ROBUST DENOISING OF 2D IMAGE USING ANN

J. Kukal, L. Hainc

Institute of Chemical Technology, Department of Computing and Control Engineering, Prague

Abstract

One of useful and popular local operations in local processing of 2D image is image de-noising with two contradictory aims: decreasing of the noise level and saving the structure of the original image. The signal to noise ratio (SNR) will increase in this case. Various types of ANN as OLAM, MLP, RBF can be used directly as a kind of sophisticated nonlinear filter on local pixel neighborhood (3x3), which is a little bit naive in general. Every intensity value from pixel neighborhood is passed to the adequate input neuron and the de-noised value is available on the single output of given ANN. Our article is oriented to more sophisticated local preprocessing which increases both the number of hidden layers of the hierarchical de-noising system and the learning abilities and the robustness of the proposed system.

1 Image Processing Preliminaries

Let $n_\mathrm{R}, n_\mathrm{C} \in \mathbf{N}$ be a number of rows and columns. Let $i \in \{1, \ldots, n_\mathrm{R}\}, j \in \{1, \ldots, n_\mathrm{C}\}$ be indices of given **pixel** $p_{i,j}$ of **intensity** $x_{i,j} \in [0;1]$. Then the **gray 2D image** is represented by the matrix $\mathbf{X} \in [0;1]^{n_\mathrm{R} \times n_\mathrm{C}}$. Let $r \in \mathbf{N}$ be a neighborhood size. Then the **neighborhood** of the pixel $p_{i,j}$ is defined as

$$\mathcal{N}_{i,j}^r = \{p_{k,l} \mid \ |k-i| \le r \wedge |l-j| \le r\}$$

and represented by the intensity values

$$\mathcal{I}_{i,j}^r = (x_{k,l} \mid \ |k-i| \le r \wedge |l-j| \le r)$$

where

$$1 + r \le i \le n_\mathrm{R} - r$$
$$1 + r \le j \le n_\mathrm{C} - r.$$

So, the list $\mathcal{I}_{i,j}^r$ consists of $n = (2r+1)^2$ values and can be also represented as the vector $\overline{x} = (x_1, \ldots, x_n) \in [0;1]^n$.

Let $y_{i,j} \in [0;1]$ be a de-noised value of $x_{i,j}$. The **local de-noising** is then represented by the mapping

$$y_{i,j} = f(\mathcal{I}_{i,j}^r) = f(\overline{x})$$

with the optimality condition

$$\mathrm{SSQ} = \sum_{i,j} (y_{i,j}^* - y_{i,j})^2 = \min$$

where $y_{i,j}^* \in [0;1]$ is given pixel intensity of an ideal image. In the special case of $r = 1$ the neighborhood consists of $n = 9$ pixels. The values of the pixel intensity are depicted on Figs 1 and 2.

$x_{i-1,j-1}$	$x_{i-1,j}$	$x_{i-1,j+1}$
$x_{i,j-1}$	$x_{i,j}$	$x_{i,j+1}$
$x_{i+1,j-1}$	$x_{i+1,j}$	$x_{i+1,j+1}$

Figure 1: $\mathcal{I}_{i,j}^1$ structure

x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9

Figure 2: Vector notation of $\mathcal{I}_{i,j}^1$

From the traditional point of view, there are simple de-noising filters represented by

$$\begin{aligned}
f(\overline{x}) &= \text{mean}(x_1, \ldots, x_9) \\
f(\overline{x}) &= \text{mean}(x_2, x_4, x_5, x_6, x_8) \\
f(\overline{x}) &= \text{median}(x_1, \ldots, x_9) \\
f(\overline{x}) &= \text{median}(x_2, x_4, x_5, x_6, x_8) \\
f(\overline{x}) &= \frac{x_5}{4} + \frac{x_2 + x_4 + x_6 + x_8}{8} + \frac{x_1 + x_3 + x_7 + x_9}{16}
\end{aligned}$$

and many other functions.

The second extreme approach based on the artificial neural networks is represented by general formula $f(\overline{x}) = \text{ANN}(\overline{x}, \overline{w})$ where $\overline{x} \in \mathbf{R}^n$, $\overline{w} \in \mathbf{R}^q$, $q \in \mathbf{N}$ is the number of weights of given ANN. The basic research in the area of image de-noising methods was performed over decades. That is why the direct learning of general ANN can hardly bring better results than the traditional de-noising. But the traditional principles can be used for both sophisticated preprocessing and postprocessing.

