Graphical Models 75 (2013) 157–176

Contents lists available at SciVerse ScienceDirect

Graphical Models

journal homepage: www.elsevier.com/locate/gmod

Multiscale 3D feature extraction and matching with an application to 3D face recognition ☆

Hadi Fadaifard [a,*], George Wolberg [b], Robert Haralick [c]

[a] Brainstorm Technology LLC (NYC), United States
[b] Department of Computer Science, City College of New York/CUNY, United States
[c] Department of Computer Science, Graduate Center of the City University of New York, United States

ARTICLE INFO

Article history:
Received 6 June 2012
Received in revised form 30 October 2012
Accepted 20 January 2013
Available online 8 February 2013

Keywords:
3D feature extraction
3D shape matching
3D face recognition
Heat equation
Mesh signal processing

ABSTRACT

We present a new multiscale surface representation for 3D shape matching that is based on scale-space theory. The representation, *Curvature Scale-Space 3D (CS3)*, is well-suited for measuring dissimilarity between (partial) surfaces having unknown position, orientation, and scale. The CS3 representation is obtained by evolving the surface curvatures according to the heat equation. This evolution process yields a stack of increasingly smoothed surface curvatures that is useful for keypoint extraction and descriptor computations. We augment this information with an associated scale parameter at each stack level to define our multiscale CS3 surface representation. The scale parameter is necessary for *automatic scale selection*, which has proven to be successful in 2D scale-invariant shape matching applications. We show that our keypoint and descriptor computation approach outperforms many of the leading methods. The main advantages of our representation are its computational efficiency, lower memory requirements, and ease of implementation.

1. Introduction

3D shape matching refers to the process of measuring the amount of dissimilarity between 3D shapes [1]. *Partial* shape matching is considered to be a more difficult subproblem, where the dissimilarity is measured between partial regions on input surfaces, and the relative position, orientation, scale, or extent of the overlap is unknown [2]. The main difficulties faced by 3D surface matching systems are

- *Representation issues:* the arbitrary organization of points in 3D makes the processing of input surfaces more difficult than processing signals in $\mathbb{R}^n$, where the data are generally organized in the form a grid. This and the non-Euclidean geometry of the surface hinder design of efficient matching algorithms.
- *Geometric transformations:* the input surfaces may have arbitrary translation, rotation, and scale. They may also have undergone non-rigid transformations.
- *Non-geometric transformations:* the surfaces may have been perturbed with varying levels of noise, contain topological noise, or have different sampling variations.

A large number of 3D shape matching techniques already exist in the literature [1,3]; each approach generally addresses a subset of the above-mentioned difficulties.

We present a new 3D surface matching approach motivated by the scale-space theory of signals, which was specifically developed to deal with noise and differences in resolution and scale. We propose a surface representation, which is stable against surface noise, and can be used to form discriminative feature vectors useful for establishing correspondences between regions on 3D surfaces. Moreover, as shown in Section 2.4, our representation can be used to extract stable keypoints on 3D surfaces, with the additional capability of associating a neighborhood size with each point. We follow the same naming convention

☆ This paper has been recommended for acceptance by Yücel Yemez and Hao (Richard) Zhang.

* Corresponding author.
E-mail addresses: hfadaifard@gc.cuny.edu (H. Fadaifard), wolberg@cs.ccny.cuny.edu (G. Wolberg).

1524-0703/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.gmod.2013.01.002

as in [4,5], and refer to the process of associating a neighborhood size to an extracted differential feature as *automatic scale selection*. Fig. 1 shows an example of automatic scale selection at a few locations on a 3D model. Note that the detected scale is intrinsic to the surface and does not depend on the spatial scale or sampling resolution of the surface. This notion of scale selection, which arises from the scale-space theory, has been central to the success of many scale-invariant 2D matching approaches.

The scale-space *representation* of a signal in $\mathbb{R}^n$ is defined in terms of the evolution of that signal according to the heat (diffusion) equation. The scale-space *theory* is concerned with the study and analysis of the properties of this representation of signals. The principal motivation for the development of the scale-space theory has been the problem of estimating differential properties of signals, which are required in various vision applications [6]. The sensitivity of differentiation to noise and the question of the size (scale) of the differential operators that are applied to the signals are the two main issues investigated by scale-space theory. The theory has been extensively studied for the case of images (signals) in $\mathbb{R}^n$ [6,7] and has become quite mature and widely used over the past few decades. It has been shown that besides having nice theoretical properties, the scale-space representation of images can be realized efficiently, with impressive practical results [8,9]. Currently, scale-space based matching techniques, such as Scale Invariant Feature Transform (SIFT) [8] and Speeded Up Robust Features (SURF) [10] have become the *de facto* standards in many 2D matching applications. Despite finding widespread use in 2D signal processing, scale-space techniques have not been widely applied to 3D surfaces.

There are two major difficulties with extending scale-space representations to 3D surfaces. These difficulties are representation issues, and lack of a consistent mechanism for estimating the scale parameter necessary for automatic scale selection. The lack of grid-like structures that are present in 2D images and the non-Euclidean geometry of surfaces make development of precise and efficient representations difficult. The scale parameter, which in the case of signals in $\mathbb{R}^n$ is defined in terms of the variance of the smoothing kernel, may not be readily available for 3D surfaces.

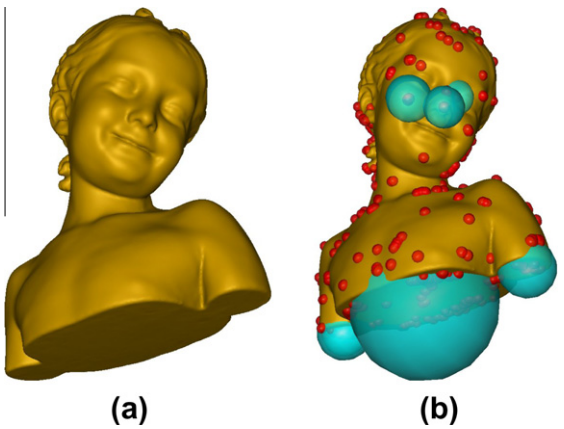


Fig. 1. Automatic scale selection on the Bimba model; (a) original model, (b) estimated scales at a few locations on the model. The red spheres in (b) indicate the locations of the extracted keypoints on the model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The goal of this work is to extend the use of scale-space theory to 3D surfaces for the purpose of partial shape matching. The main contribution is a new scale-space representation for surfaces that addresses the two major difficulties outlined above. The new representation, which we refer to as *Curvature Scale-Space 3D* (*CS3*), is shown to be stable against noise, computationally efficient, and capable of automatic scale selection. We show how our CS3 representation can be used for multiscale 3D keypoint extraction and compare the performance of our proposed keypoint extractor against competing methods. We also show an application of our representation in 3D face recognition, where CS3 is used to form feature vectors (descriptors) for measuring the dissimilarity between 3D faces. The discriminative power of our features is compared against competing methods.

1.1. Related work

Witkin introduced "scale-space filtering" in [6], where it was argued that the extrema of signals and their first few derivatives contain the most relevant information useful in vision applications. It was also argued that the main problem with using these features is estimating the signals' derivatives — more specifically the neighborhood size, known as the *scale*, needed to estimate the derivatives. It was suggested that signals and their derivatives must be investigated/analyzed at all possible scales, *simultaneously*. Witkin identified the Gaussian (and its derivatives) as the unique smoothing kernels most suitable for regularizing the required differentiation operations.

More research was devoted to the scale-space representation for images, which eventually led to the development of the "scale-space theory." The scale-space representation of a signal in $\mathbb{R}^N$ has been formally defined as the solution to the heat (diffusion) equation involving the original signal (see Section 2.1). Other representations such as the Curvature Scale Space of Mokhtarian and Mackworth [11], which is used for representing and matching planar curves, have also been proposed and employed in matching applications.

The most well-known example of the power of the scale-space representation for applications in computer vision is the Scale Invariant Feature Transform (SIFT) of Lowe [8]. SIFT is used for extracting keypoints in images and also computing local descriptor vectors, which are then used for establishing correspondences between images. One of the main attributes of SIFT features is the scale associated with each extracted keypoint, which in turn, gives rise to the invariance of the approach to scale changes between images. The first interesting application of SIFT that contributed to SIFT's fame and success, is the automatic stitching of panoramic images of Brown and Lowe [9]. Despite the success of scale-space theory in 2D computer vision applications, the extensions of the theory to 3D surfaces have not been equally effective. The main difficulties

associated with such an extension are due to surface representation issues, which prevent an *efficient* and *precise* extension to surfaces. As mentioned previously, by representation issues, we refer to the lack of grid-like organization of free-form surface vertices that is present in images, and also the non-Euclidean geometry of 3D surfaces.

The few current scale-space based surface representations can be categorized into two classes based on how a signal is derived from the surface and is consequently evolved. First, surface positions may be treated as the signal and therefore the surface geometry is modified during the evolution process that defines the scale-space representation. Second, a signal may be defined and evolved over the surface while the geometry of the surface remains unchanged. It is well-known that the evolution of surface positions generally leads to geometric problems such as shrinkage and development of cusps and foldovers [12,13]. Therefore, when we define our proposed scale-space representation for surfaces (Section 2.2), we opt for the second approach whereby we define our scale-space representation in terms of the evolution of surface curvatures.

The most straightforward of approaches for extending the scale-space representation of signals to 3D surfaces are perhaps parameterization [14] and voxelization [15]. These two approaches, however, result in new surface representations that suffer from distortions or loss of precision, respectively.

Mean curvature flow, which is closely related to surface diffusion, may also be used to smooth the surface. Under mean curvature flow, each surface point is moved along its normal proportional to its mean curvature. Schlattmann et al. [13] use a modification of this approach to obtain a scale-space representation for surfaces and show how it can be used to perform feature extraction and automatic scale selection on closed 3D models. A major problem with this approach, however, is the geometric degeneracies that generally arise from smoothing. In addition, computation times of more than two hours were reported for meshes with more than 2*K* vertices. A similar approach, which also suffers from the same problems with geometric degeneracies and computation times, is reported in [16].

As mentioned earlier, instead of smoothing the surface geometry, signals defined over the surface may be smoothed. For example, in [17], surface mean curvatures on a triangulated surface are repeatedly smoothed with a Gaussian filter. The proposed representation is then used to define a measure of *mesh saliency* over the surface and its applications in mesh simplification and viewpoint selection are shown in that paper. Similar methods are employed in [18,19].

More recently, the Heat Kernel Signature (HKS) [20] has been used in global shape matching tasks involving 3D models that may have undergone isometric deformations. In this approach, the properties of the heat kernel of a surface are used to infer information about the geometry of the surface. A scale-invariant version of HKS was also introduced in [21] and used for non-rigid 3D shape retrieval. The main drawbacks of HKS-based techniques are computation times and their inability to perform automatic scale selection, which is required in most partial shape matching tasks involving shapes of arbitrary resolution and scale. Even though in [21], much faster computations times are reported for HKS, the models used in their tests are generally low resolution (mostly, in the order of a few hundred vertices). Additionally, they only use the first hundred eigenvalues of the Laplace–Beltrami operator to construct their HKS descriptors. The descriptors constructed in this manner are coarse and more suitable for global shape matching tasks involving objects from *different classes* (e.g., planes, humans, animals, etc.). On the other hand, the discriminative power of the descriptors used in this approach were tested in datasets of objects from the *same class*, namely human faces. A fast multiresolution implementation of HKS has also been proposed in [22]. However, their approach is still computationally more expensive than ours. In [22], for example, the reported computation times for meshes with 100K+ vertices varies from 110 s to 288 s. In addition, the stability of their HKS descriptors with small t's were not tested in a matching application. Our scale-space approach presented in this paper is shown to be more efficient to compute. For instance, we obtain the scale-space representation (with 32 levels) of a surface with 267*K* vertices in 14 s on a single core of a 2.3 GHz CPU (Intel Core i7-2820QM).

In Section 2, we present our proposed scale-space representation for 3D surfaces, and show how it can be used for feature extraction and matching. In Section 2.5, we compare the performance of our keypoint extraction approach against competing methods. In Section 3, we show how our proposed scale-invariant Laplacian of surface (mean) Curvatures can be used to construct stable and yet discriminative feature vectors, which are then used in a 3D face recognition system. We test the performance of our simple PCA-based face recognition system on two well-known 3D face datasets, and compare its performance against current state-of-the-art face recognition systems. Finally, in Section 4, we provide a summary of the work and directions for future work.

