Random NL-Means to Restoration of Colour Images[1]

Dariusz Borkowski[a] Katarzyna Jańczak-Borkowska[b]

[a]Faculty of Mathematics and Computer Science
Nicolaus Copernicus University in Toruń, Poland

[b]Institute of Mathematics and Physics
University of Technology and Life Sciences in Bydgoszcz, Poland

[1]This research was supported by the National Science Centre (Poland) under decision DEC-2012/07/D/ST6/02534

The Inverse Problem for RGB Images

$u : \bar{D} \to \mathbb{R}^3$ – original colour image,
$u_0 : \bar{D} \to \mathbb{R}^3$ – noisy colour image,

$$u_0 = u + \eta,$$

η – Gaussian noise.

We are given u_0, the problem is to reconstruct u.

The Inverse Problem

- linear filtering
- wavelets theory
- variational mathods
- PDE-based methods
- stochastic modelling
- non local means approach

Non Local Means

Let $v = \{u_0(i) | i \in I = \mathbf{Z}^2 \cap D\}$ be a discrete noisy image

$$NL(v)(i) = \sum_{j \in I} w(N_i, N_j) v(j).$$

The weight $w(N_i, N_j)$ depends on the similarity of the intensity gray level or colour vectors of neighbourhoods N_i, N_j centred at pixels i and j

Algorithm for Non Local Means

Let $B_{i,F}$ be a window of size $F \times F$ centred at i

Buades et al. (2005)

$u(i) \leftarrow 0$; $Z \leftarrow 0$;
for $j \in B_{i,F}$ **do**
  $u(i) \leftarrow u(i) + \omega(i,j) \cdot v(j)$;
  $Z \leftarrow Z + \omega(i,j)$;
end for
$u(i) \leftarrow \frac{u(i)}{Z}$;

$$\omega(i,j) = \exp\left(-\frac{\max\left(\frac{\left\| B_{i,r} - B_{j,r} \right\|^2}{3(2r+1)} - 2\sigma^2, 0 \right)}{h^2} \right)$$

Buades et al. (2011)

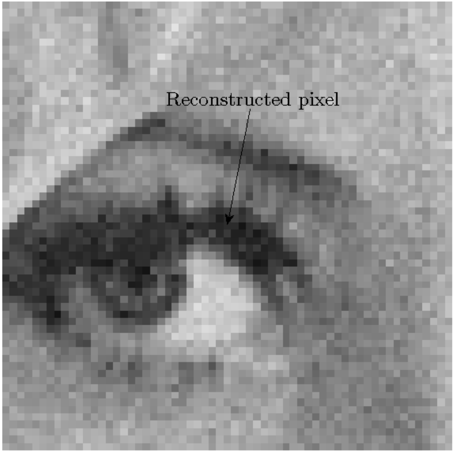


Figure : Position of the reconstructed pixel

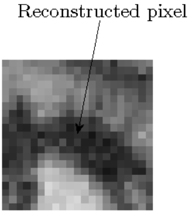


Figure : Square sub-image

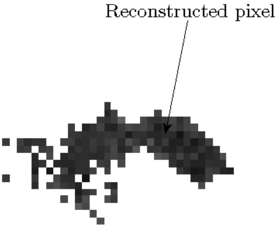


Figure : Random anisotropic sub-image

Algorithm for Random Non Local Means

$u(i) \leftarrow 0$; $X(0) \leftarrow i$; $Z \leftarrow 0$;
for $m = 1 \ldots M$ **do**
  $k \leftarrow 1$;
  while $k \leq K$ **do**
    $X(k) \leftarrow \Pi_D \left(X(k-1) + \sqrt{\sigma} \cdot (\Phi_{0,1}, \Phi_{0,1}) \right)$;
    if $\| G_\rho * v(X(k)) - G_\rho * v(X(k-1)) \| \leq \sigma$ **then**
      $u(i) \leftarrow u(i) + \omega\left(i, \lfloor X(k) \rceil\right) \cdot v(\lfloor X(k) \rceil)$;
      $Z \leftarrow Z + \omega\left(i, \lfloor X(k) \rceil\right)$;
      $k \leftarrow k + 1$;
    end if
  end while
end for
$u(i) \leftarrow \frac{u(i)}{Z}$;

Parameters

Noise	R	F	M	K	h
$0 < \sigma \leq 25$	3	21	49	9	$0.55\,\sigma$
$25 < \sigma \leq 55$	5	35	49	9	$0.40\,\sigma$
$55 < \sigma \leq 100$	7	35	49	9	$0.35\,\sigma$

Figure : Top-left: Noisy image $\rho = 10$, Top-right: NLM, Bottom-left: TV, Bottom-right: RaNLM

