Bernd Hamann

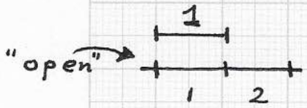
Stratovan

10/25/2021

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• **Laplacian eigenfunctions:**

Lixel eigenfunctions:



$$\begin{bmatrix} -2 & 1 \\ 1 & -2 \end{bmatrix},$$

$\mathbb{\lambda} = (\lambda_1, \lambda_2) = (-3, -1)$,

$\mathbb{e}_1^n = (-1, 1)^T / \sqrt{2}$,

$\mathbb{e}_2^n = (1, 1)^T / \sqrt{2}$.

– Expand $\mathbb{f} = (2,4)^T$ in orthonormal basis $\{\mathbb{e}_i^n\}_{i=1}^2$:

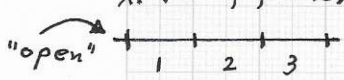
$\mathbb{f} = \sum_{i=1}^2 c_i \mathbb{e}_i^n$,

$c_1 = \langle \mathbb{f}, \mathbb{e}_1^n \rangle = \sqrt{2}$,

$c_2 = \langle \mathbb{f}, \mathbb{e}_2^n \rangle = 3\sqrt{2}$.

$\Rightarrow$ Feature vector $= (c_1, c_2) = (\sqrt{2}, 3\sqrt{2})$.

$\mathbb{\lambda} = (-2-\sqrt{2}, -2, -2+\sqrt{2})$



$$\begin{bmatrix} -2 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -2 \end{bmatrix},$$

$\mathbb{e}_1^n = (1, -\sqrt{2}, 1)^T / 2$,

$\mathbb{e}_2^n = (-1, 0, 1)^T / \sqrt{2}$,

$\mathbb{e}_3^n = (1, \sqrt{2}, 1)^T / 2$.

– Expand $\mathbb{f} = (2,4,8)^T$ in orthonormal basis $\{\mathbb{e}_i^n\}_{i=1}^3$:

$\mathbb{f} = \sum_{i=1}^3 c_i \mathbb{e}_i^n$,

...

The discussion of eigenfunctions makes it possible to consider (at least) two fundamental methods for defining and computing a multi-scale "fingerprint," "signature," feature vector of a 3D voxel segment:

1) Considering a subset(s) of a 3D voxel segment, one can determine an eigenfunction basis for this (these) subset(s), based purely on the voxel mesh and ignoring the mass values, and then expand the segment's (segments') mass function(s) in the eigenfunction basis. In other words, a 3D voxel segment with associated (mass) function value $\mathbb{f}$ would be represented in a basis $\{\mathbb{e}_i^n\}_{i=1}^N$ – with N being the number of voxels and thus voxel-associated mass values – leading to the **expansion** $\mathbb{f} = \sum_{i=1}^N c_i \mathbb{e}_i^n$. The coefficient set $\{c_i\}_{i=1}^N$ becomes the segment's multi-scale feature vector. **!**

2) Given a 3D voxel segment, one computes $M^{-1}K$ – with M being a mass matrix and K being a stiffness matrix – for the entire segment or a segment subset(s).

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• *Laplacian eigenfunctions:*

...

$c_1 = \langle \mathbf{f}, \mathbf{e}_1^n \rangle = 5-2\sqrt{2}$,

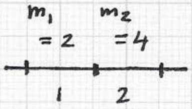
$c_2 = \langle \mathbf{f}, \mathbf{e}_2^n \rangle = 3\sqrt{2}$,

$c_3 = \langle \mathbf{f}, \mathbf{e}_3^n \rangle = 5+2\sqrt{2}$.

$\Rightarrow$ Feature vector

$= (c_1, c_2, c_3) = (5-2\sqrt{2}, 3\sqrt{2}, 5+2\sqrt{2})$.

Analysis based on $M^{-1}K$:



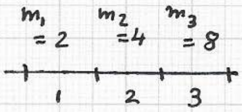
$$M^{-1}K = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/4 \end{bmatrix} \begin{bmatrix} -2 & 1 \\ 1 & -2 \end{bmatrix} = \begin{bmatrix} -1 & 1/2 \\ 1/4 & -1/2 \end{bmatrix},$$

$\lambda = (-3-\sqrt{3}, -3+\sqrt{3})/4$

$=$ Feature vector.

$\{ \mathbf{e}_1^n = (-1-\sqrt{3}, 1)^T / (5+2\sqrt{3})$,

$\mathbf{e}_2^n = (-1+\sqrt{3}, 1)^T / (5-2\sqrt{3})$. $\}$



$$M^{-1}K = \begin{bmatrix} -1 & 1/2 & 0 \\ 1/4 & -1/2 & 1/4 \\ 0 & 1/8 & -1/4 \end{bmatrix},$$

$\lambda = (-1.19, -0.44, -0.12)$

$=$ Feature vector.

