Vol.2011-CVIM-177 No.7
2011/5/19

Global Image Registration using Random Projection

Shuang Lu,[†1] Hayato Itoh,[†1] Tomoya Sakai[†2]
and Atsushi Imiya [†3]

Using the efficient random projection, we develop an efficient algorithm that establishes global image registration. Image registration, which is the first process in image data alignment and applied in remote sensing, image processing, medical image analysis, robot vision, car mounted vision system, and industrial image inspection. The current image registration techniques are classified into local registration and global registration. Our aim in this paper is to speed up global registration using random projection. Random projection refers to a simple technique of projecting a set of points from a high-dimensional space to a randomly chosen low- dimensional subspace and can preserve the geometric properties in the higher-dimensional space. We develop a global registration using random projection We use spectrum spreading and circular convolution to reduce computational cost of random projection.
Keywords **:** Image Registration, Global Registration, Random Projection, Spectrum spreading, Circular convolution, Nearest Neighbor Search

1. Introduction

In this paper, we develop a global image registration using random projection[2)–4)]. Image registration is a fundamental task for image analysis. For instance, in computer vision, image registration is a pre-processing for automatic target recognition and image mosaicing. In medical image analysis, the processing is used for monitoring tumor growth and treatment verification. In cartography and geographic information systems (GIS), the method is used for map updating and integrating of multichannel images[5)]. Therefore, image registration is a crucial step to combine various data sources for in image.

Image registration overlays two or more images, which are the same sense observed at different times, from different viewpoints, and/or by different sensors.

†1 Graduate School, Chiba University
†2 Nagasaki University
†3 IMIT, Chiba University

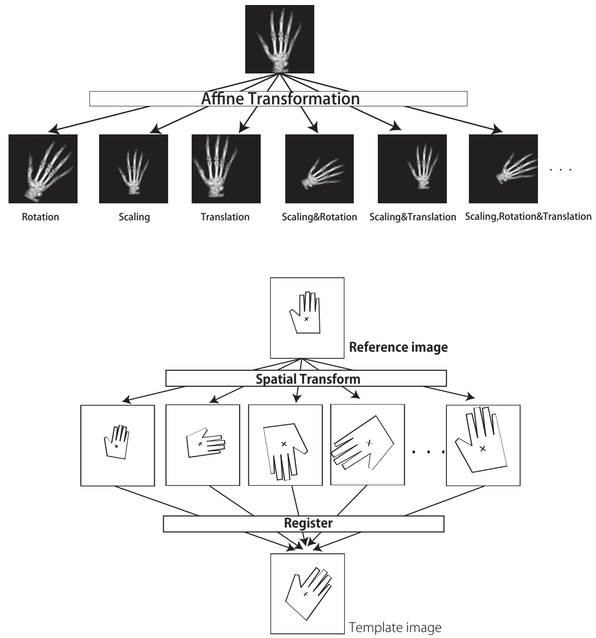


Fig. 1 Affine Transformation and Flow of Global Registration

In general, there are parallel, rotation, scaling, and shearing spatial relationship between reference image and template images. Therefore, image registration is an estimation process of these geometric transformations that transform all points or most points of the template images to points of the reference image. For estimation of these spatial transformations, various methods are developed[5),6)]. Image registration method generally classified into local image registration and global image registration. For global alignment images, linear transformation

$$\boldsymbol{x}' = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}. \tag{1}$$

is used to relate the template image to the reference image. Figures 1(a) 1(b) show linear transformation and the procedure of global registration, respectively.

the nearest neighbour based image registration detects the transformation which establishes the best matching from generated transformations. Performing

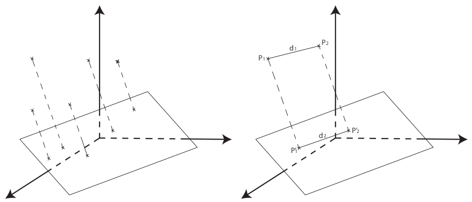


Fig. 2 Geometry of random projection. (a) Random Projection $3D \to 2D$. (b) 2 points P_1' and P_2' in 3-dimensional space correspond to points P_1 and P_2, respectively in 2-dimensional space after random projection.

the inverse affine transformation on template image with the affine parameters, we obtain a new image that approximate the reference image. Then, the registration is completed.

However, there is a computational cost problem on parameters estimation using nearest neighbour search (NNS). To reduce the computational cost of NNS without significant loss of accuracy, random projection refers to a simple technique of projecting a set of points from a high-dimensional space to a randomly chosen low- dimensional subspace[1)] and can preserve the geometric properties in the higher-dimensional space. Our aim in this paper is to speed up global registration using random projection.