Table : Lenna

Noise	PSNR			SSIM		
	NLM	RaNLM	TV	NLM	RaNLM	TV
10	33.9930	**34.4845**	33.2549	0.9733	**0.9754**	0.9614
15	32.6303	**33.0483**	31.7692	0.9567	**0.9609**	0.9430
20	31.5809	**31.9674**	30.7777	0.9400	**0.9457**	0.9276
25	30.6771	**31.0525**	29.9996	0.9247	**0.9299**	0.9131
30	29.6546	**30.2775**	29.3863	0.9131	**0.9226**	0.9004
35	29.0823	**29.6591**	28.8702	0.8963	**0.9081**	0.8885
40	28.6100	**29.1131**	28.4306	0.8840	**0.8927**	0.8777
45	28.1913	**28.5782**	28.0531	0.8717	**0.8799**	0.8680
50	27.6853	**28.0981**	27.7119	0.8602	**0.8676**	0.8586
60	27.0863	**27.3478**	27.1375	0.8423	**0.8519**	0.8423
70	26.4170	26.6170	**26.6561**	0.8213	0.8272	**0.8278**
80	25.8092	25.8685	**26.2454**	0.8001	0.8014	**0.8150**
90	25.2978	25.3654	**25.8794**	0.7848	0.7838	**0.8030**
100	24.6402	24.7026	**25.5542**	0.7602	0.7575	**0.7922**

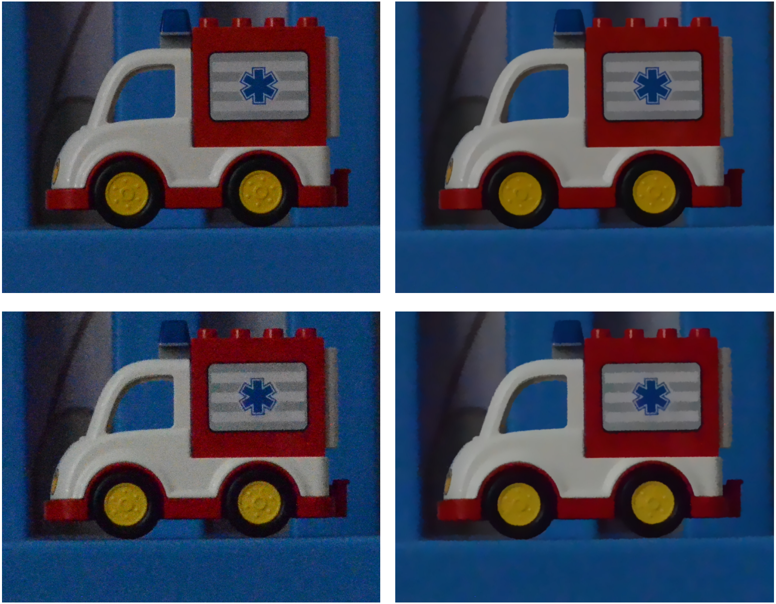
Table : Time of the reconstruction (in seconds). It has been tested for 512×512 and 256×256 RGB images, $4 \times$ CPU 2.5 GHz.

σ	Lenna 512×512		Peppers 512×512		Beans 256×256	
	NLM	RaNLM	NLM	RaNLM	NLM	RaNLM
10	**2.5**	16.5	**3.0**	18.5	**1.0**	6.0
15	**2.5**	14.0	**2.5**	15.5	**0.5**	5.0
20	**2.5**	11.0	**2.5**	12.5	**0.5**	4.0
25	**2.0**	10.0	**2.5**	11.5	**0.5**	3.5
30	**8.5**	10.5	**9.0**	12.0	**2.5**	3.0
35	**9.0**	10.5	**9.0**	11.5	**2.5**	3.5
40	**8.5**	10.0	**9.0**	11.0	**2.0**	3.0
45	**9.0**	10.0	**8.5**	10.5	**2.5**	3.0
50	**8.5**	9.5	**9.0**	10.5	**2.5**	3.0
60	16.5	**11.0**	14.5	**12.0**	3.5	**3.0**
70	14.5	**11.0**	14.0	**12.0**	3.5	**3.0**
80	15.5	**11.5**	14.5	**12.5**	3.5	**3.0**
90	15.0	**11.5**	14.5	**12.5**	4.0	**3.0**
100	15.5	**11.5**	14.0	**11.0**	**3.5**	**3.5**

Patchwise Implementation

$$\hat{B}_{i,R} = \frac{1}{Z} \sum_{j \in B_{i,F}} v(B_{j,R}) w(B_{i,R}, B_{j,R}) \qquad (NLM)$$

$$\hat{B}_{i,R} = \frac{1}{Z} \sum_{j \in X_{i,M,K}} v(B_{j,R}) w(B_{i,R}, B_{j,R}) \qquad (RaNLM)$$



Conclusion

- Our method combines the advantages of non local means and anisotropic diffusion.
- In the future work an interesting point is to apply this idea to
  - more complicated algorithms such as BM3D and NL-Bayes,
  - to another colour spaces.