2. Scale-space representation for signals in R^n and on 3D surfaces

In Section 2.1, we first present the definition of the scale-space representation for signals in $\mathbb{R}^n$. In Section 2.2, we present our proposed extension of the representation to 3D surfaces for the purpose of shape matching.

2.1. Scale-space representation of signals in $\mathbb{R}^n$

The scale-space representation of a continuous signal $f:\mathbb{R}^n \to \mathbb{R}$ is defined as the solution to the heat (diffusion) equation [7]:

$$\partial_t F = \Delta F, \tag{1}$$

with the initial condition $F(\mathbf{x};0) = f(\mathbf{x})$; Δ denotes the Laplacian. It can be shown that the Gaussian is the fundamental solution to Eq. (1) [7]. The scale-space representation of f can therefore be expressed as

$$F(\mathbf{x};t) = g(\mathbf{x};t) * f(\mathbf{x}), \tag{2}$$

where $*$ denotes convolution, $g : \mathbb{R}^n \to \mathbb{R}$ is the n-dimensional normalized Gaussian: $g(\mathbf{x}; t) = \frac{1}{(\pi t)^{n/2}} e^{-\|\mathbf{x}\|^2/t}$, and t is known as the *scale parameter*.

The non-enhancement property [7] of the scale-space representation of signals, in general, guarantees that the values of the local maxima (minima) decrease (increase) as the signal is smoothed. As shown in [7], the scale-normalized amplitudes of the spatial derivatives of F shall be useful in inferring the size of structures in f. Normalization is achieved by using the change of variable $\mathbf{v} = \frac{\mathbf{x}}{t^{\gamma/2}}$, for $\gamma > 0$. This results in the following scale-normalized spatial derivatives of the scale-space representation of the signal:

$$\partial_{\mathbf{v}^{\mathbf{m}}} F^*(\mathbf{v}; t) = t^{|\mathbf{m}|\gamma/2} \partial_{\mathbf{x}^{\mathbf{m}}} F(\mathbf{x}; t), \tag{3}$$

where $\mathbf{m} = (m_1, \ldots, m_n)$ and $\partial_{\mathbf{x}^{\mathbf{m}}}$ constitute the multi-index notation for partial derivatives; $|\mathbf{m}| = m_1 + \cdots + m_n$ denotes the *order* of the multi-index. The normalized derivatives are no longer strictly decreasing or increasing. Instead, they may assume local extrema over scales. The scale selection principle [7] states that the scale at which these normalized derivatives assume a local maximum reflects a characteristic size of a corresponding structure in the data. The process of finding this scale, known as *automatic scale selection*, has been successfully employed by approaches such as SIFT [8] to achieve scale invariance in matching applications. We seek the same type of scale-normalization in a scale-space representation of a surface signal, and employ it to infer information about the size of structures on the surface.

2.2. Scale-space representation for 3D surfaces

In this section, we formulate a similar representation for surfaces that is as close as possible to the scale-space representation of signals in $\mathbb{R}^n$. Our proposed approach is similar to the HKS-based techniques, in the sense that we derive the scale-space formulation of the surface in terms of the evolution (diffusion) of signals on the surface with the help of the Laplace–Beltrami operator. However, we analyze the surface structures by directly studying the behavior of the signal as it evolves on the surface. More specifically, we take the signal to be the surface curvatures, which are derived from the surface geometry. Operating in the curvature domain is a natural choice since all the relevant geometric information about a surface is encoded in its principal directions and curvatures. The main advantages of our approach over HKS are gains in computational efficiency and the ability to estimate the size of the surface structures. Additionally, our representation enables us to robustly and efficiently estimate the Laplacian of surface curvatures that results in a rich set of features, which is useful in subsequent matching tasks. It has also been shown [17] that features extracted in this manner are *salient* and meaningful to the human eye.

Therefore, the scale-space representation, $F : \mathcal{M} \times \mathbb{R} \to \mathbb{R}$, of 3D surface $\mathcal{M}$, is defined as the solution to the diffusion equation:

$$\partial_t F = \Delta_{\mathcal{M}} F, \tag{4}$$

with the initial condition $F(\mathbf{p}; 0) = f(\mathbf{p})$, where $f(\mathbf{p})$ denotes the mean or Gaussian curvature at point $\mathbf{p} \in \mathcal{M}$, and $\Delta_{\mathcal{M}}$ is the Laplace–Beltrami operator.

From the above formulation, a stack of Gaussian-smoothed surface curvatures is obtained that can be used directly in multiscale feature extraction and descriptor computations. However, to make the best use of the representation for automatic scale selection, the value of the scale parameter at each level must also be estimated. The smoothed curvatures together with the associated scales at each level define our multiscale surface representation, which we refer to as the *Curvature Scale-Space 3D* (CS3), as depicted in Fig. 2.

In Section 2.3, we describe how a discrete surface signal may be efficiently smoothed in a manner consistent with the scale-space representation of signals. In Section 2.4, we show how the representation may be used for feature point (keypoint) extraction with an automatic scale selection mechanism.

2.3. Gaussian smoothing a discrete surface signal

Let our discrete surface be represented by the polygonal mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_N\}$, and $\mathcal{E} = \{e_{ij} | v_i \text{isconnectedto} v_j\}$ are the vertex and edge sets,

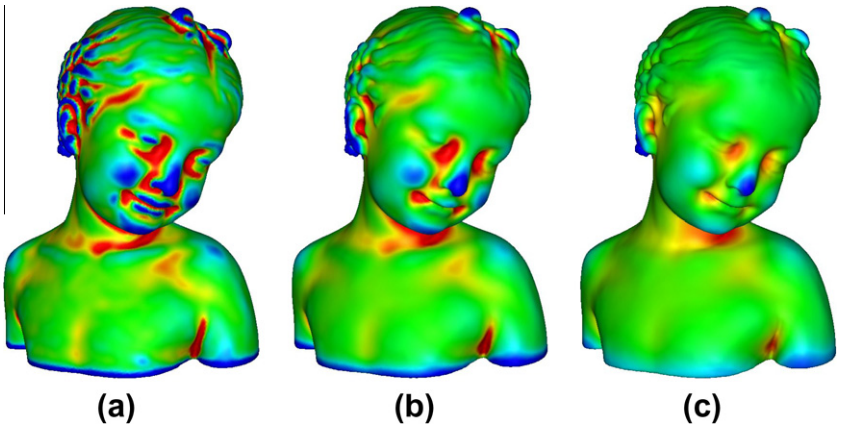


Fig. 2. The CS3 representation of the Bimba model at scales (a) $t = 3.0$, (b) $t = 7.5$, (c) $t = 13.8$.

respectively. Let $F^l : \mathcal{V} \to \mathbb{R}$ denote the smoothed discrete surface signal (curvatures) at level l, and define $\mathbf{F}^l = \begin{pmatrix} F^l(v_1) & \cdots & F^l(v_N) \end{pmatrix}^\top$. We employ the implicit surface smoothing scheme of [12] to obtain the smoothed surface signal, $\mathbf{F}^{l+1}$, at level $l+1$, by solving the following sparse system of linear equations

$$(\mathbf{I} - \lambda_l \mathbf{L})\mathbf{F}^{l+1} = \mathbf{F}^l, \tag{5}$$

where $\lambda_l > 0$ is a time step, and $\mathbf{L}$ and $\mathbf{I}$ denote the $N \times N$ Laplacian and identity matrices, respectively. The elements of the Laplacian matrix $\mathbf{L} = (w_{ij})_{N\times N}$ are given as

$$w_{ij} = \begin{cases} -1 & \text{for } i = j, \\ \frac{1}{|\mathcal{N}(i)|} & \text{for } j \in \mathcal{N}(i), \\ 0 & \text{otherwise}, \end{cases} \tag{6}$$

where $|\mathcal{N}(i)|$ denotes size of the 1-ring neighbor set $\mathcal{N}(i)$ of vertex v_i. The Laplacian matrix may also be populated with other types of weights, such as cotan weights [12]. The linear system in Eq. (5) can be efficiently solved using the Biconjugate Gradient method.

The scale-space representation of the surface signal $\mathbf{f}$ is then given by the sequence $(\mathbf{F}^0, \ldots, \mathbf{F}^{L-1})$, which is obtained recursively using

$$\mathbf{F}^l = \begin{cases} (\mathbf{I} - \lambda_{l-1}\mathbf{L})^{-1}\mathbf{F}^{l-1} & \text{if } l > 0, \\ \mathbf{f} & \text{if } l = 0, \end{cases} \tag{7}$$

for $l = 0, \ldots, L-1$.

The resulting transfer function of the implicit Laplacian smoothing in Eq. (5) is $h(\omega) = (1 + \lambda_l \omega^2)^{-1}$, where ω denotes surface signal frequency [12]. When a stack of smoothed signals with L levels is constructed according to Eq. (7), with corresponding time steps $(\lambda_0, \ldots, \lambda_{L-2})$, the transfer function of the filter at level $L-1$ is given by

$$h_{L-1}(\omega) = \prod_{l=0}^{L-2} (1 + \lambda_l \omega^2)^{-1}. \tag{8}$$

Note that the representation needs to be defined in a recursive manner since the transfer function of the filter defined by Eq. (5) is not a Gaussian. On the other hand, the transfer functions of our recursive formulation approach Gaussians as L grows.

The time steps are selected as $\lambda_l = \lambda_{l-1}\delta = \lambda_0 \delta^l$, where λ_0 denotes an initial time step and $\delta > 1$ is a constant. It is important to note that the time steps λ_l are not equivalent to the scale parameter t in the original scale-space representation of signals given by Eq. (2). Fig. 2 shows a 3D model and its corresponding CS3 representation at various scales.

2.3.1. Estimating the scale parameter

To recover the scale parameter t at each level l, we fit a Gaussian to the transfer function of the smoothing filter for that level, and define the scale of the smoothed signal as the scale of the fitted Gaussian. This is done by sampling the transfer function h_l in Eq. (8). As a result, we obtain a set of pairs $\Gamma = \{(\omega_j, h_l(\omega_j))\}_{j=0}^{j<|\Gamma|}$, which is used to estimate the scale t_l of a fitted Gaussian $g_l(\omega, t_l) = e^{-\omega^2 t_l}$, in the least-squares sense:

$$t_l = \frac{\sum_{j=0}^{j<|\Gamma|} \omega_j^2 \sum_{k=0}^{k<l-1} \ln\left(1 + \lambda_k \omega_j^2\right)}{\sum_{j=0}^{j<|\Gamma|} \omega_j^4}. \tag{9}$$

The scale parameter t_l for each level l can alternatively be defined in terms of the variance of the transfer function at that level:

$$t_l = \frac{\int_{-\infty}^{\infty} \left(\prod_{k=0}^{l-1} (1 + \lambda_k \omega^2)^{-1}\right) d\omega}{\int_{-\infty}^{\infty} \left(\omega^2 \prod_{k=0}^{l-1} (1 + \lambda_k \omega^2)^{-1}\right) d\omega}. \tag{10}$$

Since the transfer function at each level is analytic and only depends on the known sequence of time steps, λ_l, its variance can be precomputed numerically. In this work, we use Eq. (9) to estimate the scale parameter. The obtained sequence of scales, $(t_0, \ldots, t_{L-1})$, together with the stack of smoothed signals, $(\mathbf{F}^0, \ldots, \mathbf{F}^{L-1})$, define the CS3 representation of the surface.