2. Random Projection

Let $\boldsymbol{u} = (u_1, \dots, u_n)^T$ and $\boldsymbol{v} = (v_1, \dots, v_k)^T$ be a column vector in n-dimensional Euclidean space and the target column vector in k-dimensional Euclidean space($k < n$). Using a uniform random orthonormal matrix $\boldsymbol{R}$ and scale,

$$\boldsymbol{v} = \sqrt{\frac{n}{k}}\boldsymbol{R}^T\boldsymbol{u}, \tag{2}$$

where the scaling factor $\sqrt{\frac{n}{k}}$ is selected to ensure the expected squared length of $\boldsymbol{v}$ equal to the squared length of $\boldsymbol{u}$.

Random projection approximately preserves pairwise distances with high probability. The Johnson-Lindenstrauss lemma ensure this precise.

Lemma 1 $(\boldsymbol{Johnson-Lindenstrauss})$[7)] For any $\frac{1}{2} > \epsilon > 0$, and any set of points X in $\mathbb{R}^n$, according to eq. (2), upon projection to a uniform random k-dimensional subspace, where $k \geq \frac{4}{\epsilon^2/2-\epsilon^3/3}logn$ the property least $\frac{1}{2}$,

$$(1-\epsilon)|\boldsymbol{u}-\boldsymbol{v}|^2 \leq |f(\boldsymbol{u})-f(\boldsymbol{v})|^2 \leq (1+\epsilon)|\boldsymbol{u}-\boldsymbol{v}||^2 \tag{3}$$

is satisfied for every pair $\boldsymbol{u}$ and $\boldsymbol{v}$ and their projection $f(\boldsymbol{u})$ and $f(\boldsymbol{v})$.

Frankl and Maehara[8)] have shown that the Lemma 1 is true when the projection matrix $\boldsymbol{R}$ is a Random orthonormal matrix, that is, each entry of the random matrix is selected independently from the standard normal distribution $N(0, 1)$ with mean0 and variance 1. Note that the scaling factor is different. Pairwise distance is the distance between random 2 points in n-dimensional space.

3. Efficient Random Projection

Let $\boldsymbol{w} = [w_1, \dots, w_d]^\top$ be an independent, such that, with $E[\boldsymbol{w}] = 0$, $E[\boldsymbol{w}\boldsymbol{w}^\top] = \gamma^2\boldsymbol{I}$. Using he $(i-1)$-time shifting of $\boldsymbol{w}$ such that $\boldsymbol{c_i} = [w_i, \dots, w_d, w_1, \dots, w_{i-1}]^\top$, we define the matrix

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{c_1}^\top \\ \vdots \\ \boldsymbol{c_d}^\top \end{bmatrix} = \begin{bmatrix} w_1 & w_2 & \cdots & w_{d-1} & w_d \\ w_2 & w_3 & \cdots & w_d & w_1 \\ \vdots & \ddots & \cdots & \ddots & \vdots \\ w_d & w_1 & \cdots & w_{d-2} & w_{d-1} \end{bmatrix}. \tag{4}$$

Since

$$E(\boldsymbol{c_i}^\top \boldsymbol{c_j}) = \{ \begin{array}{ll} \gamma^2 d & (i=j) \\ 0 & (i \neq j) \end{array} \tag{5}$$

we have the relation

$$E[|\boldsymbol{y}|_2^2] = E[\sum_{i=1}^{k}(\boldsymbol{c_i}^\top\boldsymbol{x})^2]$$

$$= \sum_{i=1}^{k}\boldsymbol{x}^\top E[\boldsymbol{c_i}\boldsymbol{c_i}^\top]\boldsymbol{x} = \sum_{i=1}^{k}\boldsymbol{x}^\top\gamma^2\boldsymbol{I}\boldsymbol{x} = k\gamma^2|\boldsymbol{x}|^2. \tag{6}$$

We set $\gamma = 1/\sqrt{k}$. Using discrete Fourier transform, the circular convolution is efficiently computed. Even if distances are preserved, Lemma 1 is not ensured for random set $\boldsymbol{X} \subset \mathbf{R}^d$.

