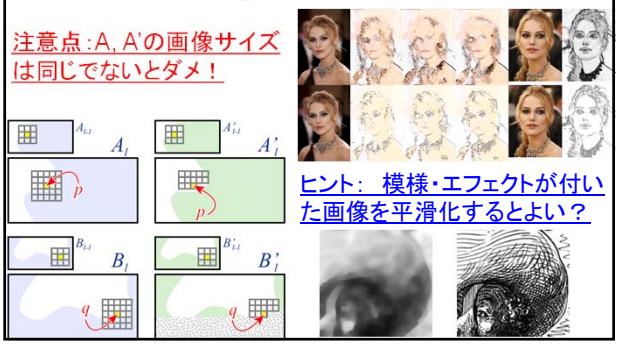


Shin Yoshizawa: shin@riken.jp

演習：自分で新しいエフェクトを作ってみよう！

Run_ArtisticFilter.shを使って以下の画像の様に自分のオリジナルのエフェクトを処理してみよう!

注意点：A, A’の画像サイズは同じでないとダメ！



ヒント：　模様・エフェクトが付いた画像を平滑化するとよい？

Shin Yoshizawa: shin@riken.jp

来週の予定

www.riken.jp/brict/Yoshizawa/Lectures/index.html

©Perez et al. SIGGRAPH 2003.

① 画像合成・Inpaintingその3

② 演習: 画像合成.