2 Statistical Preliminaries

The statistical sample of n values can be represented by various lists. Let $\mathcal{L} = (x_1, \ldots, x_n)$ be a **list of input values**. Let $\mathcal{O} = (x_{(1)}, \ldots, x_{(n)})$ consist of all values from the list $\mathcal{L}$. When $x_{(1)} \leq x_{(2)} \leq \ldots \leq x_{(n)}$, then $\mathcal{O}$ is called an **ordered list** of the values from $\mathcal{L}$. We can also define **Walsh list** of $n(n+1)/2$ values, which is generated from $\mathcal{L}$ by formula

$$\mathcal{W} = (\frac{x_i + x_j}{2} \mid 1 \leq i \leq j \leq n).$$

It is also useful to order the values from $\mathcal{W} = (\xi_1, \ldots, \xi_{n(n+1)/2})$ to the **ordered Walsh list** $\mathcal{W}^* = (\xi_{(1)}, \ldots, \xi_{(n(n+1)/2)})$.

Let $k \in \mathbf{N}_0$, $P = \lfloor \frac{n+1}{2} \rfloor$, $Q = \lceil \frac{n+1}{2} \rceil$, $R = \lfloor \frac{n}{4} \rfloor$, $S = \frac{n-k}{2}$, $T = \frac{n-k-1}{2} \in \mathbf{N}$. Then we can define useful functions for the processing of list $\mathcal{L}$:

- $\text{AVG}_k(\mathcal{L}) = \frac{1}{k} \sum_{j=1}^{k} x_{(S+j)}$
- $\text{MED}_k(\mathcal{L}) = \frac{1}{2}(x_{(P-k)} + x_{(Q+k)}),\ k < P$
- $\text{BIN}_k(\mathcal{L}) = \frac{1}{2^k} \sum_{j=0}^{k} \left(\begin{array}{c} k \\ j \end{array} \right) x_{T+j}$
- $\text{BES}(\mathcal{L}) = \frac{1}{2}(\text{MED}_0(\mathcal{L}) + \text{MED}_R(\mathcal{L}))$
- $\text{Q}_1(\mathcal{L}) = x_{P-R}$
- $\text{Q}_3(\mathcal{L}) = x_{Q+R}$
- $\text{L}(\mathcal{L}, \overline{w}) = \sum_{k=1}^{n} w_k x_{(k)}$
- $\text{FIR}(\mathcal{L}, \overline{w}) = \sum_{k=1}^{n} w_k x_k$
- $\text{HL}(\mathcal{L}) = \text{MED}_0(\mathcal{W})$
- $\text{WBIN}_k(\mathcal{L}) = \text{BIN}_k(\mathcal{W})$
- $\text{WBES}(\mathcal{L}) = \text{BES}(\mathcal{W})$
- $\text{WQ}_1(\mathcal{L}) = \text{Q}_1(\mathcal{W})$

- $\mathrm{WQ}_3(\mathcal{L}) = \mathrm{Q}_3(\mathcal{W})$
- $\mathrm{WL}(\mathcal{L}, \overline{w}) = L(\mathcal{W}, \overline{w})$

Here FIR represents general linear function, L represents general L-estimate (AVG, MED, BIN, BES, Q_1, Q_3 are special cases). Then, AVG_k is trimmed average, MED_0 is median, MED_k is quasi-median for $k > 0$, BIN_k is binomial L-estimate, BES is Turkey's best easy estimate, Q_1 is the first quartile, Q_3 is the third quartile, HL is Hodges-Lehman median and WBIN, WBES, WQ_1, WQ_3, WL are previous estimates applied to Walsh list. Except of the FIR function, the other statistical estimates are robust. It means, they have a small or zero sensitivity to extreme values x_1, x_n. When we use the robust estimates as a kernel of ANN preprocessing and postprocessing, the de-noising system will be robust, too.

3 Robust Local De-noising

Let $y_{i,j} = \mathrm{f}(x_1, \ldots, x_n)$ be local de-noising function. The local de-noising is called **k-robust** when f satisfies the condition $x_{(1+k)} \le \mathrm{f}(\overline{x}) = \varphi(x_{(1+k)}, \ldots, x_{(n-k)}) \le x_{(n-k)}$ for all $\overline{x} \in [0;1]^n$. It means the values of $x_i \notin [x_{(1+k)}; x_{(n-k)}]$ are not used in the de-noising procedure and the de-noised value is also constrained. So, the main role of preprocessing is in eliminating the extreme input values while the output of ANN is mapped into interval $[x_{(1+k)}; x_{(n-k)}]$. The general scheme of robust local de-noising is depicted on the Fig 3, where $x_{(1+k)} \le x_{\mathrm{LOW}} \le x_{\mathrm{MID}} \le x_{\mathrm{UPP}} \le x_{(n-k)}$.