Spread-spectrum is a method by which a signal generated in a particular band-

Algorithm1 Preparation of random projection

Input : d: former dimension;
k: target dimension;
$\boldsymbol{w} = [w_1, \ldots, w_d]^\top$, $w_i(i = 1, \ldots, d)$
is satisfied with independent distribution of
standard deviation $\frac{1}{\sqrt{k}}$ and
the mean of w_i is 0;
$\hat{\boldsymbol{w}} = \mathcal{F}[w]$:Fourier transform
$s = [s_1, \ldots, s_d]^\top$,
$s_i(i = 1, \ldots, d) \in \{+1, -1\}$;
$\boldsymbol{L} = l_1, \ldots, l_k \subset 1, \ldots, d$,
all elements of L is not duplicate;
Output : $\hat{\boldsymbol{w}}, \boldsymbol{s}, \boldsymbol{L}$.

width is deliberately spread in the frequency domain, resulting in a signal with a wider bandwidth. The direct-sequence spread spectrum (DSSS) is a modulation technique. As with other spread spectrum technologies, the transmitted signal preserves more bandwidth than the information signal that is being modulated. Therefore, we can use the direct-sequence spread spectrum method to transform $\mathcal{F}[\boldsymbol{x}]$ to dense vector.

Let $\boldsymbol{s} = [s_1, \ldots, s_d]^\top$ be a set of independent stochastic columns such that $E[\boldsymbol{s}] = 0$ and $E[\boldsymbol{s}\boldsymbol{s}^\top] = \sigma^2 \boldsymbol{I}$. We use $\boldsymbol{s}$ to transform $\boldsymbol{x}$ to be dense. Therefore, a dense vector $\boldsymbol{\zeta}$ is computed as

$$\boldsymbol{\zeta} = \boldsymbol{s} \odot \boldsymbol{x} = \mathrm{diag}(\boldsymbol{s})\boldsymbol{x}, \tag{7}$$

where $[\boldsymbol{s}]$ is the diagonal matrix whose elements are $\{s_i\}_{i=1}^d$. Then, we compute

$$\boldsymbol{y} = [\eta_1, \ldots, \eta_k]^\top, \ \boldsymbol{\eta} = \boldsymbol{C}\boldsymbol{\zeta} = \boldsymbol{C}[\boldsymbol{s}]\boldsymbol{x} \tag{8}$$

The expectation of norm is

$$E[|\boldsymbol{y}|_2^2] = k\gamma^2\sigma^2|x|_2^2. \tag{9}$$

To preserve $E[|\boldsymbol{y}|_2^2] = |\boldsymbol{x}|_2^2$, for $\gamma = 1/\sqrt{k}$ and $\sigma = 1$. Therefore, $\boldsymbol{X}$ is projected to $\boldsymbol{Y}$ using O(d) memory area and $\mathcal{O}(nd\log d)$ calculation time.

Using efficient random projection, we propose an algorithm for image registration. The algorithm can not only save the computing cost, also register images accurately. The flow charts are shown in Fig. 3.

Algorithm2 Efficient random projection

Input : $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbf{R}^d\}$,
$\hat{\boldsymbol{w}}, \boldsymbol{s}, \boldsymbol{L}$: the set from Algorithm 1;
for $\forall \boldsymbol{x_i} \in \boldsymbol{X}$ do
  $\boldsymbol{\zeta} = \boldsymbol{s} \odot \boldsymbol{x_i}$;
  $\boldsymbol{\eta} = \mathcal{F}^{-1}[\hat{\boldsymbol{w}} \odot \mathcal{F}[\boldsymbol{\zeta}]^*]$;
  $\boldsymbol{y_i} = [\eta_{l_1}, \ldots, \eta_{l_k}]^\top$;
end for
Output : $\boldsymbol{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \in \mathbb{R}^k\}$

D	40000	16384	10000	6400	4096
$Time_{RP}$(s)	51.59	20.91	12.80	8.19	5.24
$Time_{ERP}$(s)	0.017	0.017	0.016	0.016	0.017
E_{result}	34187	34132	34418	34148	34322
$E_{Original}$	34328	34328	34328	34328	34328
D	2500	1024	400	100	25
$Time_{RP}$(s)	3.21	1.32	0.52	0.13	0.03
$Time_{ERP}$(s)	0.017	0.016	0.017	0.017	0.017
E_{result}	34434	34519	32781	33742	31066
$E_{Original}$	34328	34328	34328	34328	34328

Table 1 Computational Comparision: $Time_{RP}$: Computation time of basic random projection contains the time of choosing random matrix and performing random projection. $Time_{ERP}$: Computation time of efficient random projection contains the time of preparation andperforming random projection. E_{result}: Energy of result images. $E_{Original}$: Energy.