$\{ \mathbf{e}_1^n = \ldots, \mathbf{e}_3^n = \ldots . \}$

The matrix $M^{-1}K$ (or $-M^{-1}K$) "captures the eigenfrequency behavior" of a 3D set of voxels with associated mass values. Thus, the eigenvalues of $M^{-1}K$ directly define a 3D voxel segment's eigenfrequencies – when viewing the segment as a mass-spring system that oscillates, could potentially oscillate. Therefore, one can also define a multi-scale feature vector based on the eigenvalues λ_i (or eigenfrequencies ω_i) of $M^{-1}K$. **!**

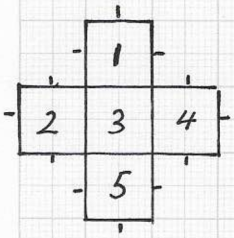
• **Note:** If one were to apply method 2) to an entire – possibly very large – 3D voxel segment, one must keep in mind that (i) M^{-1} has non-zero values (mass m_i-values) only on its diagonal and m_i-values are "close in value" (not always!) as they belong to the same segment, and (ii) K has the value -6 on its diagonal (6 being the number of a voxel's face neighbors) and maximally 7 non-zero values per row.

Via bandwidth minimization $M^{-1}K$ can be stored and analyzed efficiently.

Stratovan

▪ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• **Laplacian eigenfunctions:**



$$\begin{bmatrix} -4 & 0 & 1 & 0 & 0 \\ 0 & -4 & 1 & 0 & 0 \\ 1 & 1 & -4 & 1 & 1 \\ 0 & 0 & 1 & -4 & 0 \\ 0 & 0 & 1 & 0 & -4 \end{bmatrix},$$

$$\lambda = (-6, -4, -4, -4, -2),$$

$$\begin{aligned} \mathbb{e}_1^n &= (1, 1, -2, 1, 1)^T / 2\sqrt{2}, \\ \mathbb{e}_2^n &= (-1, 0, 0, 0, 1)^T / \sqrt{2}, \\ \mathbb{e}_3^n &= (-1, 0, 0, 1, 0)^T / \sqrt{2}, \\ \mathbb{e}_4^n &= (-1, 1, 0, 0, 0)^T / \sqrt{2}, \\ \mathbb{e}_5^n &= (1, 1, 2, 1, 1)^T / 2\sqrt{2}. \end{aligned}$$

Data defining the 5-pixel "open" template when used for expanding a 5-pixel segment function $\mathbb{f}$ via the template's multi-scale eigenfunctions $\mathbb{e}_i^n$. Note: $\{\mathbb{e}_i^n\}$ is not an orthogonal basis!

Expand $\mathbb{f}$ as $\mathbb{f} = \sum_{i=1}^{5} c_i \mathbb{e}_i^n$;

solve normal equations:

$$\left(\langle \mathbb{e}_i^n, \mathbb{e}_j^n \rangle \right) \begin{pmatrix} c_1 \\ \vdots \\ c_5 \end{pmatrix} = \begin{pmatrix} \vdots \\ \langle \mathbb{f}, \mathbb{e}_i^n \rangle \\ \vdots \end{pmatrix}$$

$$\Leftrightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1/2 & 1/2 & 0 \\ 0 & 1/2 & 1 & 1/2 & 0 \\ 0 & 1/2 & 1/2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} \langle \mathbb{f}, \mathbb{e}_1^n \rangle \\ \vdots \\ \langle \mathbb{f}, \mathbb{e}_5^n \rangle \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3/2 & -1/2 & -1/2 & 0 \\ 0 & -1/2 & 3/2 & -1/2 & 0 \\ 0 & -1/2 & -1/2 & 3/2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \langle \mathbb{f}, \mathbb{e}_1^n \rangle \\ \vdots \\ \langle \mathbb{f}, \mathbb{e}_5^n \rangle \end{bmatrix}$$

MASS FUNCTION $\mathbb{f} = (m_1, m_2, m_3, m_4, m_5)^T$

• • •

Method 1 can be adapted to expand MULTIPLE (mass) functions $\mathbb{f}$ defined by sets of segment subsets, all having the same underlying template, in terms of an eigenfunction basis $\{\mathbb{e}_i^n\}$. For example, the 5-pixel template shown in the figure (left) can be used to expand subsets of a "large" 2D pixel segment, as long as the subsets are locally defined over the shown 5-pixel structure – and all five of a template's pixels belong to / lie inside the segment to be analyzed.

The objective is to design an algorithm based on method 2 that is efficient. The principle of such a design is the (pre-) computation of a "filter mask" that can be applied directly to a (subset of a) 2D segment. This principle is explained via an example shown on this and the next page (left). A 5-pixel-based template is used; one computes its eigenvalues and normalized – but not orthogonal – eigenfunctions, generating the basis $\{\mathbb{e}_i^n\}_{i=1}^5$.

