Introduction to Project Topics

Kaicong Sun

Organizational Issues

- Projects can be done in groups of two or three.
- Submit the codes and report to Kaicong.Sun@ipvs.uni-stuttgart.de
  - Template in ILias
  - Workflow
  - Codes with comments
  - If possible, compare with CPU implemetation
- Written report: 6-12 pages
- Submission deadline: 31.09.2019

Topics

- Implementation of Modulation Transfer Function (MTF) Measurement on Cylindrical Object: ASTM-E 1695-95.
- Implementation of Modulation Transfer Function (MTF) Measurement on Slanted Edge: ISO 12233:2017.
- Implementation of the Calculation of Sparse Matrix (CRS).
- Implementation of (Alternating Direction Method of Multipliers) ADMM Optimizor for Given Energy Function Using Scaled Conjugate Gradient (SCG) Method.
- Implementation of (Alternating Direction Method of Multipliers) ADMM Optimizor for Given Energy Function Using ADAM Method.
- 2D Fourier Transform.
- Canny Edge Detector.

Topics

Modulation Transfer Function (MTF) Measurement on Cylindrical Object: ASTM-E1695-95

Modulation transfer function (MTF) is widely used as a metric for spatial resolution assessment. This project is aimed to implement the MTF measurement of the computed tomography (CT) system based on the norm ASTM-E 1695-95 [1].

CT images of the test object, i.e., a phantom disk made of Aluminium with diameter 20mm, are given. Specifically,

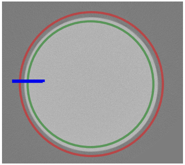
- Three test CT images are given with Input4, Input7, Input10.
- In the file Readme you can find the pixelsize of each CT image, which will be needed when you calculate MTF.
- Compare your MTF curves with the corresponding given MTF curves, noticing the setup parameters: binsize, search distance and fit point count.

Modulation Transfer Function (MTF) Measurement on Cylindrical Object: ASTM-E1695-95

- Calculate the centre of the phantom in the CT slice.
- Choose inner and outer radii with respect to the centre of circle that bracket the edge.
- Segregate the region between inner and outer radii with bins sized to a small fraction of one pixel.
- Averaging the value of bins according to the distance to the centre.
- Smoothing the averaged curve crossing the edge and do a piece-wise, least-squares cubic fit (ERF).
- Calculate the first derivative of the curve ERF to get PSF.
- Calculate the Fourier Transform of the PSF and normalize the maxima to one (MTF).

Modulation Transfer Function (MTF) Measurement on Cylindrical Object: ASTM-E1695-95

Inner and outer radii



1. Split 1 pixel to subpixels (according to table in ASTM)
2. Calculate the center and radius of the circle
3. Define the region with outer and inner radii
4. Average all the profiles in the region
5. Piece-weise least-squares cubic fit
6. Got the white curve beneath

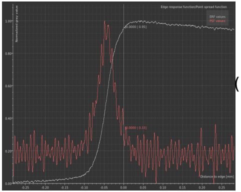
Edge Spread Function (ESF)

First derivative

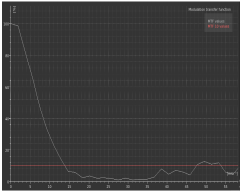
Point Spread Function (PSF)

1D Fourier Transform

Modulation Transfer Function (MTF)



1D FFT (padding)



MTF Measurement on Slanted Edge: ISO 12233:2017[2]

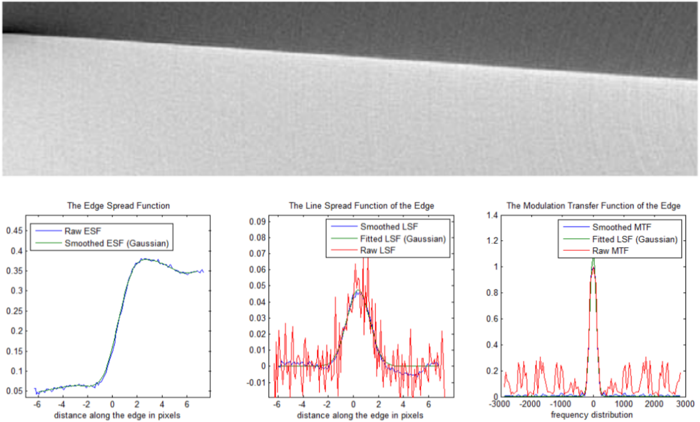


Figure: Source: Daniel Weiß, et al, "Measuring the 3D resolution of a micro-focus X-ray CT setup [3]." (Refer to and Compare with matlab script: Slant Edge Script)