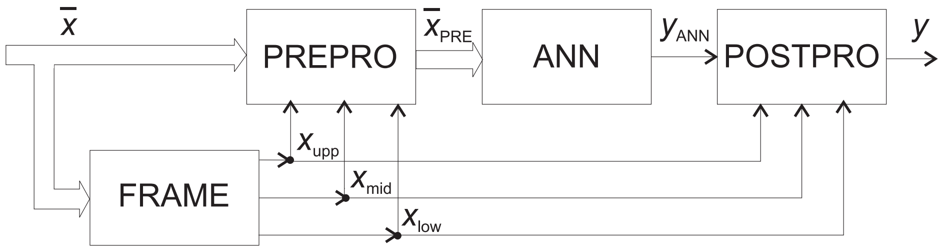

Figure 3: Robust local de-noising

3.1 Robust preprocessing

Let $n = (2r+1)^2$ be neighborhood size. Let $k \in \mathbf{N}$, $k < \frac{n-1}{2}$ be order of robustness. Let $\mathrm{cut}(\alpha) = \min(1, \max(0, \alpha))$. Let $x_{\mathrm{LOW}}, x_{\mathrm{MID}}, x_{\mathrm{UPP}}$ satisfy $x_{(1+k)} \le x_{\mathrm{LOW}} \le x_{\mathrm{MID}} \le x_{\mathrm{UPP}} \le x_{(n-k)}$.

Preprocessing $\Phi_{\mathrm{A}} : [0;1]^n \to [-1;1]^n$ is defined by the formulas

$$\overline{x}_{\mathrm{PRE}} = \Phi_{\mathrm{A}}(\overline{x})$$

$$x_{\mathrm{PRE},i} = 2 \ \mathrm{cut}\left(\frac{x_i - x_{\mathrm{LOW}}}{x_{\mathrm{UPP}} - x_{\mathrm{LOW}}}\right) - 1, \ \text{for } x_{\mathrm{UPP}} > x_{\mathrm{LOW}}$$

$$x_{PRE,i} = 0, \ \text{for } x_{\mathrm{UPP}} = x_{\mathrm{LOW}}$$

where $i = 1, \ldots, n$.

Preprocessing $\Phi_{\mathrm{B}} : [0;1]^n \to [-1;1]^n$ is defined by the formulas

$$\overline{x}_{\mathrm{PRE}} = \Phi_{\mathrm{B}}(\overline{x})$$

$$a_i = \mathrm{cut}\left(\frac{x_i - x_{\mathrm{MID}}}{x_{\mathrm{UPP}} - x_{\mathrm{MID}}}\right), \quad \text{for} \quad x_{\mathrm{UPP}} > x_{\mathrm{MID}}$$

$$a_i = 0 \ , \quad \text{for} \quad x_{\mathrm{UPP}} = x_{\mathrm{MID}}$$

$$
\begin{aligned}
b_i = \mathrm{cut}\left(\frac{x_i - x_{\mathrm{MID}}}{x_{\mathrm{LOW}} - x_{\mathrm{MID}}}\right), &\quad \text{for} \quad x_{\mathrm{LOW}} < x_{\mathrm{MID}} \\
b_i = 0\ , &\quad \text{for} \quad x_{\mathrm{LOW}} = x_{\mathrm{MID}} \\
x_{\mathrm{PRE},i} = a_i - b_i &
\end{aligned}
$$

for $i = 1, \dots, n$.

Preprocessing $\Phi_{\mathrm{C}} : [0;1]^n \to [0;1]^{n-2k}$ is defined by the formulas

$$
\begin{aligned}
\overline{x}_{\mathrm{PRE}} &= \Phi_{\mathrm{C}}(\overline{x}) \\
x_{\mathrm{PRE},i} &= x_{(i+k)}
\end{aligned}
$$

for $i = 1, \dots, n - 2k$.

Preprocessing strategies can be applied to Walsh list $\mathcal{W}(\overline{x})$ to obtain another preprocessing

$$
\begin{aligned}
\Phi_{\mathrm{D}}(\overline{x}) &= \Phi_{\mathrm{A}}(\mathcal{W}(\overline{x})) \\
\Phi_{\mathrm{E}}(\overline{x}) &= \Phi_{\mathrm{B}}(\mathcal{W}(\overline{x})) \\
\Phi_{\mathrm{F}}(\overline{x}) &= \Phi_{\mathrm{C}}(\mathcal{W}(\overline{x})).
\end{aligned}
$$

It is clear, that Φ_{A}, Φ_{B}, Φ_{C} do not use the values $x_{(1)}, \dots, x_{(k)}$ and $x_{(n-k)}, \dots, x_{(n)}$ and the preprocessing are k-robust.

Walsh list $\mathcal{W}(\overline{x})$ consists of $n^* = n(n+1)/2$ elements. The original value $x_{(1)}$ has influence on n values from $\mathcal{W}(\overline{x})$ and $x_{(k)}$ influences $k^* = k(2n - k + 1)/2$ values from Walsh list. So, we use only the values $\xi(1 + k^*), \dots, \xi(n^* - k^*)$ from $\mathcal{W}(\overline{x})$ to be sure in k-robustness of Walsh preprocessing. The preprocessing with or without Walsh list is reasonable when at least two (potentially different) values are passed. In case of Walsh list we must accept the condition