2.4. Feature extraction with automatic scale selection

The CS3 representation of a 3D surface may be used directly for feature extraction. Let $\Phi(\mathcal{M}) = (\mathbf{F}^0, \ldots, \mathbf{F}^{L-1})$ and $\Psi(\mathcal{M}) = (t_0, \ldots, t_{L-1})$ correspond to the CS3 representation of surface $\mathcal{M}$. The difference between the smoothed signals at consecutive levels l and $l+1$ can be used to approximate the Laplacian of the signal at level l. This difference can be stated in terms of convolution of the original signal with Gaussian filters as

$$\mathbf{F}^{l+1} - \mathbf{F}^l \approx \mathbf{F}^0 * (g(\cdot; t_{l+1}) - g(\cdot; t_l)), \tag{11}$$

where $*$ denotes convolution defined over the surface, and $g(\cdot; t_l)$ is a Gaussian with scale t_l. Noting that $\frac{\partial g}{\partial t} = 0.5\Delta g$, we have

$$\frac{\partial g}{\partial t} = 0.5\Delta g \approx \frac{g(\cdot; t_{l+1}) - g(\cdot; t_l)}{t_{l+1} - t_l}, \tag{12}$$

and consequently,

$$\mathbf{F}^{l+1} - \mathbf{F}^l \approx 0.5(t_{l+1} - t_l)\mathbf{F}^0 * \Delta g. \tag{13}$$

Therefore, the estimated Laplacian of $\mathbf{F}^0$, at level l, which we denote by $\Delta\mathbf{F}^l$, is approximated by

$$\Delta\mathbf{F}^l \approx \frac{2(\mathbf{F}^{l+1} - \mathbf{F}^l)}{t_{l+1} - t_l}. \tag{14}$$

We define the *scale-normalized* Laplacian of the surface signal at scale t_l as

$$\Delta_{norm}\mathbf{F}^l = t_l \Delta\mathbf{F}^l = \frac{2t_l(\mathbf{F}^{l+1} - \mathbf{F}^l)}{t_{l+1} - t_l}. \tag{15}$$

Throughout this work, we assume the surface signal corresponds to the surface mean curvatures. $\Delta_{norm}\mathbf{F}$ then corresponds to the scale-normalized Laplacian of mean Curvatures (LoC).

The local extrema of $\Delta_{norm}\mathbf{F}$ could be used to define feature points (keypoints) on a 3D model. For example, Fig. 3 depicts the computed scale-normalized Laplacian of mean curvatures on a 3D model and its noisy counterpart, at level $l = 20$ (scale $t = 21.7$); the red spheres indicate the loca-

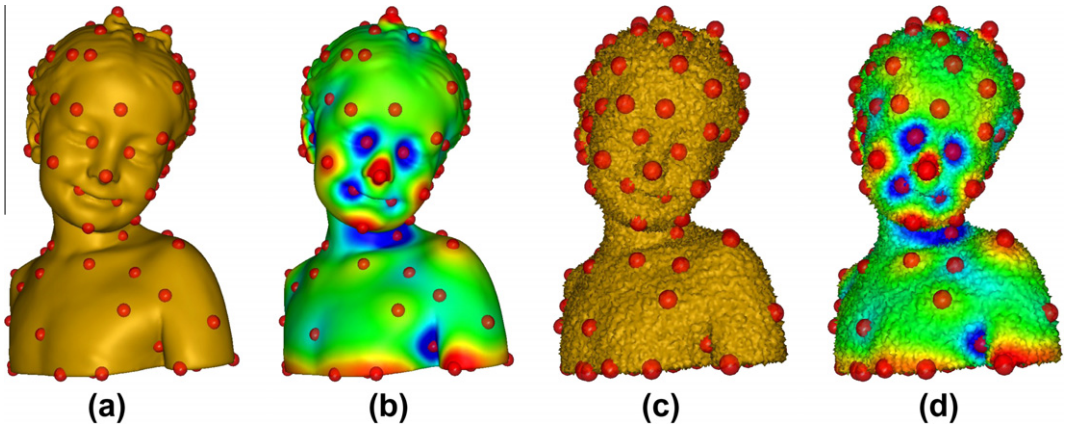


Fig. 3. Extracted features on (a) original, and (c) noisy Bimba models at $t = 21.7$; the false-colors in (b) and (d) reflect the response of the Δ^{si} (Eq. (16)) at each vertex on the original and noisy models, respectively. The models in (c) and (d) contain 80% Gaussian noise. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

tions where LoC is locally maximum or minimum at the displayed level. As seen in the figure, the detected locations of the extrema of LoC, despite their high differential order, are robust against noise and may be used for extraction of stable and well-localized feature points. Additionally, note that the extracted features are distributed throughout the entire surface.

The plots in Fig. 4 show the computed LoC values at a few selected vertices on the models in Fig. 3 as functions of scale. As expected, the values for both the noisy and noise-free models converge at the higher scales. However, the corresponding LoC values of the vertices at the scale shown in Fig. 3 are not the same between the two models due to the noise. To alleviate this, we introduce the *scale-invariant* LoC as

$$\Delta^{si}\mathbf{F}^l = \frac{\Delta\mathbf{F}^l - \bar{\mathbf{F}}^l}{\sigma_l}, \tag{16}$$

where

$$\bar{\mathbf{F}}^l = \frac{1}{N}\mathbf{1}^\top\Delta\mathbf{F}^l\mathbf{1}, \quad \sigma_l = \frac{1}{\sqrt{N}}\|\Delta\mathbf{F}^l - \bar{\mathbf{F}}^l\|, \tag{17}$$

denote the vector-form mean, and standard deviation of the LoC values at level l, respectively; N is the total number of vertices in $\mathcal{M}$, and $\mathbf{1}$ is an N-dimensional vector of all 1's.

Fig. 5 shows the scale-invariant LoC plots of the same vertices as in Fig. 4. As can be seen, the LoC curves of the two surfaces begin to converge at a much finer scale, and look more similar. The scale-invariant LoC is resilient to changes in resolution, spatial scaling, and additive i.i.d. noise.

According to the principle of automatic scale selection [7], the scale(s) where $\Delta_{norm}\mathbf{F}_i$ becomes a local extremum across scales can be expected to correspond to the size of surface structures at vertex v_i. This is visually verified in Figs. 6 and 1, where the size of the blue spheres indicate the computed spatial scale (neighborhood size) at a few selected keypoints. An approach similar to Lowe's [8] was used to select the keypoints (shown as red spheres) on the models, in the two figures. The keypoints were selected as the vertices that were local extrema among their immediate neighbors, both on the current level and two adjacent levels on the stack: let set $Q^l(i) = \{\mathbf{F}_j^{l+k}\} \cup \{\mathbf{F}_i^{l-1}, \mathbf{F}_i^{l+1}\}$, for $k = -1, 0, 1$, and $j \in \mathcal{N}(i)$. Then, vertex v_i, at level l, is selected as a keypoint if $\mathbf{F}_i^l > q_j, \forall q_j \in Q^l(i)$ or $\mathbf{F}_i^l < q_j, \forall q_j \in Q^l(i)$. Let t_l be the scale associated with level l. t_l then defines the scale of the detected keypoint v_i. The radius of each blue sphere in Figs. 6 and 1 was computed using

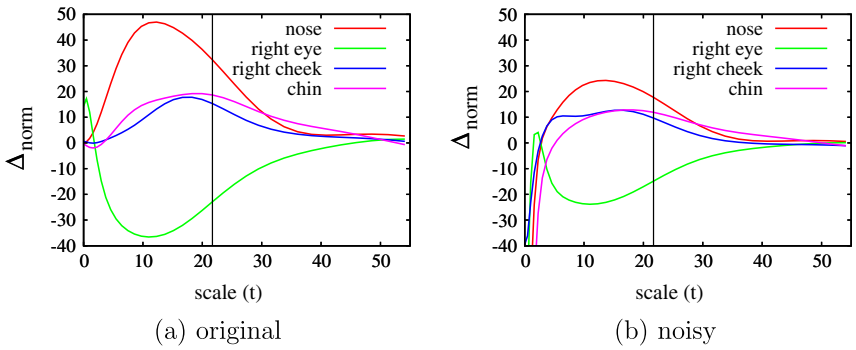


Fig. 4. Plots of LoC values of a few vertices on the surfaces in Fig. 3. The vertical black lines indicate the location of the displayed scale ($t = 21.7$) in Fig. 3.

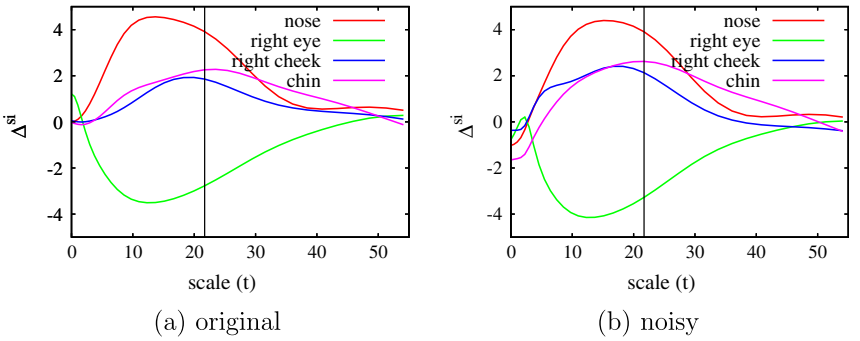


Fig. 5. Plots of the *scale-invariant* LoC values of a few vertices on the surfaces in Fig. 3.

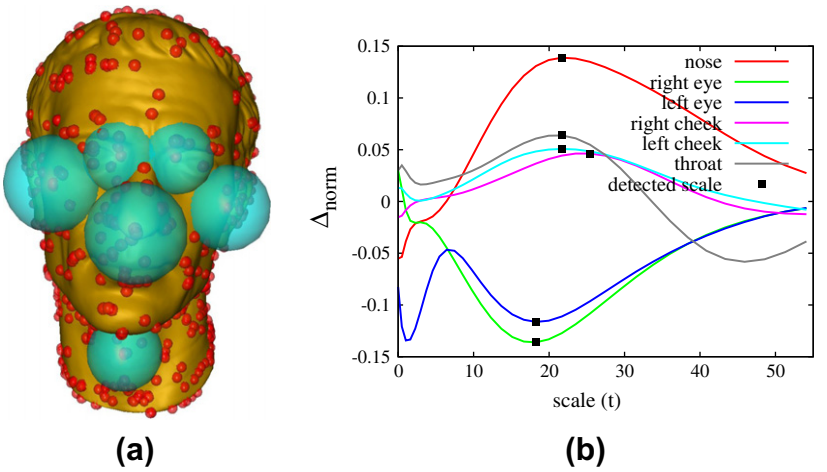


Fig. 6. Automatic scale selection on the Caesar model. (a) Estimated scales at a few locations; the radii of the blue spheres are computed using Eq. (18). (b) Plots of the scale-normalized Laplacian of the surface mean curvatures at the selected vertices as functions of scale; the locations of the filled squares on the scale-axis indicate the detected scale for the keypoints. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$r = t_l \bar{e}, \tag{18}$$

where $\bar{e}$ is the average edge length in the surface mesh.

The graph in Fig. 6c shows the plots of LoC values at the few selected keypoints (blue spheres) on the model in Fig. 6a. The filled squares on the curves indicate the location of the detected scale for each keypoint. In our experiments we noticed that the estimated scale parameter for keypoints extracted at the lower scales ($t < 3$) was more sensitive to noise and therefore less reliable. As a result, we clip the detected scale of all keypoints with a value of $t < 3$ to 3.

Fig. 7 provides a comparison between the Δ^{si} plots on the original, scaled, and higher resolution versions of the same model as in Fig. 3a. The higher resolution version of the model was obtained by applying one iteration of Loop's subdivision scheme, which increases the number of mesh vertices by a factor of four, and approximately halves the average edge length in the resulting mesh. As can be seen, spatial scaling of the model has no effect on the plotted Δ^{si} curves. On the other hand, the increase in the resolution of the surface scales the LoC curves, and consequently the locations of their extrema, by a factor of two. This results in the detected scale, t_l, for each vertex to be scaled by two. Since the increase in the resolution of the surface halved the average edge length, $\bar{e}$, the extracted radii of the surface features (r in Eq. (18)) remain approximately the same between the original and higher resolution model. This guarantees that the detected surface feature sizes are intrinsic to the surface.

In Fig. 8, we show the effects of noise on the positions of extracted keypoints on a model. The figure shows the generalized Voronoi diagram of the keypoints on the surface of the model. Each cell's false color shows how much its corresponding node was displaced between the original and the noisy model. As the table below the figure shows, the average displacement in terms of the average edge length on the surface mesh is approximately four vertices. Additionally, as evident in the figure, the displacement is small in areas with high curvature, and large on more planar regions.