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions:

...

$$\begin{bmatrix} \langle \mathbb{f}, \mathbb{e}_1^n \rangle \\ \langle \mathbb{f}, \mathbb{e}_2^n \rangle \\ \langle \mathbb{f}, \mathbb{e}_3^n \rangle \\ \langle \mathbb{f}, \mathbb{e}_4^n \rangle \\ \langle \mathbb{f}, \mathbb{e}_5^n \rangle \end{bmatrix} = \begin{bmatrix} (m_1 + m_2 - 2m_3 + m_4 + m_5) / 2\sqrt{2} \\ (-m_1 + m_5) / \sqrt{2} \\ (-m_1 + m_4) / \sqrt{2} \\ (-m_1 + m_2) / \sqrt{2} \\ (m_1 + m_2 + 2m_3 + m_4 + m_5) / 2\sqrt{2} \end{bmatrix}$$

$\Rightarrow$

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} m_1 + m_2 - 2m_3 + m_4 + m_5 \\ -m_1 - m_2 \quad - m_4 + 3m_5 \\ -m_1 - m_2 \quad + 3m_4 - m_5 \\ -m_1 + 3m_2 \quad - m_4 - m_5 \\ m_1 + m_2 + 2m_3 + m_4 + m_5 \end{bmatrix} \cdot \frac{\sqrt{2}}{4}$$

$$= \frac{\sqrt{2}}{4} \begin{bmatrix} 1 & 1 & -2 & 1 & 1 \\ -1 & -1 & 0 & -1 & 3 \\ -1 & -1 & 0 & 3 & -1 \\ -1 & 3 & 0 & -1 & -1 \\ 1 & 1 & 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \end{bmatrix}$$

! "MASK"

{ **TEST:** Given $\mathbb{f} = (m_1, m_2, m_3, m_4, m_5)^T$, determine whether $\mathbb{f}$ is correctly expanded in basis $\{\mathbb{e}_i^n\}$ when using this coefficient vector $\mathbb{c} = (c_1, c_2, c_3, c_4, c_5)^T$ in $\sum_{i=1}^5 c_i \mathbb{e}_i^n$: INDEED, $c_1 \mathbb{e}_1^n + \ldots + c_5 \mathbb{e}_5^n$ yields $\mathbb{f}$. }

$\Rightarrow$ **Expanding $\mathbb{f}$ in a 5-pixel-based basis of eigenfunctions $\mathbb{e}_i^n$ can be done via a MASK.**

A "5-pixel mass function $\mathbb{f}$," $\mathbb{f} = (m_1, m_2, m_3, m_4, m_5)^T$, must be expanded in the template's eigenbasis $\{\mathbb{e}_i^n\}$. Thus, one must compute the coefficients c_i of $\mathbb{f}$'s expansion in $\{\mathbb{e}_i^n\}$: $\mathbb{f} = \sum_{i=1}^5 c_i \mathbb{e}_i^n$. The NORMAL EQUATIONS define the linear system of equations one must solve to obtain the c_i-values. (Generally, such a system will have a block-diagonal structure, or can be transformed to such a structure, permitting efficient system matrix inversion!) It is possible to set up this problem symbolically – with the goal of generating a symbolic solution that one can eventually use for direct, IMMEDIATE, computation of c_i-values for given specific pixel mass values m_i, $i = 1 \ldots 5$. Such a symbolic solution is derived in the example; the solution is the **MASK / FILTER MASK / FILTER MATRIX MASK** shown here.

■ OBJECT AND MATERIAL EIGENFUNCTIONS — Cont'd.

• **Laplacian eigenfunctions:** In 2D (pixel) and 3D (voxel) image processing, filters/filter masks are often represented by depicting the specific local region/template to which they are applied — and listing the specific pixel (voxel) weights one must use in a linear combination of the respective pixel (voxel) values to obtain the quantity one must compute; this quantity is usually computed for and associated with the "center of the template." The (weighted) linear combinations of given mass values m_i, $i=1 \ldots 5$, are provided here (left), to compute the coefficents c_i of a 5-pixel-based expansion $\mathbf{f} = \sum_{i=1}^{5} c_i \mathbf{e}_i^n$. Pre-computing the weights for such linear combinations becomes very important when using local pixel-/voxel-neighborhoods consisting of "large(r) numbers" of pixels/voxels. Denoting the number of pixels/voxels by N, this number should be sufficiently large enough to capture the multi-scale nature of a function $\mathbf{f}$ that must be classified.

c_1

	m_1: 1		
m_2: 1	m_3: -2	m_4: 1	$\cdot \frac{\sqrt{2}}{4}$
	m_5: 1		

c_2

	-1		
-1	0	-1	$\cdot \frac{\sqrt{2}}{4}$
	3		

c_3

	-1		
-1	0	3	$\cdot \frac{\sqrt{2}}{4}$
	-1		

c_4

	-1		
3	0	-1	$\cdot \frac{\sqrt{2}}{4}$
	-1		

c_5

	1		
1	2	1	$\cdot \frac{\sqrt{2}}{4}$
	1		

Commonly used depiction of filter (convolution) mask: Values shown in the 5-pixel template define "weights" one must apply to compute a specific c_i - value, e.g., $c_1 = \frac{\sqrt{2}}{4} \cdot (m_1 + m_2 - 2m_3 + m_4 + m_5)$.