Calculation of Sparse Matrix (CRS)

Implement the calculation of a sparse matrix in format compressed row storage (CRS). Instead of using

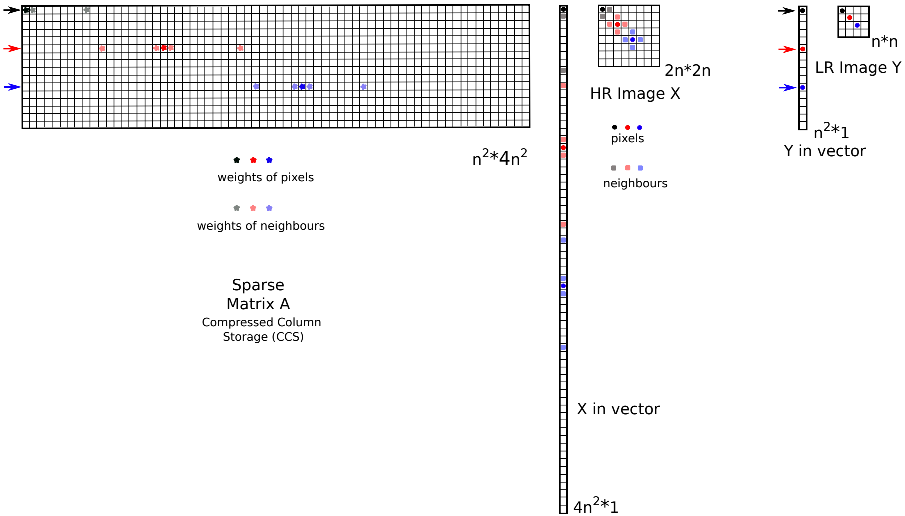
```
Eigen::SparseMatrix<T,Eigen::RowMajor, int> Mmatrix(r, c);
Mmatrix.coeffRef(index, neighborBL) = x;
```

x should directly saved based on ”Values, InnerIndices, OuterStarts”. The sparse matrix A is applied to degrade (downsampling, shift, blur) a high resolution (HR) image X with size $2n \times 2n$ to a low resolution (LR) image y with size $n \times n$.

$$A_{n^2 \times 4n^2} \cdot X_{4n^2 \times 1} = y_{n^2 \times 1}$$

Further information: `https://en.wikipedia.org/wiki/Sparse_matrix`
`https://eigen.tuxfamily.org/dox/group__TutorialSparse.html`
`https://eigen.tuxfamily.org/dox/group__SparseQuickRefPage.html`

Calculation of Sparse Matrix (CRS)



Alternating Direction Method of Multipliers (ADMM)

- An optimization problem solover with good robustness of method of multipliers
- Support decomposition

ADMM (Alternating Direction Method of Multipliers) deals with the following problem [4].

$$\begin{aligned} \textit{minimize} \quad & f(x) + g(y) \\ \textit{subject to} \quad & Ax + By = c \end{aligned} \tag{1}$$

Alternating Direction Method of Multipliers (ADMM)

Here, f, g are assumed convex. An auxiliary variable (Lagrange multiplier) z is introduced to form an function $L_\rho(x, y, z)$

$$L_\rho(x, y, z) = f(x) + g(y) + z^T(Ax + By - c) + \frac{\rho}{2}\|Ax + By - c\|_2^2 \quad (2)$$

where ρ is a tunning parameter. Then, we can iteratively solve for x, y, z in three seperate steps (subproblems):

$$\begin{aligned} x^{k+1} &= \arg\min_x L_\rho(x, y^k, z^k) \\ y^{k+1} &= \arg\min_y L_\rho(x^{k+1}, y, z^k) \\ z^{k+1} &= z^k + \rho(Ax^{k+1} + By^{k+1} - c) \end{aligned} \quad (3)$$