$$1 + k^* < n^* - k^*$$

which is equivalent to $k^* < (n^* - 1)/2$. After the substitution we obtain

$$\frac{k(2n - k + 1)}{2} < \left(\frac{n(n+1)}{2} - 1\right)/2$$

and the explicit constrain

$$k < \frac{2n + 1 - \sqrt{2n^2 + 2n + 5}}{2} \leq \frac{n-1}{2}\ .$$

In case of Walsh list absence we have a simpler and wider condition $1 + k < n - k$ which has the explicit form $k < (n - 1)/2$. Thus, the maximum robustness of Walsh preprocessing is

$$k_{\mathrm{WALSH}} = \left\lceil \frac{2n - 1 - \sqrt{2n^2 + 2n + 5}}{2} \right\rceil$$

while the maximum robustness without Walsh list is

$$k_{\mathrm{MAX}} = \left\lceil \frac{n-3}{2} \right\rceil .$$

Then any preprocessing which begins with $\Phi_{\mathrm{A}}, \dots, \Phi_{\mathrm{F}}$ and continue with $\overline{x}_{\mathrm{PRE}}$ instead of $\overline{x}$ is k-robust with the zero sensitivity to the outliers.

3.2 ANN processing

Let $n_{\mathrm{PRE}} > 1$ be the number of the preprocessing output. Let $\overline{x}_{\mathrm{PRE}} \in [-1;+1]^{n_{\mathrm{PRE}}}$ be the **preprocessing output** defined as $\overline{x}_{\mathrm{PRE}} = \mathrm{F}(\Phi(\overline{x}))$ where Φ is one of k-robust preprocessing. The signal $\overline{x}_{\mathrm{PRE}}$ is incoming to the input of ANN which is supported to realize mapping

$$\mathrm{ANN} : [-1;+1]^{n_{\mathrm{PRE}}} \to [-1;+1]$$

without the output value $y_{\mathrm{ANN}} = \mathrm{ANN}(\overline{x}_{\mathrm{PRE}})$.

There are many possibilities how to design the neural network with one output: an optimum linear neuron (OLAM), a constrained linear neuron, a sigmoidal neuron, a multilayer perceptron (MLP) or a network with radial basis (RBF).

3.2.1 OLAM processing

In case of preprocessing $\overline{x}_{\mathrm{PRE}} = \Phi_{\mathrm{C}}(\overline{x})$ or $\overline{x}_{\mathrm{PRE}} = \Phi_{\mathrm{F}}(\overline{x})$ we can use an optimum linear neuron with zero bias to obtain

$$y_{\mathrm{ANN}} = \sum_{j=1}^{n_{\mathrm{PRE}}} w_j x_{\mathrm{PRE},j}$$

with weights constrains $\sum_{j=1}^{n_{\mathrm{PRE}}} w_j = 1$, $\overline{w} \in [0;1]^{n_{\mathrm{PRE}}}$. Having $x_{\mathrm{PRE},j} \in [x_{(1+k)}; x_{(n-k)}]$, the condition $y_{\mathrm{ANN}} \in [x_{(1+k)}; x_{(n-k)}]$ holds and no other robust processing is necessary.

3.2.2 Constrained linear neuron

We can use any preprocessing together with any linear neuron. But the normalization is necessary. The cut function helps us to perform a constrained linear neuron as

$$y_{\mathrm{ANN}} = 2\ \mathrm{cut}(w_0 + \sum_{j=1}^{n_{\mathrm{PRE}}} w_j x_{\mathrm{PRE},j}) - 1$$

where $\overline{w} \in \mathbf{R}^{n_{\mathrm{PRE}}+1}$.

3.2.3 Sigmoidal neuron

The traditional bipolar smooth model of neuron is described as

$$y_{\mathrm{ANN}} = \tanh(w_0 + \sum_{j=1}^{n_{\mathrm{I}}} w_j x_{\mathrm{PRE},j})$$

where $\overline{w} \in \mathbf{R}^{n_{\mathrm{I}}+1}$. The effect of other sigmoidal characteristics was not studied here.

3.2.4 Multilayer perceptron (MLP)

The existence of a single hidden layer within an artificial neural network improves the approximation power of ANN. Let $H \geq 2$ be number of hidden neurons with a hidden vector $\overline{h} = (h_1, \ldots, h_H) \in (-1;+1)^H$. Then MLP is described by the formulas

$$y_{\mathrm{ANN}} = \tanh(v_0 + \sum_{i=1}^{H} v_i h_i)$$

$$h_i = \tanh(w_{i,0} + \sum_{j=1}^{n_{\mathrm{PRE}}} w_{i,j} x_{\mathrm{PRE},j})$$

where $i = 1, \ldots, H,\ \ \overline{v} \in \mathbf{R}^{H+1},\ \ \mathbf{W} \in \mathbf{R}^{H \times (n_{\mathrm{PRE}}+1)}$.