4. Numerical Examples

Using efficient random projection, we propose an algorithm for image registration. The algorithm can not only save the computing cost, also register images

Projected dimension	100000	10000	1000	100	10
d	816.8	816.8	816.8	816.8	816.8
d_i	818.6	819.9	818.5	807.5	673.4

Table 2 Pairwise distance

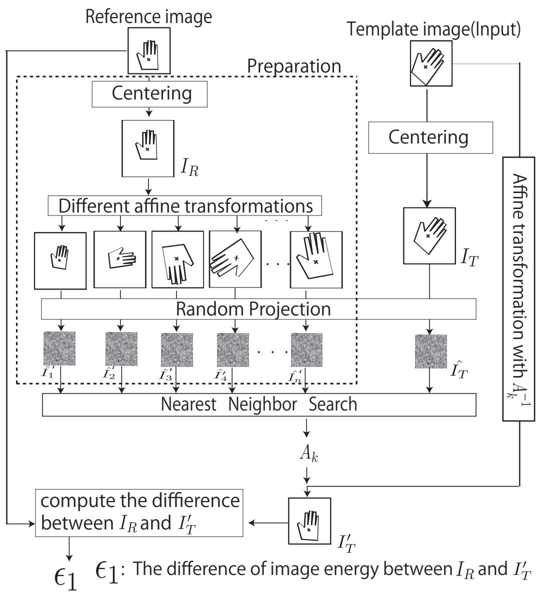


Fig. 3 Algorithm Flow

accurately. The flow charts are shown in Fig. 3.

We first evaluate performance of our random projection algorithm These results show that the computational time of basic random projection is be directly proportional to the projected dimension number, although the computational time of efficient random projection is always less than the time of basic random projection and it is just change more or less. About the energy of result images, Except energy of the result image that projected to 25-dimensional space, the energy is approximately preserved during the projection. The one of 25-dimensional space is fail, because of the projected dimension is too small($25 \ll 65536$). The computational time of efficient random projection is not consider to the projected dimension number, so there is no necessary to choose a very small projected dimensional space. The results show that if the projected dimension number is not too small, pairwise distance will be aproximately preserved during the projection.

Next, we show the registration results under verious dimensionality reduction.

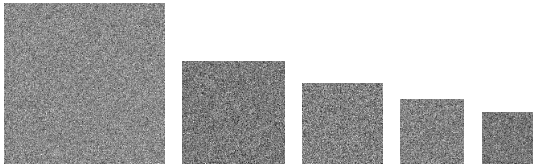
Algorithm

Input : $\boldsymbol{I_R}$: Reference Image,
$\boldsymbol{I_T}$: Template Image
Output : $\boldsymbol{\hat{I}_T}$

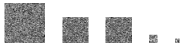
1. Enlarge the size of the reference image and the template image to the same size. Then, coordinate the barycenter of reference image and template image to the center of the image domain.
2. Transform centered reference image $\boldsymbol{I_R}$ to n images $\boldsymbol{I_1}, \ldots, \boldsymbol{I_n}$ by using affine transformation. ($\boldsymbol{A_i}, i = (1, \ldots, n)$: affine matrix)
3. Do efficient random projection on images of 2. ($\boldsymbol{\hat{I}_1}, \ldots, \boldsymbol{\hat{I}_n}$)
4. Do efficient random projection on centered template image $\boldsymbol{I_T}$ with the same random matrix as 2 used. ($\boldsymbol{I_{T-r}}$)
5. Find the most approximate one $\boldsymbol{\hat{I}_k}$ from $\boldsymbol{\hat{I}_i}$ by using nearestneighbor search. The corresponding affine matrix is $\boldsymbol{A_k}$.
6. Using inserse affine transformation, $\boldsymbol{\hat{I}_T} = \boldsymbol{A_k^{-1}} \boldsymbol{I'_T}$.