Alternating Direction Method of Multipliers (ADMM)

Proximal operator of a function $f(x)$ is defined as:

$$prox_f(v) = \arg\min_x (f(x) + \frac{1}{2}\|x - v\|_2^2) \tag{4}$$

- Proximal operator is defined in a certain format [5]
- Proximal operator can be solved analytically which accerlates the computation speed

The above mentioned three steps of ADMM can benefit from the proximal operator in terms of computation complexity if a reasonable decomposition of your energy function can be determined so that any step of the three could match the format of proximal operator.

Energy Function

$$
\begin{aligned}
J = \frac{1}{2}\sum_{i=1}^{m} (\| \mathbf{y}_i - A_i\mathbf{x} - \mu_i \|_{W_i}^2 + \langle \log (A_i\mathbf{x} + \sigma_i^2), 1) \rangle \\
+ \beta \sum_{p=0}^{w}\sum_{q=0}^{w} \gamma(p,q) \| x - S_x^p S_y^q x \|_1 + \chi_B(\mathbf{x})
\end{aligned}
\tag{5}
$$

where $< \cdot >$ indicates a pointwise multiplication of two vectors.
A_i is constant matrices with size mxn.
S_x, S_y are shift operators along x- and y-axis. σ_i is constant vector.
w is a constant for the window size and β is constant weight.
y_i is the input image with size $mx1$. x is output image with size $nx1$.
m expresses the number of inputs y_i, we make $m = 4$.
W_i is a diagonal weight matrix and can be expressed as

$$W_i = \text{diag}\{\frac{1}{[A_i]_k\mathbf{x} + [\sigma_i]_k^2}\}.$$

where $[A_i]_k$ is the kth row of matrix A_i.

Energy Function

$$J = \frac{1}{2}\sum_{i=1}^{m}(\| \mathbf{y}_i - A_i\mathbf{x} - \mu_i \|_{W_i}^2 + \langle \log(A_i\mathbf{x} + \sigma_i^2), 1)\rangle$$
$$+ \beta \sum_{p=0}^{w}\sum_{q=0}^{w} \gamma(p,q) \| x - S_x^p S_y^q x \|_1 + \chi_B(\mathbf{x}) \quad (6)$$

Here, $\| \cdot \|_1$ indicates the Euclidean l-1 norm.
We can define $\gamma(p,q) = \alpha^{|p|+|q|}$ where α is a constant.
$\mathcal{X}_B(X)$ is the indicator function of the convex set B which constrains the nonnegativity of the reconstructed $\mathbf{x}$

$$\chi_B(\mathbf{x}) = \begin{cases} 0, & \mathbf{x} \subseteq B, \\ +\infty, & \mathbf{x} \subsetneq B, \end{cases} \quad (7)$$

with $B = \{\mathbf{x} : x_k \geq 0, \forall k \in [1, n]\}$.

Energy Function in ADMM

The energy function J can be formulated:

$$J(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{m+w^2+1} g_i(\mathbf{z}_i) \quad \text{subject to } T_i\mathbf{x} - \mathbf{z}_i = 0, \quad \forall i \in [1, m + w^2 + 1] \tag{8}$$

with:

$$T_i = \begin{cases} I_{n\times n} & , i \in [1, m] \\ I_{n\times n} - S_{\mathbf{d}} & , i \in [m+1, m+w^2] \\ I_{n\times n} & , i = m + w^2 + 1 \end{cases}$$