3.2.5 Constrained RBF network

Another well known model of hierarchical processing is called radial basis function (RBF) network. It also contains single hidden layer of size $H \geq 2$ with hidden vector $\overline{h} = (h_1, \ldots, h_H) \in (0;1]^H$. The RBF network is described by formulas

$$y_{\text{ANN}} = 2\ \text{cut}(v_0 + \sum_{i=1}^{H} v_i h_i) - 1$$

$$h_i = \exp\left(-\frac{1}{2\sigma_i^2} \sum_{j=1}^{n_{\text{PRE}}} (x_{\text{PRE},j} - w_{i,j})^2 \right)$$

where $i = 1, \ldots, H,\ \ \overline{v} \in \mathbf{R}^{H+1},\ \ \overline{\sigma} \in \mathbf{R}_+^H,\ \ \mathbf{W} \in \mathbf{R}^{H \times n_{\text{PRE}}}$.

3.2.6 ANN learning

The vectors $\overline{w}$, $\overline{v}$, $\overline{\sigma}$ and the matrix $\mathbf{W}$ are unknown and can be subject of estimation, learning or optimization. Let $m \in \mathbf{N}$ be a number of patterns. The i^{th} **pattern** is a pair $(\overline{x}_{\text{PRE},i}\ ,\ y^*_{\text{ANN},i})$ for $i = 1, \ldots, m$. Here the vector $\overline{x}_{\text{PRE},i}$ is obtained via preprocessing from the neighborhood of i^{th} pixel taken at random from noised 2D gray image. The value $y^*_{\text{ANN},i} \in [-1;+1]$ represents given output of ANN for i^{th} pixel of an ideal image. There is a relationship between $y^*_{\text{ANN},i}$ and $y^*_{\text{IDEAL},i}$ which is done by a robust postprocessing. Here $y^*_{\text{IDEAL},i} \in [0;1]$ represents given intensity of i^{th} pixel from an ideal image. The patterns form a **pattern set**

$$\mathcal{P} = \{(\overline{x}_{\text{PRE},i}\ ,\ y^*_{\text{ANN},i})\ |\ \ i = 1, \ldots, m\}.$$

In case of a general artificial neural network we have $y_{\text{ANN}} = \text{ANN}(\overline{x}_{\text{PRE}})$ and the method of least squares can be used for the optimization of $\overline{w}$, $\overline{v}$, $\overline{\sigma}$, $\mathbf{W}$. The objective function for minimization is then

$$\text{SSQ} = \sum_{i=1}^{m} (y^*_{\text{ANN},i} - \text{ANN}(\overline{x}_{\text{PRE},i}))^2$$

There are many gradient, stochastic gradient, conjugate gradient, variable metric and other methods for finding the local optimum values of ANN weights. (See [...,...,...]).

3.3 Robust postprocessing

The last step of local k-robust image de-noising realizes the mapping $y = \Psi(y_{\text{ANN}})$ satisfying $y \in [x_{\text{LOW}}; y_{\text{UPP}}]$ for all $y_{\text{ANN}} \in [-1;+1]$ where $x_{(1+k)} \leq x_{\text{LOW}} \leq x_{\text{MID}} \leq x_{\text{UPP}} \leq x_{(n-k)}$. The mapping Ψ is called **robust postprocessing**. Now we can define three basic postprocessing.

Postprocessing Ψ_A is defined by the formulas

$$y = \Psi_{\text{A}}(y_{\text{ANN}})$$

$$y = x_{\text{LOW}} + \frac{y_{\text{ANN}} + 1}{2}(x_{\text{UPP}} - x_{\text{LOW}})$$

Postprocessing Ψ_{B} is defined by the formulas

$$y = \Psi_{\text{B}}(y_{\text{ANN}})$$

$$y = x_{\text{MID}} + \max(0, y_{\text{ANN}})(x_{\text{UPP}} - x_{\text{MID}}) + \min(0, y_{\text{ANN}})(x_{\text{MID}} - x_{\text{LOW}})$$

Postprocessing Ψ_{C} is defined by the formulas

$$y = \Psi_{\text{C}}(y_{\text{ANN}})$$

$$y = \min(x_{\text{UPP}}, \max(x_{\text{LOW}}, y_{\text{ANN}}))$$

Now we are prepared to build up any **k-robust local image de-noising** from the robust preprocessing, ANN inside and the robust postprocessing. The last question is how to obtain given value y^*_{ANN} for ANN learning. It can be obtained from y^* which is given value of a pixel intensity from an ideal image. Except of the outlier values of y^*, the inversion Ψ^{-1} of the postprocessing function Ψ is necessary. When $y^* > x_{\text{UPP}}$ then $y^*_{\text{ANN}} = 1$. When $y^* < x_{\text{LOW}}$ then $y^*_{\text{ANN}} = -1$. When $x_{\text{UPP}} = x_{\text{LOW}}$ then $y^*_{\text{ANN}} = 0$. In the last case, when $y^* \in (x_{\text{LOW}}; x_{\text{UPP}})$, the inversions provide adequate results.