Figure 5 shows results of registration under verious dimensionality reduction in random projection. Eq. (10) shows the estimated affine transformations under verious dimensionality reduction in random projection.

$$\boldsymbol{A_{16384}} = \boldsymbol{A_{10000}} = \boldsymbol{A_{6400}} = \boldsymbol{A_{4096}} = \begin{pmatrix} 0.9135 & -0.4067 & 0 \\ 0.4067 & 0.9135 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



(a) D_{40000} (b) D_{16384} (c) D_{10000} (d) D_{6400} (e) D_{4096}



(f) D_{2500} (g) D_{1024} (h) D_{400} (i) D_{100} (j) D_{25}

Fig. 4 Rrandom projections of Images

Template	Running time of ERP	Running time of parameters estimation
Image 1	339.303s	0.319s
Image 2	340.268s	0.317s

Table 3 Computational Times

$$\boldsymbol{A_{2500}} = \boldsymbol{A_{1024}} = \begin{pmatrix} 0.9205 & -0.3907 & 0 \\ 0.3907 & 0.9205 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (10)$$

$$\boldsymbol{A_{25}} = \begin{pmatrix} 1.3661 & 0.4177 & 0 \\ -0.4177 & 1.3661 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Finally, we evaluate performance of the algorithm for different template.

The computational times are shown in table 3 and the estimated parameters are.

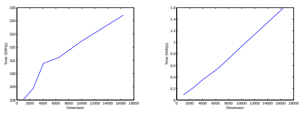
(a) Reference (b) Template (c) Result: to 16384 dimension

(d) Result: to 10000 dimension (e) Result: to 6400 dimension (f) Result: to 4096 dimension

(g) Result: to 2500 dimension (h) Result: to 1024 dimension (i) Result: to 25 dimension

Fig. 5 Result images for verious dimensionality reduction with 256×256 pixel-resolution.

$$\boldsymbol{A1} = \begin{pmatrix} 0.9135 & -0.4067 & 0 \\ 0.4067 & 0.9135 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \boldsymbol{A2} = \begin{pmatrix} 0.7697 & 0.9850 & 0 \\ -0.9850 & 0.7697 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (11)$$



(a) Relationship between dimension and Running Time for ERP
(b) Relationship between dimension and Time for NNS

Fig. 6 Running times for projected dimnsion.

5. Conclusions

In this paper, using the efficient random projection, we developed an efficient algorithm that establishes global image registration. We introduced to use spectrum spreading and circular convolution to reduce computational cost of random projection.

Extensions of the method to range images and 3D volumetric images are strighforward.

This research was supported by "Computational anatomy for computer-aided diagnosis and therapy: Frontiers of medical image sciences" funded by Grant-in-Aid for Scientific Research on Innovative Areas, MEXT, Japan, Grants-in-Aid for Scientific Research founded by Japan Society of the Promotion of Sciences and Grant-in-Aid for Young Scientists (A), NEXT, Japan.

(a) I_R
(b) I_T1
(c) Result Image of template image 1

(d) I_R
(e) I_T2
(f) Result Image of template image 2

Fig. 7 Result image $resolution = 256 \times 256$ pixels

References

1) , S. S. Vempala, *The Random Projection Method*, DIMACS, 65, (2004).
2) D. M. Healy, G. K. Rohde, Fast global image registration using random projection, 476-479, Proc. Biomedical Imaging: From Nano to Macro, (2007).
3) T. Sakai", An efficient algorithm of random projection by spectrum spreading and circular convolution, Inner Report IMIT Chiba University, (2009).
4) T. Sakai, A. Imiya, Practical algorithms of spectral clustering: Toward large-scale vision-based motion analysis, 2011. in L. Wang, G. Zhao, L.Cheng, M. Pietikäinen eds. *Machine Learning for Vision-Based Motion Analysis Theory and Techniques*, Advances in Pattern Recognition, Springer, (2011).
5) B. Zitová, J. Flusser, Image registration methods: A Survey, Image Vision and Computing **21**, 977-1000, (2003).
6) J. Modersitzki, *Numerical Methods for Image Registration*, CUP, (2004).
7) W. Johnson, J. Lindenstrauss, Extensions of Lipschitz maps into a Hilbert space, Contemporary Mathematics, **26**, 189-206, (1984).
8) P. Frankl, H. Maehara, The Johnson-Lindenstrauss lemma and the sphericity of some graphs", Journal of Combinatorial Theory Series A, **44**, 355-362, (1987).

Appendix

A. The Nearest Neighbor Search

The nearest neighbor search (NNS) problem is the following: Given a set of n points $\boldsymbol{P} = p_1, \ldots, p_n$ in a metric space $\boldsymbol{X}$, preprocess $\boldsymbol{P}$ closest to a query point $q \in \boldsymbol{X}$. The two most well-known methods are the k-nearest neighbor search and the ε-approximate nearest neighbor search.

(1) Near neighbour (range search): find one/all points in $\boldsymbol{P}$ within distance r from q.

(2) Spatial join: given two sets $\boldsymbol{P}, \boldsymbol{Q}$, and find all pairs $p \in \boldsymbol{P}$, $q \in \boldsymbol{Q}$, such that p is within distance r from q.