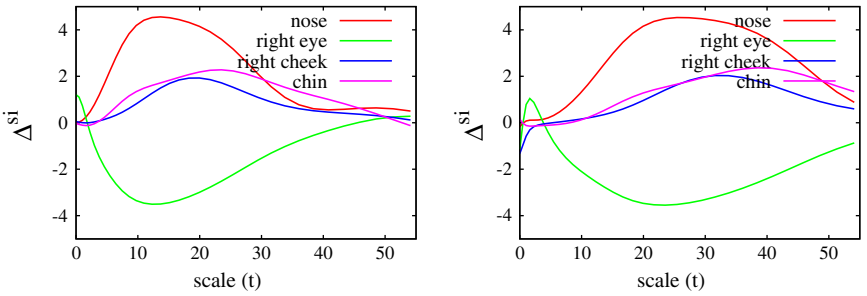
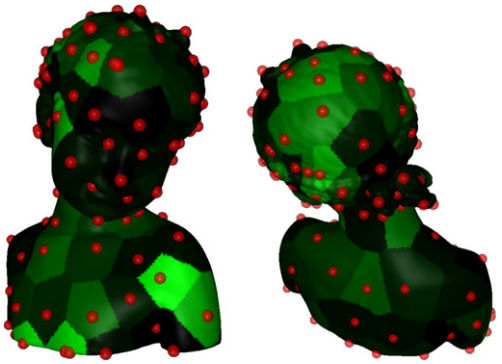


Fig. 7. Comparison of scale-invariant LoC plots of the Bimba model (Fig. 3a) with different spatial scales and sampling resolutions. Plot in (a) is identical to the plot for the original model, shown in Fig. 5a, while (b) has been scaled by a factor of approximately two.



min	max	avg.	std.	avg. edge len	avg. disp. nbrsz
0.000	0.127	0.023	0.022	0.0057	4.044

Fig. 8. Displacements of extracted keypoints between the original and a noisy version (80% Gaussian) of the Bimba model. The darker and lighter cell colors indicate smaller and larger displacements, respectively. The table shows a summary of the displacement statistics on the model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

2.5. Performance evaluation

In this section, we evaluate the performance of our proposed keypoint extractor using the evaluation benchmark of [23], where the methods are categorized into two classes: *fixed scale* and *adaptive scale* detectors. As the names imply, a fixed scale detector operates at a constant scale, while an adaptive scale detector operates on a range of possible scales. The following detectors were evaluated in the experiments:

- *Fixed scale*: Local Surface Patches (LSPs) [24], Intrinsic Shape Signatures (ISSs) [25], KeyPoint Quality (KPQ) [26], Heat Kernel Signature (HKS) [20].
- *Adaptive scale*: Laplace–Beltrami Scale-Space (LBSS) [19], MeshDoG [18], KeyPoint Quality—Adaptive-Scale (KPQ-AS) [26], Salient Points (SP) [16].

The experiments were run on five datasets:

- *Kinect*: data obtained using a Microsoft Kinect device.
- *Space time*: data obtained using a stereo reconstruction technique.
- *Laser scanner*: data obtained from a laser scanner [26].
- *Retrieval*: synthetic noisy data created using models from the Stanford Repository—single, complete 3D surfaces were used to create the individual models and uncluttered scenes with no occlusions.
- *Random views*: synthetic noisy data created using models from the Stanford Repository—unlike the *Retrieval* dataset, the scenes contained multiple models, clutter and occlusions.

These are the same datasets used in [23] and are included here for direct comparison with the competing methods they evaluated in their benchmark. We compared the performance of our CS3 representation against both fixed and adaptive scale classes of detectors. In all adaptive scale experiments, a CS3 stack with 32 levels ($1 < t < 48.6$) was obtained for all input surfaces and the automatic scale selection mechanism described in Section 2.4 was used to extract keypoints. In the fixed scale experiments, the keypoints were obtained as the local extrema of the Laplacian of surface mean curvatures at the desired scale (t) in the CS3 stack. The level whose estimated scale (t) was closest to the experiment's scale was used in each case.

The following definitions for absolute repeatability, relative repeatability, and scale repeatability are used from [23]: A keypoint k_h^i extracted from model M_h is said to be repeatable in scene S_l, under the ground truth rotation R_{hl} and translation t_{hl}, if a keypoint k_l^j exists in S_l such that

$$\|R_{hl}k_h^i + t_{hl} - k_l^j\| < \epsilon, \tag{19}$$

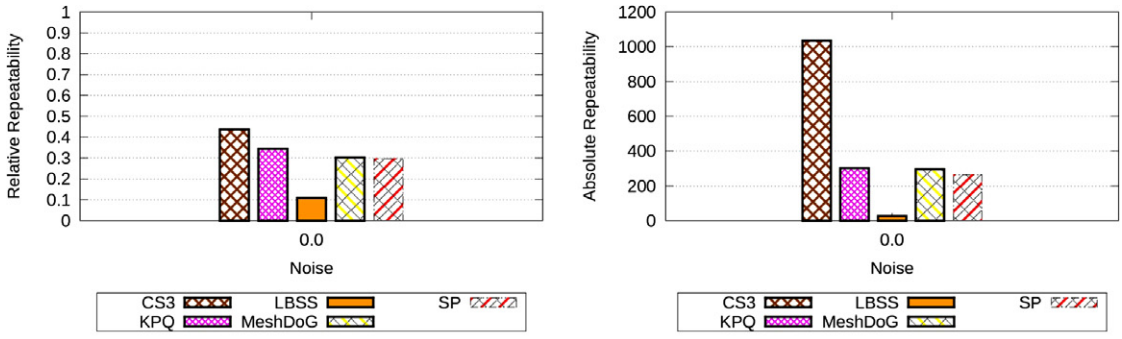
where ϵ is a distance threshold. Let RK_{hl} denote the set of repeatable keypoints between the model/scene pair (M_h,-S_l). The *absolute repeatability* is defined as

$$r_{abs} = |RK_{hl}|, \tag{20}$$

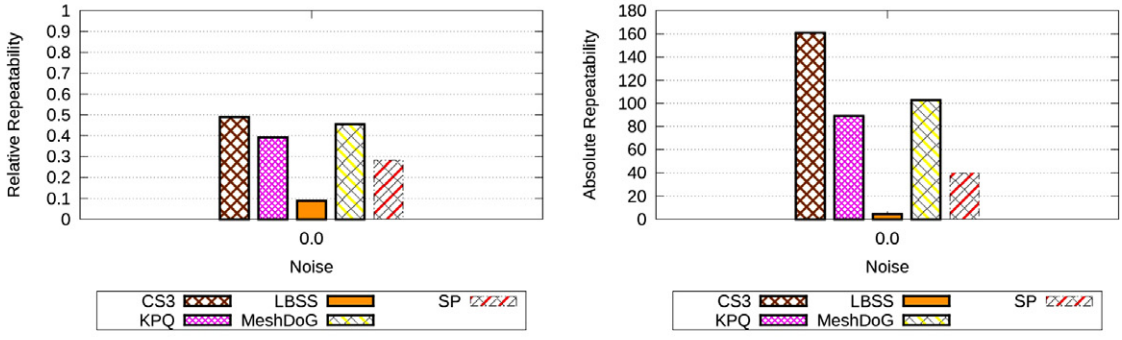
and the *relative repeatability* is defined as

$$r = \frac{|RK_{hl}|}{|K_{hl}|}, \tag{21}$$

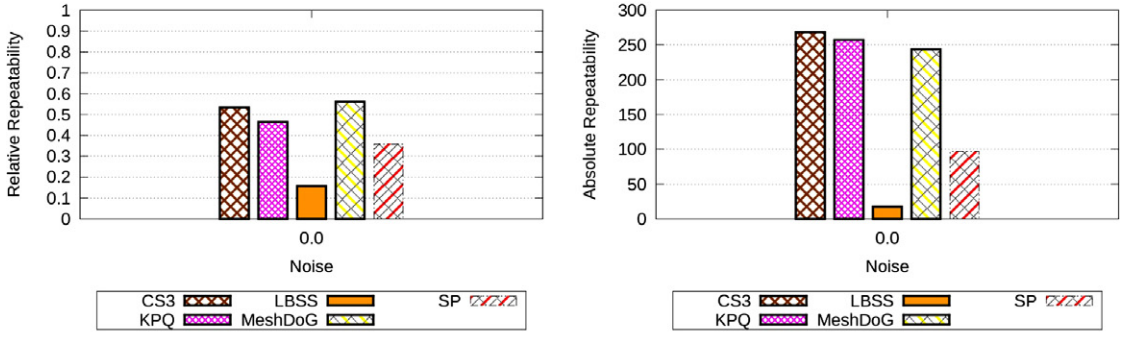
where K_{kl} is the set of all keypoints extracted from model M_h that are not occluded in scene S_l. Distance threshold of $\epsilon = 2 \times$ *mesh resolution* (mr) was used in the experiments; mesh resolution denotes the average edge length in a mesh.



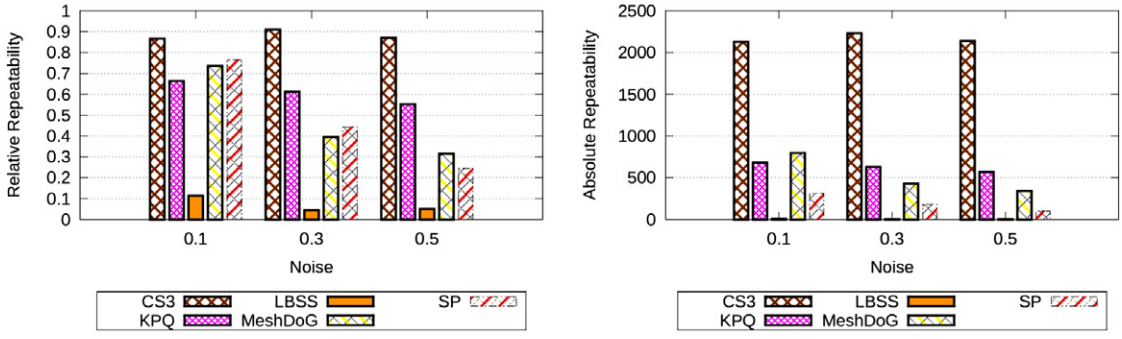
Laser Scanner



Kinect



Space Time



Retrieval

Fig. 9. Relative and absolute repeatability scores of adaptive scale detectors.

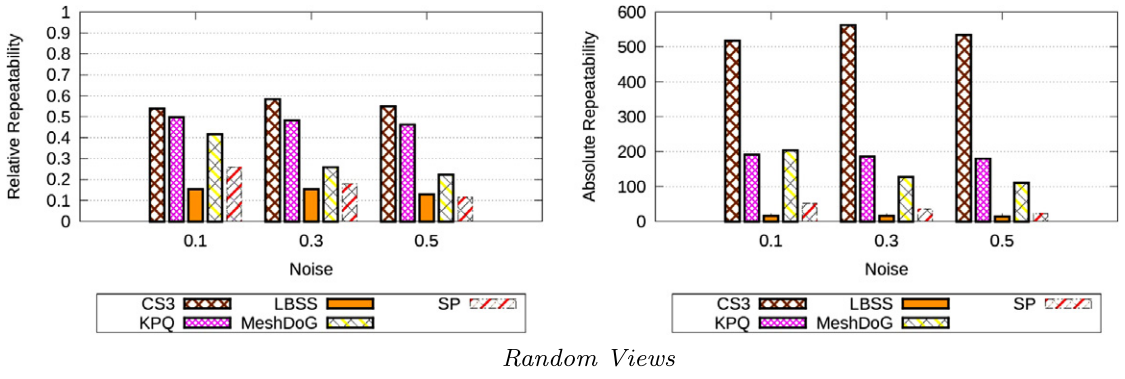


Fig. 9. (*continued*)

The scale repeatability of pair of keypoints k_h^i and k_l^j with respective scales σ_h^i and σ_l^j is defined as

$$r_{scale}^{ij} = \frac{V\left(Sphere(\sigma_h^i) \cap Sphere\left(\sigma_l^j\right)\right)}{V\left(Sphere(\sigma_h^i) \cup Sphere\left(\sigma_l^j\right)\right)}, \tag{22}$$

where $Sphere(\sigma)$ and $V(Sp)$ denote the sphere with radius σ and volume of region Sp in R^3, respectively. The *overall scale repeatability* of the set of keypoints extracted for a model/scene pair is defined as

$$r_{scale} = \frac{\sum_{(k_h^i, k_l^j) \in RK_{hl}} r_{scale}^{ij}}{|RK_{hl}|}. \tag{23}$$

Fig. 9 compares the relative and absolute repeatability of all adaptive scale detectors in the experiments. Both the absolute and relative repeatability score of CS3 were consistently at the same level or better than the other approaches. The relative repeatability of CS3 is slightly lower than MeshDoG's for the *Space Time* dataset, however, in all other cases, CS3 performs better than MeshDoG.