For Ψ_{A} we obtain

$$y^*_{\text{ANN}} = \Psi_{\text{A}}^{-1}(y^*) = 2\frac{y^* - x_{\text{LOW}}}{x_{\text{UPP}} - x_{\text{LOW}}} - 1.$$

After the inversion of Ψ_{B} we obtain three results. When $y^* \in (x_{\text{MID}}; x_{\text{UPP}})$ then

$$y^*_{\text{ANN}} = \Psi_{\text{B}}^{-1}(y^*) = \frac{y^* - x_{\text{MID}}}{x_{\text{UPP}} - x_{\text{LOW}}}.$$

When $y^* \in (x_{\text{LOW}}; x_{\text{MID}})$ then

$$y^*_{\text{ANN}} = \Psi_{\text{B}}^{-1}(y)^* = -\frac{y^* - x_{\text{MID}}}{x_{\text{LOW}} - x_{\text{MID}}}.$$

When $y^* = x_{\text{MID}}$ then

$$y^*_{\text{ANN}} = \Psi_{\text{B}}^{-1}(y^*) = 0.$$

The inversion of Ψ_{C} is trivial as $y^*_{\text{ANN}} = \Psi_{\text{C}}^{-1}(y^*) = y^*$.

If you compare the preprocessing Φ_{A} and inverse postprocessing Φ_{A}^{-1}, you can recognize their similarity. When we apply cut form of Φ_{A}^{-1} to every element of vector $\overline{x}$, the mapping Φ_{A} is obtained.

4 De-noising Strategies

The real implementation of k-robust consists of selected preprocessing, ANN and postprocessing. The de-noising strategy begins with the selection of the neighborhood size r and the robustness order k. We recommend $r = 1,\ k \in \{1; 2\}$ for the first experiments. Thus $n = 9,\ n^* = 45,\ k^* \in \{9; 17\}$ and then 1-robust and 2-robust de-noising system can be constructed with or without Walsh list, and with OLAM, MLP or RBF ANN inside. There are three main strategies of selection x_{LOW}, x_{MID}, x_{UPP}.

4.1 Referential filter in basic frame

Having our favorite de-noising filter $y = f_{\text{REF}}(\overline{x})$, we can call it the **referential filter**. The frame is derived from the order of robustness and then the **referential filter in frame** brings the constrains

$$x_{\text{LOW}} = x_{(1+k)},$$

$$x_{\text{UPP}} = x_{(n-k)},$$

$$x_{\text{MID}} = \min(x_{\text{UPP}}, \max(x_{\text{LOW}}, f_{\text{REF}}(\overline{x})))$$

where $k < (n-1)/2$.

4.2 Referential filter in Walsh frame

We can constrain the referential filter according to Walsh list $\mathcal{W}(\overline{x})$ to obtain another formulas

$$x_{\text{LOW}} = \xi_{(1+k^*)},$$

$$x_{\text{UPP}} = \xi_{(n^*-k^*)},$$

$$x_{\text{MID}} = \min(x_{\text{UPP}}, \max(x_{\text{LOW}}, \mathrm{f}_{\text{REF}}(\overline{x})))$$

where $n^* = n(n+1)/2,\ k^* = k(2n-k+1)/2,\ k^* < (n^*-1)/2.$

4.3 Co-referential frame

Let f_{REF} be any referential filter. Let $n_{\text{CR}} \in \mathbf{N}$ be number of co-referential filters. Let f_{CR_i} be i^{th} co-referential k-robust local de-noising filter for $i = 1, \ldots, n_{\text{CR}}$. Then the **co-referential frame** is defined as

$$x_{\text{LOW}} = \min_{i=1,\ldots,n_{\text{CR}}} (\mathrm{f}_{\text{CR}_i}(\overline{x}))$$

$$x_{\text{UPP}} = \max(\mathrm{f}_{\text{CR}_i}(\overline{x}))$$

$$x_{\text{MID}} = \min(x_{\text{UPP}}, \max(x_{\text{LOW}}, \mathrm{f}_{\text{REF}}(\overline{x})))$$

This approach brings a very sophisticated tool for the image de-noising.

5 Experimental Part

The MRI T2 2D slice of human brain was used for the testing of 1-robust and 2-robust ANN filters. The original image has size 512x512 pixels. The sub-images of size 70x70 pixels were cut out for the learning and verification. The Gaussian noise was added to obtain sources for ANN learning. The original image is depicted on the figure 4. The figures 5 and 6 demonstrate two versions of noised 2D image. The referential filter f_{REF} was set to be median ($\text{MED}_0(\mathcal{L})$). Five co-referential filters were used: BES ($\text{BES}(\mathcal{L})$), quasimedian ($\text{MED}_1(\mathcal{L})$), median of Walsh list ($\text{MED}_0(\mathcal{W})$), quasimedian of Walsh list ($\text{MED}_1(\mathcal{W})$) and BES of Walsh list ($\text{BES}(\mathcal{W})$). The set of co-referential filters satisfies the condition of 1-robustness. The robust preprocessing schemes Φ_A, Φ_B were conquered with adequate postprocessing Ψ_A, Ψ_B for robustness indict k=1 and k=2. The influence of ANN type was studied for OLAM, MLP and RBF networks. The effect of Walsh preprocessing and co-referential filtering was also measured. The quality of de-noising was measured via SNR of ANN enhanced 2D image. The difference between any k-robust filter and referential filter is denoted here as $\Delta\text{SNR} = \text{SNR} - \text{SNR}_{\mathrm{f}_{\text{REF}}}$. The weights of ANN was learned on learning frame of original image (Fig4). The results of verification are demonstrated in Tables 1, 2. Table 1 consists of results for 1^{st} frame within 1^{st} noised image (Fig5) while table 2 collects the results for 2^{nd} frame within 2^{nd} noised image (Fig6). The quality of de-noising is also evaluated for the referential median filter, which is 4-robust and also for co-referential filters which are 1-robust at least. The results of verification on the whole set of six frames can be generalized to several rules of application:

- increasing of k-robustness while decrease SNR of optimum system
- preprocessing Φ_A with postprocessing Ψ_A is better than Φ_B with Ψ_B in the majority of cases
- MLP network is better than OLAM and RBF in the majority of cases
- co-referential frame is better than the worst individual co-referential filter
- Walsh frame is better than co-referential frame

- basic frame is better than co-referential frame
- basic frame and Walsh frame are very close in SNR

6 Conclusions

The k-robust de-noising filter with ANN inside were defined first and then realized for $k \in \{1; 2\}$. The main recommendation from experimental testing on MR image of human brain with Gaussian noise are: use basic or Walsh frame for k=1,Φ_A preprocessing, MLP network and Ψ_A postprocessing to obtain good ΔSNR values. In the case of higher probability of impulse noise, the value k=2 is necessary. The traditional median of nine values is reserved only for the extreme case.

7 Acknowledgement

The work has been supported by the fund No. MSM 6046137306 of Ministry of education of the Czech Republic. This support is very gratefully acknowledged.

References

[1] Haykin, S.: *Neural Networks*. Macmillan, New York, 1994.

[2] Fausett, L.: *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*. Prentince Hall, New Jersey, 1994.

[3] Klette, R., Zamperoni, P.: *Handbook of Image Preprocessing Operators*. John Wiley and Sons, Chichester, 1996.

[4] Hodges, J. L., Lehmann, E. L.: *On Medians and Quasimedians*. Journal of the American Statistical Asscociation, 62:926-931, 1967.

Jaromír Kukal, Luboš Hainc
Institute of Chemical Technology, Prague
Department of Computing and Control Engineering
Technická 5, 166 28 Prague 6 Dejvice
Phone: 420-2-2435 4212, E-mail: Jaromir.Kukal@vscht.cz

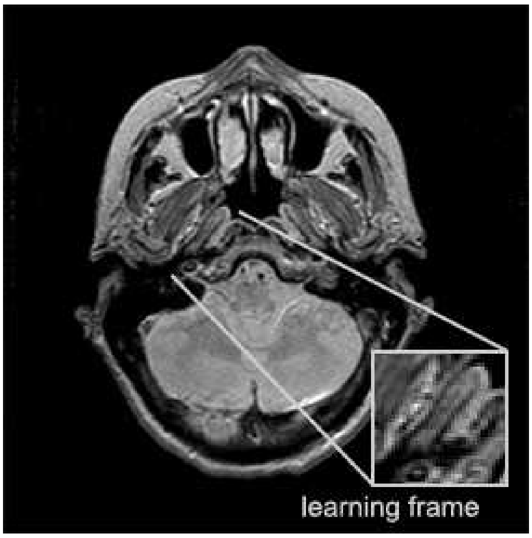

Figure 4: Original image

filter	k=1		k=2	
	SNR	Δ SNR	SNR	Δ SNR
NOISED	8.2982	–	–	–
f_{REF} - median	12.1031	0	–	–
OLAM with Φ_{A}	13.7043	1.6012	13.5234	1.4203
OLAM with Φ_{B}	13.4160	1.3129	13.2187	1.1156
MLP with Φ_{A}	13.9269	1.8238	13.7313	1.6282
MLP with Φ_{B}	13.5703	1.4672	13.3683	1.2652
RBF with Φ_{A}	13.7830	1.6799	13.6203	1.5172
RBF with Φ_{B}	13.6207	1.5176	13.3052	1.2021
OLAM with Φ_{A} and Walsh list	13.7255	1.6224	12.9406	0.8375
OLAM with Φ_{B} and Walsh list	13.5976	1.4945	12.8924	0.7893
MLP with Φ_{A} and Walsh list	13.9243	1.8212	13.2182	1.1151
MLP with Φ_{B} and Walsh list	13.6974	1.5943	13.0829	0.9798
RBF with Φ_{A} and Walsh list	13.7049	1.6018	13.1672	1.0641
RBF with Φ_{B} and Walsh list	13.6952	1.5921	13.1588	1.0557
f_{CR_1} (BES)	12.2676	0.1645	–	–
f_{CR_2} (quasimedian)	12.8443	0.7399	–	–
f_{CR_3} (median Walsh)	11.5736	-0.5295	–	–
f_{CR_4} (quasimedian Walsh)	12.3779	0.2748	–	–
f_{CR_5} (BES Walsh)	12.2635	0.1604	–	–
COREF OLAM with Φ_{A}	12.7708	0.6677	–	–
COREF OLAM with Φ_{B}	12.5636	0.4605	–	–
COREF MLP with Φ_{A}	12.7510	0.6479	–	–
COREF MLP with Φ_{B}	12.5702	0.4671	–	–
COREF RBF with Φ_{A}	12.6892	0.5861	–	–
COREF RBF with Φ_{B}	12.6230	0.5199	–	–