(3) Approximate near neighbor (ANN): find one/all points p in $\boldsymbol{P}$, whose distance to q is at most $(1+e)$ times of the distance from q to its nearest neighbor.

In the algorithm of this page, we have used the approximate nearest neighbours(ANN) which is: Find a point $p \in \boldsymbol{P}$ that is an ε-approximate nearest neighbor of the query q in that for all $p' \in \boldsymbol{P}, d(p,q) \leq (1+\epsilon)d(p',q)$.

5.1 B. Image Compression by Random Projection

By expressing an image, for instance an 256×256, as the matrix $\boldsymbol{A}$ such that

$$\boldsymbol{A} = (\boldsymbol{a_1}, \boldsymbol{a_1}, \ldots, \boldsymbol{a_{256}}) \tag{12}$$

where $\boldsymbol{A} \in \mathbf{R}^{256\times256}$ and $\boldsymbol{a_i} \in \mathbf{R}^{256}$ we first transform $\boldsymbol{A}$ to $\boldsymbol{U}$,

$$\boldsymbol{u} \in \mathbf{R}^{65536}, \boldsymbol{u} = \begin{pmatrix} \boldsymbol{a_1} \\ \vdots \\ \boldsymbol{a_{256}} \end{pmatrix}. \tag{13}$$

For a random projection matrix $\boldsymbol{R} \in \mathbf{R}^{1024\times65536}$ which satisfies the relation $N(0,1)$, we compute

$$\boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{1024} \end{pmatrix}, \quad \boldsymbol{v} = \sqrt{\frac{1}{1024}} \boldsymbol{R}^\top \boldsymbol{u}, \tag{14}$$

and transform $\boldsymbol{v}$ to $\boldsymbol{B}$ such that

$$\boldsymbol{B} \in \boldsymbol{R}^{32\times32}, \boldsymbol{B} = \begin{pmatrix} v_1 & v_{33} & v_{65} & \cdots & v_{993} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ v_{32} & v_{64} & v_{96} & \cdots & v_{1024} \end{pmatrix}. \tag{15}$$

Then, we can have compressed image $\boldsymbol{B}$ from $\boldsymbol{A}$.

C. Compression by The Pyramid Transform

The pyramid transform reduces the size of signals and images with preserving global appearances of them. For the discrete signal $\boldsymbol{u} = (u_1, u_2 \ldots, u_N)^\top$ the pyramid transform of the factor 2

$$v_n = w_{2n}, \quad w_n = \frac{1}{4}(u_{n-1} + 2u_n + u_{n+1}).$$

is expressed in the matrix form as

$$\boldsymbol{R} = \boldsymbol{DG},\ \boldsymbol{D} = \boldsymbol{I} \otimes \boldsymbol{e}_2^\top,\ \boldsymbol{e}_2^\top = (0,1)^\top,\ \boldsymbol{G} = \frac{1}{4} \begin{pmatrix} 1 & 1 & 0 & \ldots & 0 & 0 \\ 1 & 2 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 1 & 1 \end{pmatrix}.$$

with the von-Neumann boundary condition. Furthermore, the pyramid transform of factor k is expressed as $\boldsymbol{R}^{k-1}$. Furthermore, for discrete images, $\boldsymbol{R}$ is defined as

$$\boldsymbol{R} = \frac{1}{2}(\boldsymbol{DG} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{DG}),$$

where $\boldsymbol{A} \otimes \boldsymbol{B}$ is the Kroneker product of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$.

Since $\rho(\boldsymbol{R}) \leq 1$, we have the relation

$$|\boldsymbol{Ru}_1 - \boldsymbol{Ru}_2| \leq |\boldsymbol{u}_1 - \boldsymbol{u}_2|.$$

This relation leads to the conclusion that the pyramid transform does not preserve distance properties. Moreover, since $(\boldsymbol{Ru}_1, \boldsymbol{Ru}_2) = (\boldsymbol{R}^\top \boldsymbol{Ru}_1, \boldsymbol{u}_2)$, generally

$$\frac{(\boldsymbol{Ru}_1, \boldsymbol{Ru}_2)}{|\boldsymbol{Ru}_1||\boldsymbol{Ru}_2|} \neq \frac{(\boldsymbol{u}_1, \boldsymbol{u}_2)}{|\boldsymbol{u}_1||\boldsymbol{u}_2|}.$$

Therefore, the pyramid transform does not generally preserve angle properties.