In Fig. 10, we compare the scale repeatability scores of the adaptive scale detectors. In all experiments, CS3 performed slightly worse than LBSS but outperformed all other methods. LBSS, while having the best scale repeatability score, consistently performed worse than all other methods in both absolute and relative repeatability. This behavior, however, is to be expected as LBSS tends to select much fewer keypoints than other methods.

In Fig. 11, we compare the relative and absolute repeatability scores of fixed scale detectors at various scales on the *Laser Scanner*, *Kinect*, and *Space Time* datasets. The absolute repeatability of CS3 is higher than the rest of the methods in most cases, while its relative repeatability is at the same level as those of ISS and KPQ: on the *Laser Scanner* dataset, ISS performs slightly better than CS3 in all scales, while CS3 performs better than ISS on the *Kinect* dataset. However, the absolute repeatability of ISS is much lower than that of CS3 in all cases. Fig. 12 shows the relative and absolute repeatability scores of fixed scale detectors for the *Retrieval* and *Random Views* datasets. In the majority of cases, CS3 has a higher absolute repeatability score, while its relative repeatability score drops lower

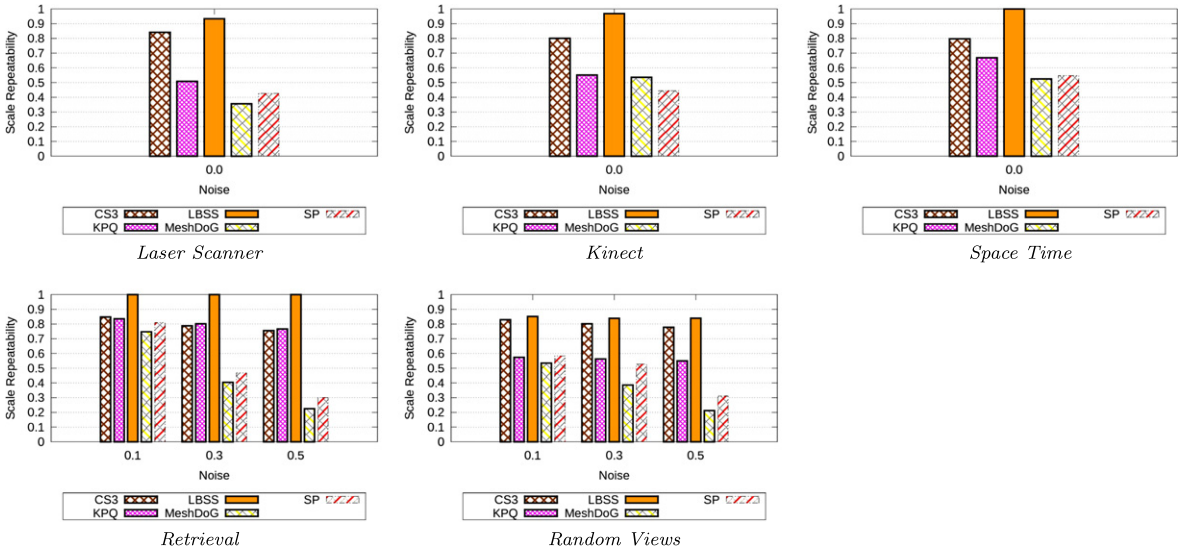


Fig. 10. Scale repeatability scores of adaptive scale detectors.

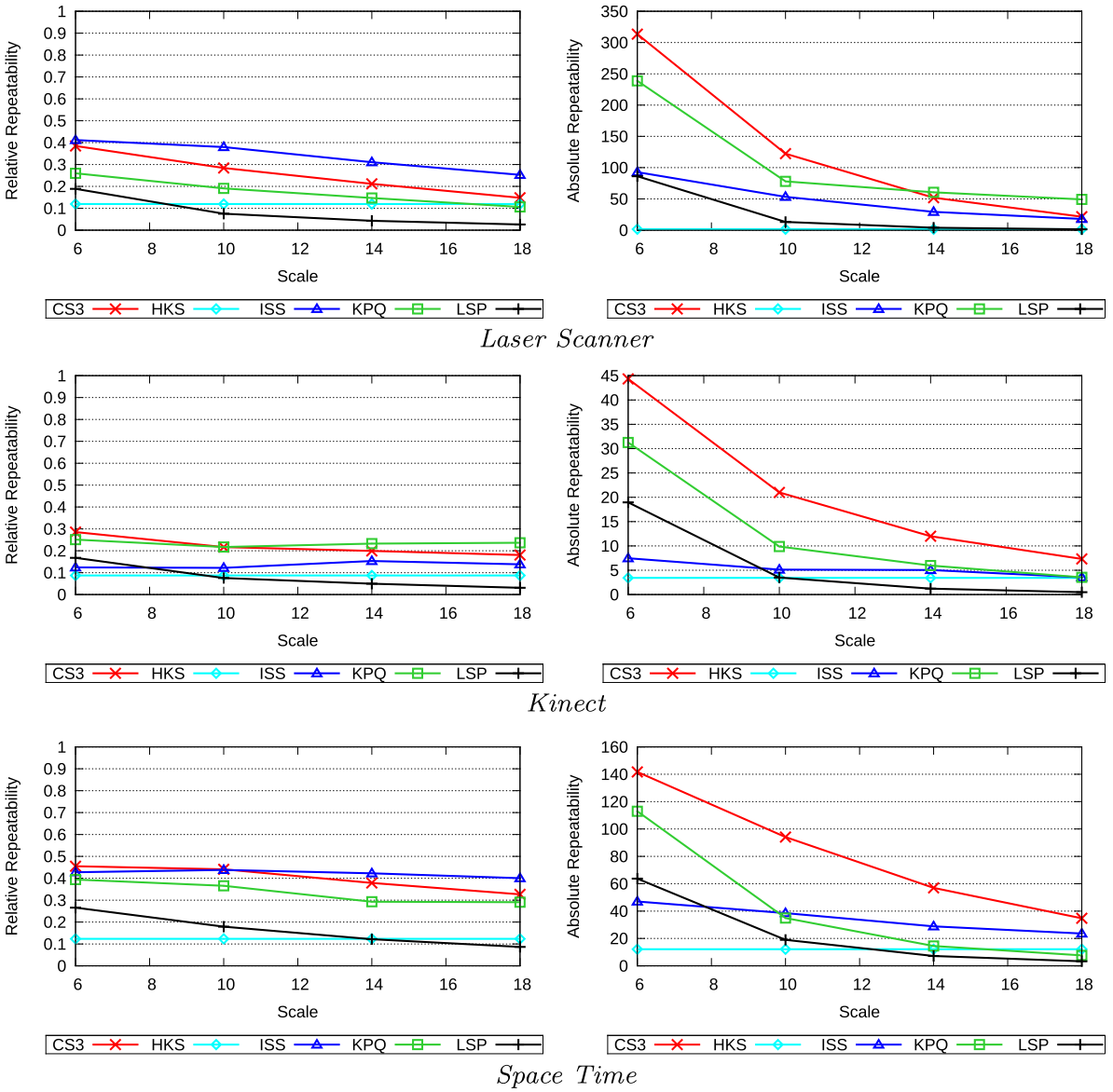
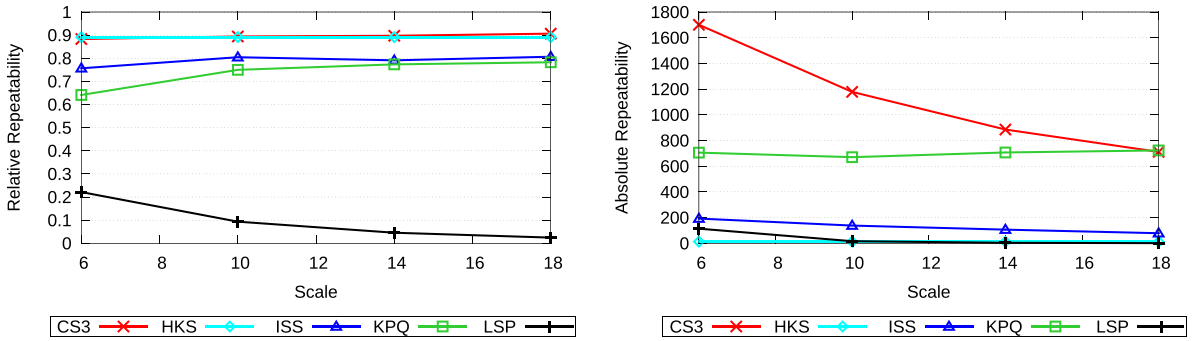


Fig. 11. Relative and absolute repeatability scores of fixed scale detectors on the *Laser Scanner*, *Kinect*, and *Space Time* datasets.

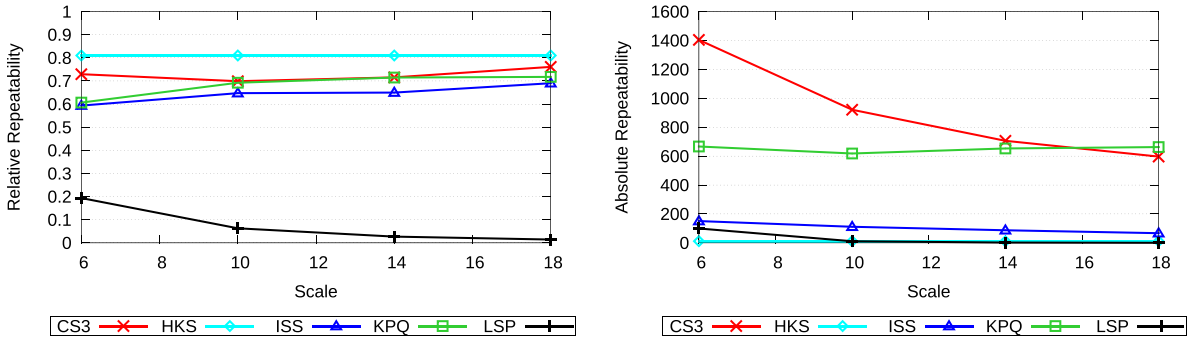
than HKS and KPQ in the *Retrieval* experiment with noise level of 0.5 mr.

It is important to note that the orientations of a number of mesh faces in the 3D models in the *Retrieval* and *Random Views* datasets were incorrect (flipped). For example, out of 45,195 faces in the Happy Buddha model, 4293 faces were incorrectly flipped. When reading these models, our PLY parser added those faces as separate faces (thereby modifying the geometry and topology of the surfaces). Since the noise and geometric transformations in the two datasets were synthetic (i.e., scenes were obtained from the same problematic 3D models), the locations of the problematic faces may have served as landmarks for our detector. This may explain the suspiciously good performance of our detector for the *Random Views* experiment in Fig. 12. Nonetheless, the performance of our fixed and adaptive scale detectors are consistently on par or better than the other methods in the experiments involving the other datasets.