Energy Function in ADMM

Specially, multiple g_i are defined as follows:

$$g_i(\mathbf{z}_i) := \frac{1}{2}(||\mathbf{y}_i - A_i\mathbf{z}_i - \mu_i||^2_{W_i} + \langle \log(A_i\mathbf{z}_i + \sigma_i^2), 1\rangle), i \in [1, m],$$
$$g_i(\mathbf{z}_i) := \beta\gamma(\mathbf{d})||\mathbf{z}_i||_1,\ i \in [m+1, m+w^2],$$
$$g_i(\mathbf{z}_i) := \chi_B(\mathbf{z}_i),\ i = m + w^2 + 1.$$

Energy Function in ADMM

The augmented Lagrangian function is formulated as

$$\mathcal{L}_H(\mathbf{x}, \mathbf{z}, \mathbf{p}) := \sum_{i=1}^{m+w^2+1} \mathcal{L}_{H_i}(\mathbf{x}, \mathbf{z}_i, \mathbf{p}_i)$$

$$:= \sum_{i=1}^{m+w^2+1} \left(g_i(\mathbf{z}_i) + \langle \mathbf{p}_i, T_i\mathbf{x} - \mathbf{z}_i \rangle + \frac{1}{2} || T_i\mathbf{x} - \mathbf{z}_i ||_{H_i}^2 \right)$$

where matrix H_i is defined as

$$H_i := \mathrm{diag}[\rho_i, \ldots, \rho_i], \quad \forall i \in [1, \cdots, m + w^2 + 1]$$

with ρ_i being constant.

Energy Function in ADMM

In our case we have $m = 4$ low resolution images and the following iteration scheme:

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \sum_{i=1}^{m+w^2+1} \frac{\rho_i}{2} || T_i\mathbf{x} - \mathbf{z}_i^k + \frac{\mathbf{p}_i^k}{\rho_i} ||_2^2 \tag{9a}$$

$$\mathbf{z}_i^{k+1} = \arg\min_{\mathbf{z}_i} g_i(\mathbf{z}_i) + \frac{\rho_i}{2} ||\mathbf{z}_i - T_i\mathbf{x}^{k+1} - \frac{\mathbf{p}_i^k}{\rho_i}||_2^2 \tag{9b}$$

$$\mathbf{p}_i^{k+1} = \mathbf{p}_i^k + \rho_i(T_i\mathbf{x}^{k+1} - \mathbf{z}_i^{k+1}). \tag{9c}$$

The update of x^{k+1} can be solved by e.g. conjugate gradient. To update $z_i^{k+1}, i \in [1, \cdots, m], g_i(z_i)$ will be solved using e.g. Newton's method, L-BFGS and ADAM and for the rest $z_i^{k+1}, i \in \left[m+1, \cdots, m+w^2\right], g_i(z_i)$ can be calculated using Proximal operator (See following slides).

Energy Function in ADMM

To update x^{k+1} using the conjugate gradient, the partial gradient of $V_i(x) := \frac{\rho_i}{2}||T_i x - z_i^k + \frac{p_i^k}{\rho_i}||_2^2$ must be calculated by

$$\nabla V_i(x) = \rho_i T_i^T (T_i x - z_i^k + \frac{p_i^k}{\rho_i}), \tag{10}$$

where ρ_i is a tunable user-define scalar.
To update z_i^{k+1}, the partial gradient and Hessian (for Newton's method) have to be calculated as below (See following slides).

Hadamard Product

We define the product between a matrix and a vector notated by $\odot$:

$$A_{mxn} \odot \vec{b}_{mx1} = \begin{pmatrix} a_{11} \cdot b_1, & \cdots, & a_{1n} \cdot b_1 \\ \vdots & \ddots & \vdots \\ a_{m1} \cdot b_m, & \cdots, & a_{mn} \cdot b_m \end{pmatrix}$$

$$\vec{a}_{mx1} \odot \vec{b}_{mx1} = \begin{pmatrix} a_1 \cdot b_1 \\ \vdots \\ a_m \cdot b_m. \end{pmatrix}$$

It can be easily implemented by:

$$A_{mxn} \odot \vec{b}_{mx1} = \text{diag}\left[b_1, \cdots, b_m\right] \cdot A_{mxn}$$

$$\vec{a}_{mx1} \odot \vec{b}_{mx1} = \text{diag}\left[a_1, \cdots, a_m\right] \cdot \vec{b}_{mx1}$$