Table 1: Filter properties (Fig. 5, Frame 1)

filter	k=1		k=2	
	SNR	Δ SNR	SNR	Δ SNR
NOISED	8.2982	–	–	–
f_{REF} - median	9.5856	0	–	–
OLAM with Φ_{A}	11.0099	1.4243	10.6751	1.0895
OLAM with Φ_{B}	10.8239	1.2383	10.5512	0.9656
MLP with Φ_{A}	11.0925	1.5069	10.6882	1.1026
MLP with Φ_{B}	10.6032	1.0176	10.5355	0.9499
RBF with Φ_{A}	10.6220	1.0364	10.5903	1.0047
RBF with Φ_{B}	10.5018	0.9162	10.5428	0.9572
OLAM with Φ_{A} and Walsh list	10.7252	1.1396	10.1118	0.5262
OLAM with Φ_{B} and Walsh list	10.6792	1.0936	10.1019	0.5163
MLP with Φ_{A} and Walsh list	10.6704	1.0848	10.1076	0.5220
MLP with Φ_{B} and Walsh list	10.1567	1.5711	10.1188	0.5332
RBF with Φ_{A} and Walsh list	10.2012	1.6156	10.1956	0.6100
RBF with Φ_{B} and Walsh list	10.1835	1.5979	10.1574	0.5718
f_{CR_1} (BES)	9.6230	0.0374	–	–
f_{CR_2} (quasimedian)	9.6012	0.0156	–	–
f_{CR_3} (median Walsh)	9.5989	0.0133	–	–
f_{CR_4} (quasimedian Walsh)	9.6401	0.0574	–	–
f_{CR_5} (BES Walsh)	9.6725	0.0869	–	–
COREF OLAM with Φ_{A}	9.7652	0.1796	–	–
COREF OLAM with Φ_{B}	9.7012	0.1156	–	–
COREF MLP with Φ_{A}	9.7545	0.1689	–	–
COREF MLP with Φ_{B}	9.7059	0.1203	–	–
COREF RBF with Φ_{A}	9.7324	0.1468	–	–
COREF RBF with Φ_{B}	9.7241	0.1385	–	–

Table 2: Filter properties (Fig. 6, Frame 2)

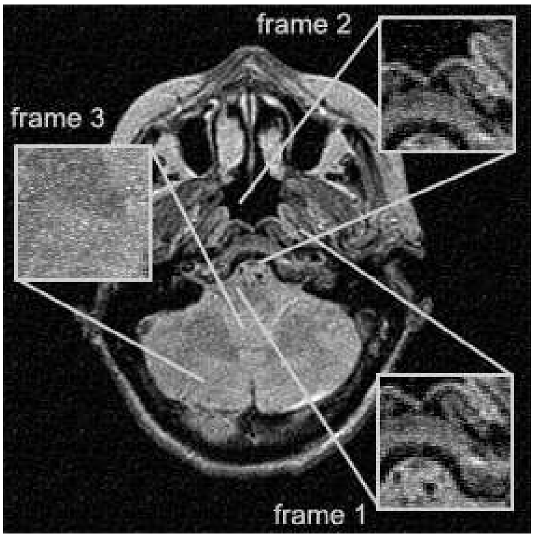

Figure 5: First noised image

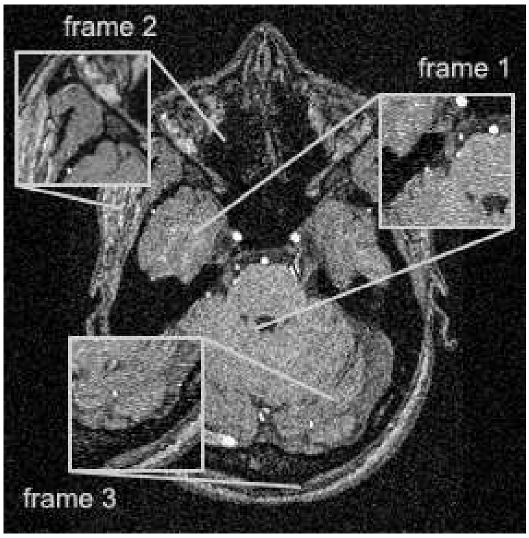

Figure 6: Second noised image