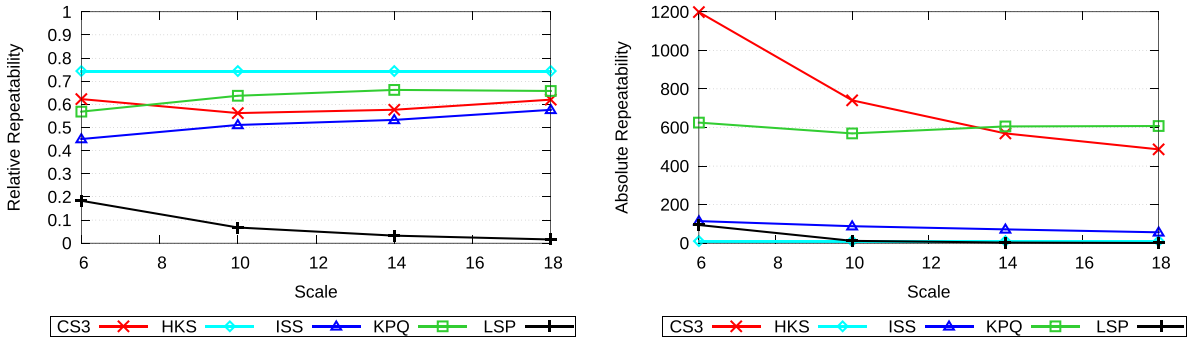
We followed the same methodology as in [23] to obtain the timings reported in Fig. 13: lower resolutions of a mesh from the *Kinect* dataset with approximately 267 K vertices were obtained by successively decimating it using the approach of [27]. We then ran our adaptive and fixed scale detectors on the resulting meshes on a single core of a 2.3 GHz CPU (Intel Core i7-2820QM). For each fixed scale detector, the reported timing in [23] corresponds to the scale at which the detector had the best relative repeatability score. Our fixed scale detector, similar to the other methods, had its best performance at scale 6. We report



Retrieval (Gaussian noise std: 0.1mr)



Retrieval (Gaussian noise std: 0.3mr)



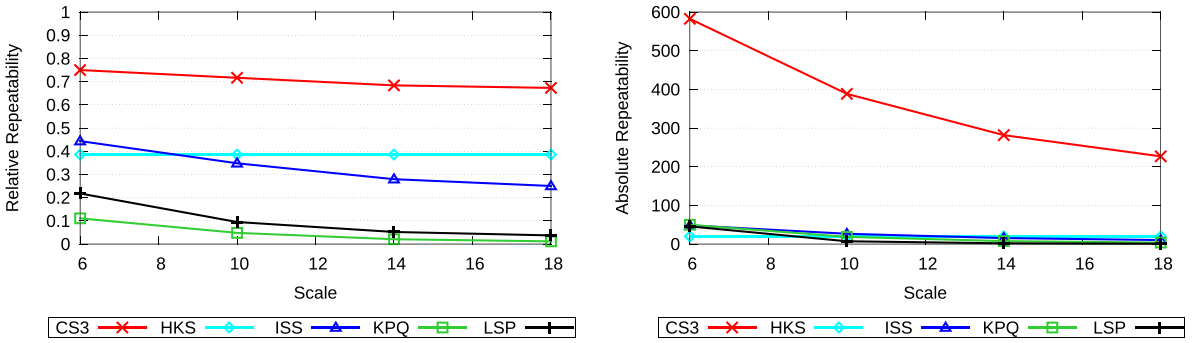
Retrieval (Gaussian noise std: 0.5mr)

Fig. 12. Relative and absolute repeatability scores of fixed scale detectors on the *Retrieval* and *Random Views* datasets.

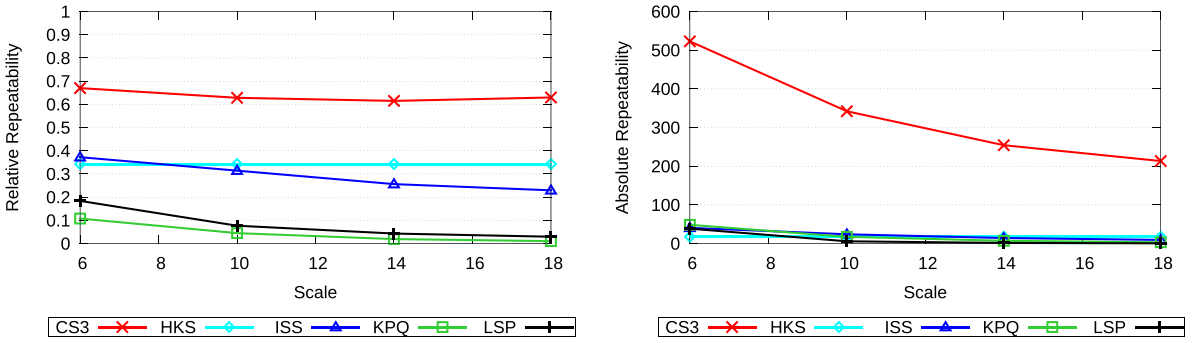
the timings for our adaptive scale and fixed scale detectors in Fig. 13b. In the same graph we have also included the timings for the most time consuming scale ($t = 18$) in the fixed scale experiments. As can be seen, both our fixed scale and adaptive scale detectors outperform the other methods. Specifically, the KPQ detector which has the best relative repeatability scores among the other methods, performs more than two orders of magnitude slower than the CS3 detector. Additionally, the HKS detector, because of its memory requirements, was unable to handle meshes larger than 30 K vertices. Fixed scale detectors ISS and LSP report similar timings as CS3. However, LSP performs consistently worse than CS3 in all fixed scale experiments, while ISS performs at the same level as CS3 in the *Kinect*, *Laser Scanner*, and *Space Time* experiments, and worse in the *Retrieval* and *Random Views* experiments. Moreover, ISS, unlike CS3, is not capable of performing adaptive (automatic) scale selection. The efficiency of adaptive scale detector MeshDoG is similar to CS3's. However, it performs worse than CS3 in the repeatability experiments.

3. Application: 3D face recognition

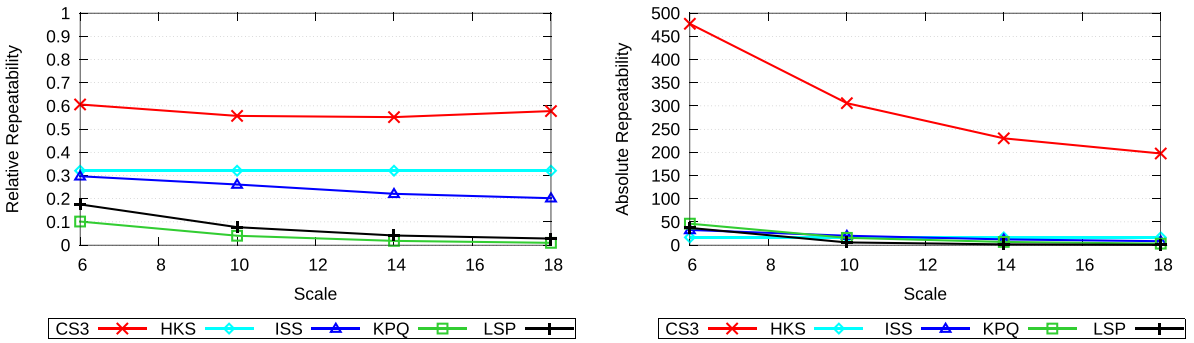
We tested the discriminative power of our proposed scale-invariant Laplacians of surface curvatures in a simple PCA-based 3D face recognition system. The input to the



Random Views (Gaussian noise std: 0.1mr)



Random Views (Gaussian noise std: 0.3mr)



Random Views (Gaussian noise std: 0.5mr)

Fig. 12. (*continued*)

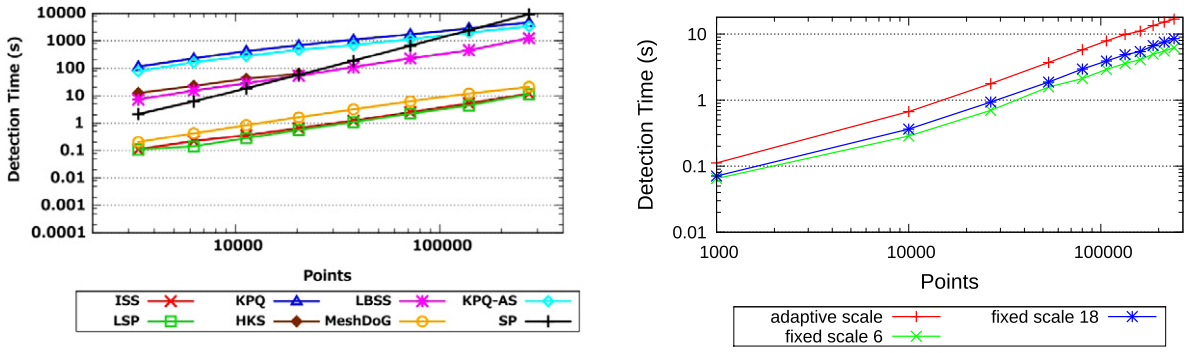
system was a set of 3D faces that were already registered and consistently remeshed using the approach of [28]. Here, "consistent" means that all meshes have the same number of vertices and a one-to-one correspondence between the vertices on the meshes is known. This property simplifies the process of converting the meshes into feature vectors, which are used for recognition.

A large body of literature exists on 2D and 3D face recognition [29,30]. For example, approaches such as the Bayesian face recognition of [31], or face recognition using sparse representation [32,28] may be used for this task. However, in our face recognition system, we employ the most well-known approach of eigenfaces [33]. We choose this approach due to its simplicity and ease of implementation. More importantly, this choice enables us to attribute the better performance of our recognition system (see Section 3.2) to its feature extraction component, rather than the classifier it employs.

In Section 3.1, we discuss the steps involved in the training and recognition phases of our algorithm. In Section 3.2, we present the recognition results of our system on different datasets and compare its performance against other state-of-the-art 3D face recognition techniques.

3.1. 3D face recognition using CS3 representation

In both the training and recognition stages of our algorithm, each face mask f_m with N vertices is converted into



(a) other methods (as reported in [23]) (b) CS3

Fig. 13. Timing comparison of CS3 with other methods.

an N-dimensional feature vector by first constructing its CS3 representation and computing the scale-invariant Laplacians of Curvatures (si-LoCs) at some level l in the CS3 stack (see Eq. (16)); the optimal choice of l is discussed in the results section. The feature vector is then formed by arranging the si-LoC values of the mesh vertices into an N-dimensional vector. Since the face masks for all faces are obtained using the same procedure, the vertices in all meshes have the same ordering and, as a result, the constructed feature vectors are consistent. We denote the feature vector corresponding to face mask f_m by vector $\mathbf{x}_m \in \mathbb{R}^N$.

Each vector $\mathbf{x}_m$ corresponds to a point in the N-dimensional feature space. Under the assumption that the feature vectors are constructed judiciously, multiple feature vectors corresponding to different 3D scans of the *same individual* are expected to form a cluster in the feature space. The objective of the training phase is then to obtain information about the characteristics (e.g., the shapes) of these clusters. In the recognition phase, this information is used to decide to which cluster a given input feature vector belongs. Therefore, in this paradigm, face recognition is treated as a feature classification problem.

The eigenfaces approach that is employed in this work is a simple nearest neighbor classifier. As mentioned previously, we show that despite this choice, our system is capable of outperforming most state-of-the-art 3D face recognition techniques. We argue that this good performance is due to both the *discriminative power* of our feature vectors and their *resilience to noise*. In this application domain, the noise may be due to *surface perturbations* or *facial expressions* in the input faces.

3.1.1. 3D eigenfaces

Let $\mathbf{X} = \{(\mathbf{x}_m, c_m)\}_{m=1}^M$ be the training set; $\mathbf{x}_m \in \mathbb{R}^N$ and c_m denote the mth feature vector and its associated class in the training set, respectively. Let

$$\boldsymbol{\mu} = \frac{1}{M}\sum_{m=1}^{M}\mathbf{x}_m, \quad \boldsymbol{\Sigma} = \frac{1}{M}\sum_{m=1}^{M}(\mathbf{x}_m - \boldsymbol{\mu})(\mathbf{x}_m - \boldsymbol{\mu})^\top \tag{24}$$

denote the mean vector and covariance matrix of the feature vectors in $\mathbf{X}$. The eigendecomposition of $\boldsymbol{\Sigma}$ is given as $\boldsymbol{\Sigma} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^\top$, where the $N \times N$ matrices

$$\boldsymbol{\Phi} = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \cdots & \mathbf{u}_N \\ | & & | \end{pmatrix} \quad \text{and} \quad \boldsymbol{\Lambda} = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_N \end{pmatrix} \tag{25}$$

contain the eigenvectors and eigenvalues of $\boldsymbol{\Sigma}$, respectively. It is assumed that the eigenvalues are ordered in descending order; i.e., $\lambda_1 \geqslant \cdots \geqslant \lambda_N \geqslant 0$.