Partial Derivative of Weighted L2

To update $\mathbf{z}_i^{k+1}, i \in [1, m]$, $g_i(\mathbf{z}_i)$ is nonconvex and can be solved using, e.g., Newton's method, L-BFGS and scaled congugate gradient (SCG). For clarity, we define:

$$
\begin{aligned}
g_i(\mathbf{z}_i) &:= f_i(\mathbf{z}_i) + h_i(\mathbf{z}_i), \\
f_i(\mathbf{z}_i) &:= \frac{1}{2} \parallel \mathbf{y}_i - A_i \mathbf{z}_i - \mu_i \parallel_{W_i}^2, \\
h_i(\mathbf{z}_i) &:= \frac{1}{2} \langle \log (A_i \mathbf{z}_i + \sigma_i^2), 1 \rangle, \\
u_i(\mathbf{z}_i) &:= \frac{\rho_i}{2} || \mathbf{z}_i - T_i \mathbf{x}^{k+1} - \frac{\mathbf{p}_i^k}{\rho_i} ||_2^2.
\end{aligned}
$$

Partial Derivative of Weighted L2

The gradient of (9b) on $\mathbf{z}_i$ can be calculated by

$$
\begin{aligned}
&\nabla f_i(\mathbf{z}_i) + \nabla h_i(\mathbf{z}_i)) + \nabla u_i(\mathbf{z}_i) \\
&= \frac{1}{2}(-2A_i^T \frac{\mathbf{y}_i - A_i\mathbf{z}_i - \mu_i}{A_i\mathbf{z}_i + \sigma_i^2} - A_i^T \frac{(\mathbf{y}_i - A_i\mathbf{z}_i - \mu_i)^2}{(A_i\mathbf{z}_i + \sigma_i^2)^2} \\
&\quad + A_i^T \frac{1}{A_i\mathbf{z}_i + \sigma_i^2}) + \rho_i(\mathbf{z}_i - T_i\mathbf{x}^{k+1} - \frac{\mathbf{p}_i^k}{\rho_i}).
\end{aligned}
\tag{11}
$$

Proximal Operators

For calculating $\mathbf{z}_i^{k+1}$ with $i \in \left[m+1, m+w^2\right]$, $g_i(\mathbf{z}_i)$ can be calculated according to the proximal operator of the $\ell 1$ norm as following:

$$\begin{aligned} \mathbf{z}_i^{k+1} &= \arg\min_{\mathbf{z}_i} \beta\gamma(\mathbf{d})||\mathbf{z}_i||_1 + \frac{\rho_i}{2}||\mathbf{z}_i - T_i\mathbf{x}^k - \frac{\mathbf{p}_i^k}{\rho_i}||_2^2 \\ &= prox_{\beta\gamma(\mathbf{d})(\rho_i)^{-1}||\cdot||_1}(T_i\mathbf{x}^k + \frac{\mathbf{p}_i^k}{\rho_i}) \end{aligned} \tag{12}$$

$$[\mathbf{z}_i^{k+1}]_j = \begin{cases} [T_i\mathbf{x}^k + \frac{\mathbf{p}_i^k}{\rho_i}]_j - \frac{\beta\gamma(\mathbf{d})}{\rho_i}, & [T_i\mathbf{x}^k + \frac{\mathbf{p}_i^k}{\rho_i}]_j \geq \frac{\beta\gamma(\mathbf{d})}{\rho_i}, \\ 0, & [|T_i\mathbf{x}^k + \frac{\mathbf{p}_i^k}{\rho_i}|]_j \leq \frac{\beta\gamma(\mathbf{d})}{\rho_i}, \\ [T_i\mathbf{x}^k + \frac{\mathbf{p}_i^k}{\rho_i}]_j + \frac{\beta\gamma(\mathbf{d})}{\rho_i}, & [T_i\mathbf{x}^k + \frac{\mathbf{p}_i^k}{\rho_i}]_j \leq -\frac{\beta\gamma(\mathbf{d})}{\rho_i}. \end{cases}$$