In the case of face recognition, where each $\mathbf{x}_m$ is derived from a face model, the K-major eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_K$ are referred to as "eigenfaces". The eigenfaces span a K-dimensional subspace of the feature space with the smallest total orthogonal distance from the feature points in the training set. Therefore, the projection of the feature vectors onto the subspace spanned by the eigenfaces results in dimensionality reduction of the feature vectors, with the minimal loss of variance. We refer to the subspace spanned by the eigenfaces simply as the *eigenspace*. When $N \gg K$, this dimensionality reduction enables efficient processing of the data, which would otherwise be computationally prohibitive.

In the eigenfaces approach, both the training and test sets are projected onto the eigenspace, and the classification tasks are performed in this space. Let $\mathbf{x}' \in \mathbb{R}^K$ denote the projection of feature vector $\mathbf{x} \in \mathbb{R}^N$ onto the eigenspace:

$$\mathbf{x}' = \mathbf{U}^\top(\mathbf{x} - \boldsymbol{\mu}), \tag{26}$$

where

$$\mathbf{U} = (\,\mathbf{u}_1 \quad \cdots \quad \mathbf{u}_K\,)_{N\times K}. \tag{27}$$

Let set $\mathbf{X}' = \{(\mathbf{x}'_m, c_m)\}_{m=1}^M$ denote the transformed training set obtained by projecting the feature vectors in the training set onto the eigenspace. In the recognition phase, classification is performed by assigning each test feature vector $\mathbf{x}_t \in \mathbb{R}^N$ to class c^* of $\mathbf{x}^* \in \mathbf{X}$, which satisfies

$$(\mathbf{x}^*, c^*) = \arg\min_{(\mathbf{x}'_m, c_m)\in\mathbf{X}'} \|\mathbf{x}'_t - \mathbf{x}'_m\|_p, \tag{28}$$

where $\mathbf{x}'_t = \mathbf{U}^\top(\mathbf{x}_t - \boldsymbol{\mu})$, and $\|\cdot\|_p$ denotes the L_p-norm in $\mathbb{R}^K$, for some $p \geqslant 1$. The optimal choice of p is discussed in Section 3.2.

3.2. 3D face recognition results

We tested the performance of our proposed 3D face recognition system on the GavabDB [34] and FRGC [35] datasets. We first present the recognition results of our system for GavabDB and then FRGC.

The models in GavabDB are noisier and of lower resolution than those in FRGC. The scanned faces for each individual in GavabDB contains the following poses and expressions: 1 scan looking up, 1 scan looking down, 2 frontal scans, 1 scan with random gesture, 1 scan with laughter, and 1 scan with smile.

In the first set of tests, we followed the same leave-one-out cross-validation procedure as in [28] to test the accuracy of our system. In each trial, one class of faces (e.g., scans looking up) were used as the test set and the remaining faces in the dataset were used as the training set. The recognition accuracy was measured as the percentage of times the system returned the correct individual for each query face from the test set. Table 1 shows the correct recognition rates of our system for each test set. In the experiments, si-LoC values at level 10 of the CS3 stack were used to form the feature vectors for both the training and test sets. Additionally, the L_1-norm was used in the matching stage of the algorithm, when searching for nearest neighbors in the eigenspace. The test sets in Table 1 have been grouped together into two categories of "neutral" and "non-neutral", to indicate which sets of scans contained facial expressions. The accuracy rates for the two groups (column 4), and the overall accuracy of the system (column 5) were computed by averaging their associated rows in column 2 of the table.

Since CS3 is a multiscale representation, a level l from the CS3 stack must be selected in order to construct the feature vectors for training and matching. Therefore, l is an unknown parameter whose optimal value must be estimated using the training set. Other parameters that will also influence system performance are the initial time step, λ_0, and the factor, δ, by which the step size is increased at each level in the CS3 stack ($\lambda_l = \lambda_0\delta^l$). Throughout this work, we used the following values for these two variables: $\lambda_0 = 1.0$, $\delta = 1.2$. However, the value of l must be selected more carefully as it has a higher influence on the performance of the system.

In Fig. 14, we show how the performance of our recognition system is affected by the choice of l, for each class of test sets. In all cases the accuracy first increases and then decreases. Additionally, Table 2 uses the data from Fig. 14 to show how the recognition rates for the neutral/non-neutral groups of test sets are influenced by the choice of the CS3 level. As can be seen, again level $l = 10$ yields the optimal performance for both classes of tests.

We define the optimal CS3 level for recognition, as the level where the overall accuracy of the system is maximal. Procedure `FindOptimalLevel` in Algorithm 1 summarizes the steps involved in finding the optimal level l^*, using only the training set. The procedure may be iterated a number of times to obtain a set of values for l^*; the arithmetic mean or median of these values may then be used to select the optimal level.

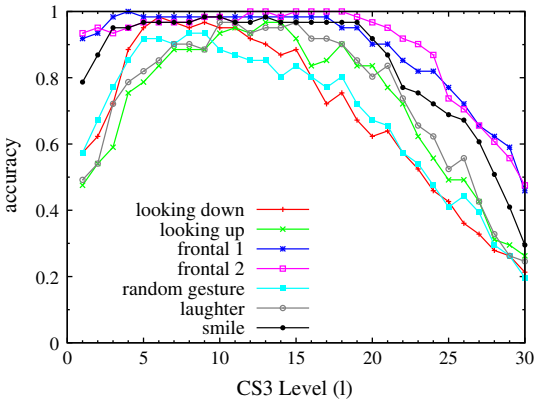


Fig. 14. Accuracy rates for different classes of test sets as functions of the CS3 level. At each level, the si-LoC values were used to form the feature vectors. The L_1-norm was used to find the nearest neighbors in the eigenspace.

Table 1
Face recognition results on GavabDB. The pose column indicates which class of scans was taken as the test set while the remaining scans were used as the training set.

Pose	Acc. (%)	Group	Group acc. (%)	Overall (%)
Looking down	95.08	Neutral	96.31	95.55
Looking up	93.44			
Frontal 1	98.36			
Frontal 2	98.36			
Random gesture	88.52	Non-neutral	94.54	
Laughter	96.72			
Smile	98.36			

Table 2
The accuracy rates of the system for different choices of the CS3 level where the si-LoC values are selected to form the feature vectors; the L_1-norm was used for matching.

Level	Neutral	Non-neutral	Overall
1	0.7254	0.6174	0.6714
⋮	⋮	⋮	⋮
9	0.9549	0.9344	0.9446
10	*0.9631**	*0.9454**	*0.9542**
11	0.9631	0.9344	0.9487
⋮	⋮	⋮	⋮
30	0.3524	0.2459	0.2991

Algorithm 1. `FindOptimalLevel`

Input: Training set $\mathbf{T} = \{(f_m, c_m)\}_{m=1}^{M}, p \geqslant 1$
Output: Optimal CS3 level l^*
1: Build CS3 stack with L levels
2: **for** each level $l \in \{1,\ldots,L\}$ **do**
3: **for** $m = 1$ to M **do**
4: Get feature vector $\mathbf{x}_m$ from f_m using CS3 values at level l
5: **end for**
6: $\mathbf{X} \leftarrow \{(\mathbf{x}_m, c_m)\}_{m=1}^{M}$
7: $\boldsymbol{\mu} \leftarrow \frac{1}{M}\sum_{m=1}^{M}\mathbf{x}_m$
8: $\boldsymbol{\Sigma} \leftarrow \frac{1}{M}\sum_{m=1}^{M}(\mathbf{x}_m - \boldsymbol{\mu})(\mathbf{x}_m - \boldsymbol{\mu})^{\top}$
9: Build $\mathbf{U}$ from eigendecomposition of $\boldsymbol{\Sigma}$ (Eq. (27))
10: Partition $\mathbf{X}$ into two sets $\mathbf{X}_{train} = \{(\mathbf{x}_r, c_r)\}_{r=1}^{R}$ and $\mathbf{X}_{test} = \{(\mathbf{x}_s, c_s)\}_{s=1}^{S}$
11: **for** each $\mathbf{x}_r \in \mathbf{X}_{train}$ **do**
12: $\mathbf{x}'_r \leftarrow \mathbf{U}^{\top}(\mathbf{x}_r - \boldsymbol{\mu})$
13: **end for**
14: $\mathbf{X}'_{train} \leftarrow \{(\mathbf{x}'_r, c_r)\}_{r=1}^{R}$
15: **for** each $\mathbf{x}_s \in \mathbf{X}_{test}$ **do**
16: $\mathbf{x}'_s \leftarrow \mathbf{U}^{\top}(\mathbf{x}_s - \boldsymbol{\mu})$
17: **end for**
18: $\mathbf{X}'_{test} \leftarrow \{(\mathbf{x}'_s, c_s)\}_{s=1}^{S}$
19: $\text{correct}_l \leftarrow 0$
20: **for** each $(\mathbf{x}'_s, c_s) \in \mathbf{X}'_{test}$ **do**
21: $(\mathbf{x}^*, c^*) \leftarrow \arg\min_{(\mathbf{x}'_r, c_r)\in\mathbf{X}'_{train}} |\mathbf{x}'_s - \mathbf{x}'_r|_p$
22: **if** $c_s = c^*$ **then**
23: $\text{correct}_l \leftarrow \text{correct}_l + 1$
24: **end if**
25: **end for**
26: $\text{correct}_l \leftarrow \frac{\text{correct}_l}{S}$
27: **end for**
28: $l^* \leftarrow \arg\max_{l\in\{1,\ldots,L\}} \text{correct}_l$

The choice of the distance function used by the classifier is another issue that needs to be investigated. We also tested the performance of our recognition system with L_2-norm and the Mahalanobis distance as the metric used by the classifier in Eq. (28). However, on average, the L_1-norm yielded the best results.

In each experiment shown in Table 1, the training and test sets contained 366 and 61 meshes, respectively (each with 3169 vertices). The overall time required to run each experiment was approximately 230 s on a 2.0 GHz Intel CPU: 110 s to read the meshes in the training and compute the feature vectors, 100 s to solve the resulting eigensystem, and 20 s to read and match all the 61 faces in the test set (approximately 0.33 s to read and match each 3D face).

In the following, we compare the performance of our system on GavabDB, against competing methods in the literature. Unfortunately, different authors used different testing procedures when reporting their results. In order to provide a fair comparison, in each case, we use the same testing procedure as the one used by the method against which we are comparing our system.

Mahoor and Abdel-Mottaleb [36] and Berretti et al. [37] use only one of the frontal scans as the training set, while using the remaining scans as test sets. In Table 3, we compare our results with theirs. As can be seen, because of the reduction in the number of scans per subject in the training set, the performance of our system has dramatically reduced when compared to our results in Table 1. However, the overall recognition rate of our system is still slightly better than the other approaches. Also, note that in [36], the faces in the test set, which contained expressions were cropped such that only the eyes and nose regions were used in matching. Moreover, the two approaches do not report on how the performance of their systems are affected when more samples per subject are provided. Therefore, there is no indication that the performances of their systems improve as the number of scans per subject in the training set is increased. However, we show that the performance of our system improves greatly as more samples are provided (Fig. 16). This is a desirable (if not necessary) property, since in most real-world applications, more than one sample per subject is provided in the training set. In fact, the majority of face recognition approaches (e.g., Fisherfaces [38], SVM [39], Bayesian face recognition [31], sparse representation [28,32]) require more than one sample per subject, in order to estimate information about the distribution of the class associated with each subject in the feature space.

Moreno et al. [39] use two types of experiments to evaluate the performance of their PCA and SVM-based 3D face recognition systems. In the "controlled" setting, the test set consists of one frontal scan per subject, while the training set consists of the remaining scans in the dataset. Therefore, the sizes of the test and training sets are 61 and 366, respectively. In the "non-controlled" setting, they create the test set by randomly selecting two (out of 7) scans for each subject, and using the remaining scans for the subjects in the training set. As a result, the test and training sets contain 122 and 305 scans, respectively. We use the same procedure to compare the performance of our system with theirs, and show the results in Table 4. In the non-controlled setting, we repeated the experiment seven times and the results in Table 4 show the average of the experiments; the best and worst performances were 99.18% and 92.62%, respectively. As can be seen, in all cases, our system outperforms the method of [39].