Proximal Operators

The nonnegativity constraint $g_i(\mathbf{z}_i)$ with $i = m + w^2 + 1$ can be solved in the following form based on the proximal operator of the indicator function:

$$\mathbf{z}_i^{k+1} = \max(\mathbf{x}^{k+1} + \frac{\mathbf{p}_i^k}{\rho_i}, 0). \tag{13}$$

Workflow of the Algorithm

Initialize $\beta, w, \mu, \sigma, \rho, \alpha$, *iter_admm*.
Load LR images $\mathbf{y}_i,\ i \in [1, \cdots, m]$.
procedure SOLVING ADMM
  $\mathbf{z}_i^0 = Upscaling(y_1),\ i \in [1, m], \mathbf{z}_i^0 = 0,\ i \in \left[m+1, m+w^2+1\right], \mathbf{x}^0 = 0, p_i^0 = 0,\ i \in \left[1, m+w^2+1\right]$.
  while $k <$ *iter_admm* **do**
    CG($\mathbf{x}^k$) ▷ by 9a, 10
    for $i = 1$ to $m + w^2 + 1$ **do**
      if $i < m + 1$ **then**
        SCG($\mathbf{z}_i^k$) ▷ by 9b, 11
      else if $i < m + w^2 + 1$ **then**
        Prox($\mathbf{z}_i^k$) ▷ by 9b, 12
      else
        Prox($\mathbf{z}_i^k$) ▷ by 9b, 13
      Update $\mathbf{p}_i^k$. ▷ by 9c
    $k = k + 1$
  end while
  return Reconstructed HR image $\mathbf{x}$.

ADMM with Scaled Conjugate Gradient (SCG)

Although ADMM is originally designated for convex problem. It is robust enough even for some nonconvex problems.

For the nonconvex subproblems of ADMM, one could use Scaled Conjugate Gradient (SCG) to solve them [6] based on the gradient on X. For the other subproblems, one could take advantage of proximal operator.

Compare the resultant HR image $\hat{\mathbf{x}}$ with ground truth image $\mathbf{x}^*$ named by *GT.tif* using PSNR (peak signal-to-noise ratio) with $I_{max} = 60$.

$$PSNR(\hat{\mathbf{x}}, \mathbf{x}^*) = 10 \log_{10}\left(\frac{I_{max}^2}{MSE}\right) \tag{14}$$

ADMM with ADAM

For the nonconvex subproblems, one could use adaptive moment estimation (ADAM)to solve them [7].

- A method for stochastic optimation which combines the advantages of two stochastic gradient descent methods AdaGrad and RMSProp.
- An algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. Specifically, it updates the stepsize not only based on the average first moment (the mean) as in RMSProp, but also making use of the average of the second moments of the gradients (the uncentered variance).
- Compare the resultant HR image $\hat{\mathbf{x}}$ with ground truth image $\mathbf{x}^*$ named by *GT.tif* using PSNR with $I_{max} = 60$.

$$PSNR(\hat{\mathbf{x}}, \mathbf{x}^*) = 10 \log_{10}\left(\frac{I_{max}^2}{MSE}\right) \tag{15}$$

2D Fast Fourier Transform

Implement a Fast Fourier Transform on the GPU. Support for non-power-of-two input sizes is optional.

Canny Edge Detector

Implement the Canny edge detector on the GPU. The program should include graphical output (e.g. using OpenGL). There also should be an option to output the various intermediate stages.

Sources

1 ASTM-E1695-95 ”Standard Test Method for Measurement of Computed Tomography (CT) System Performance”.

2 ISO 12233:2017 ”Photography – Electronic still picture imaging – Resolution and spatial frequency responses”.

3 Measuring the 3D resolution of a micro-focus X-ray CT setup.

4 Alternating Direction Method of Multipliers.

5 Proximal Algorithms.

6 A Scaled Conjugate Gradient Algorithm

7 ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION.

8 Quasi-Newton methods.

9 Optimization methods.

10 Newton’s Method for Unconstrained Optimization. for Fast Supervised Learning.