In [28], the authors test the performance of their 3D face recognition system on GavabDB. However, in their

Table 3
Performance accuracy on GavabDB.

Pose	This work (%)	Mahoor and Abdel-Mottaleb [36] (%)	Berretti et al. [37] (%)
Frontal	95.08	95.0	94
Smile	93.44	83.6	85
Laughter	80.33	68.9	81
Random gesture	78.69	63.4	77
Looking down	88.52	85.3	80
Looking up	85.25	88.6	79
Overall	86.89	82.83	84.29

Table 4
Comparison of recognition accuracies of our system with [39].

Approach	Controlled (%)	Non-controlled (%)
This work	98.36	96.02
Moreno (PCA)	82.00	76.20
Moreno (SVM)	90.16	77.90

Table 5
Comparison of the recognition accuracies of our system with [28].

	This work	This work	Li [28]
Gavab dataset			
# Subjects	61	61	61
# Scans/subj.	4	4	4
FRGC dataset			
# Subjects	59	553	59
# Scans/subj.	6	1–30	4
Search space			
# Subjects	120	614	120
# Scans	598	5032	480
Accuracy			
Neutral faces (%)	97.54	98.36	96.67
Non-neutral (%)	95.08	92.90	93.33
Overall (%)	96.07	95.08	94.68

experiments, they extend the size of the dataset by adding 59 additional 3D faces from the FRGC dataset, while omitting the "looking up" and "looking down" scans from the Gavab dataset. The performance of the system was then tested by running five different sets of experiments. In each experiment, the test set consisted of 61 scans (1 scan per subject) from the 5 different groups of scans (two sets of frontal scans, 1 set with random gestures, 1 set with laughter and another set with smile), while the training set consisted of the remaining scans in the dataset. The recognition results were then grouped into two classes. The frontal scans formed the "neutral" class, while the other scans (with random gesture, laughter, and smile) formed the "non-neutral" class. The recognition accuracy for each class was then computed as the average of the recognition results of its members. In Table 5, we compare the performance of our system with [28]. Note that we conducted two sets of experiments. In the first set of experiments, we followed the same procedure as in [28], but added two additional scans for each subject from the FRGC dataset. This increased the size of the search space by 118 scans from faces *not* in the test set, and hence made the recognition task even more difficult. The second column of Table 5 shows the results for this set of experiments. As can be seen, our approach outperforms the approach of [28]. To make the recognition task even more challenging, we added 4788 scans from the FRGC dataset to the training set, while keeping the number of scans from the Gavab dataset the same as before, and followed the same procedure as before to measure the accuracy rate of our recognition system. The third column of Table 5 shows the results of this experiment. As can be seen, while the accuracy rate of our system for non-neutral faces becomes slightly lower

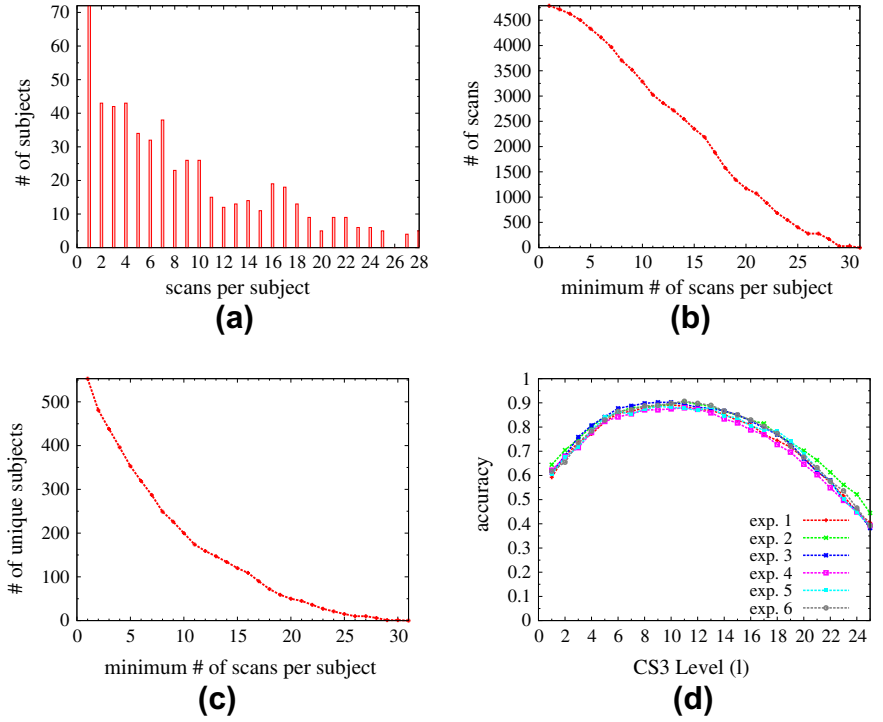


Fig. 15. (a) Histogram of the number of scans per subject for the set of 3D faces we use from the FRGC dataset; (b) dataset size as a function of the minimum number of scans per subject; (c) subject count as a function of the minimum number of scans per subject; (d) accuracy rates on FRGC dataset as functions of the CS3 levels used to construct feature vectors.

than [28], our overall accuracy still remains higher than that of [28].

We also tested our recognition system on 4788 3D faces from the FRGC dataset, which corresponded to 553 individuals. Unlike GavabDB, in the FRGC dataset, the number of scans for all individuals (subjects) was not the same. In Fig. 15a, we plot the histogram of the number of scans per subject in the dataset that we used in our experiments. For example, 72 subjects had only 1 scan, and 43 subjects had 2 scans. In Fig. 15b, we show how the size of the dataset decreases as we increase the required minimum number of scans per subject. For example, the total number of scans in the dataset decreases to 4716, when only subjects with at least two scans are considered, while the dataset size becomes 3286, when only subjects with at least 10 scans are kept. In Fig. 15c, we show how the number of subjects in the dataset decreases as the required minimum number of scans per subject is increased. For example, the number of subjects decreases to 481 and 200, when the minimum required number of scans are set to 2 and 10, respectively. As is shown in the following experiments, the minimum number of scans per subject used in the training set affects the performance of our recognition system, even though our PCA-based system does not explicitly attempt to recover information about the class conditional probability density function of each face class in the feature space.

In Fig. 15d, we show how the accuracy of the system for the FRGC dataset is influenced by the choice of the CS3 level used when constructing the feature vectors. As can be seen, again the optimal performance is achieved approximately at level $l = 10$ (with average accuracy rate of 89.05%). Therefore, throughout all our experiments, we used 10 CS3 levels to construct the required feature vectors. In the six experiments conducted to obtain the plots in Fig. 15d, we used a subset of the scans in the dataset, which contained at least two scans per subject. This enabled us to partition the dataset into two disjoint sets to obtain the training and test sets. In each experiment, the test set was constructed by randomly selecting one scan for each individual in the dataset, and the remaining scans were used as the training set.

We argue that the decreased accuracy of the system for the FRGC dataset (compared to GavabDB) is due to the large number of subjects in the training set with small number of scans, and that the increased size of the search space has a smaller influence on the performance of the system. All (100%) of the subjects in the GavabDB experiments had 6 scans in the training set, whereas in FRGC only 287 out of 553 (51.9%) of the subjects had at least that many scans. In Fig. 16a, we show how the performance of the system is improved as the minimum number of scans per subject is increased. The graph plots the average and standard deviation of the accuracy rate of the system

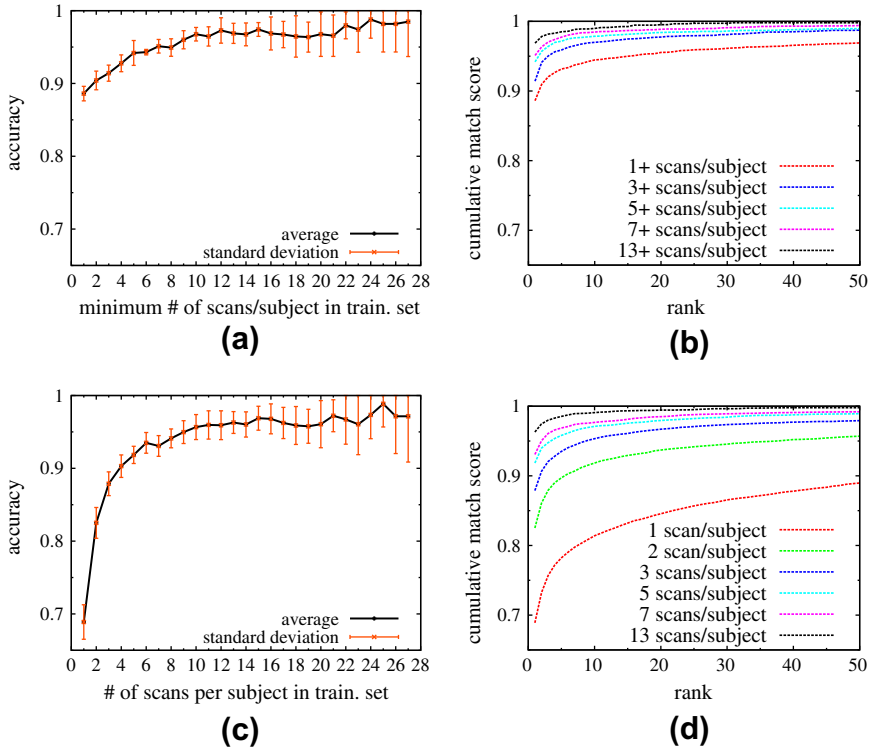


Fig. 16. Accuracy and CMC curves of our recognition system for FRGC dataset (1) (a) Average accuracy rate as a function of the minimum number of scans per subject in the training set; the red bars show the standard deviation of the results in the experiments. (b) Cumulative Match Characteristic curves for different minimum numbers of scans per subject. (c) Accuracy rate as a function of the number of scans per subject. (d) Cumulative Match Characteristic curves for different numbers of scans per subject. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

for a set of 11 experiments where disjoint training and test sets were randomly constructed. As can be seen, the accuracy of the system increases as the minimum required number of scans per subject is increased. The average accuracy rate of the system on a subset of the FRGC dataset where each subject has at least six scans in the training set is 94.30%. The training set for this subset contains 287 subjects, and a total of 3683 scans. The performance of system is further increased to 96.00% when all subjects in the training set have at least nine scans (200 subjects, and a total of 3086 scans in the training set). Fig. 16b shows the averaged Cumulative Match Characteristic (CMC) curves of our system for different minimum number of scans per subject in the dataset.

In Fig. 16c and d, we plot the average accuracy rates and CMC curves of our system for a set of 35 experiments. However, for these experiments, a *fixed* number of scans per subject was used in each experiment—instead of a minimum number of scans; e.g., the red curve in Fig. 16d, plots the average CMC curve of 35 experiments, where in each experiment, the training set was constructed by randomly selecting *only one* scan from each subject in the FRGC dataset and using the remaining scans in the test set. As the graphs show, the performance of the system improves dramatically as the number of scans per subject increases in the training set. For example, increasing the number of scans for each subject in the training set from 1 to 3, improves the accuracy rate of the system by approximately 20%.

4. Conclusion

We presented a new scale-space based representation for 3D surfaces that was shown to be useful for feature extraction and shape matching. We showed our proposed representation to be robust to noise and capable of automatic scale selection. The major benefits of our approach over existing methods such as [20,21] are automatic scale selection, improved computational efficiency, lower memory requirements, and ease of implementation. We compared the performance of our CS3-based keypoint extractor with competing methods, such as [19,18,26], and showed that it was able to outperform the other methods in the majority of cases in terms of speed, relative and absolute repeatability. We also demonstrated an application of our CS3 representation to 3D face recognition, where our proposed scale-invariant Laplacian of surface curvatures (si-LoC) was employed to form feature vectors for measuring the dissimilarity between the faces. We tested the performance of the recognition system on two well-known 3D face datasets, and showed its better performance over state-of-the-art 3D face recognition systems.

Acknowledgments

The 3D models shown in this paper are courtesy of the AIM@Shape Shape Repository. We would like to thank Federico Tombari and his team at University of Bologna for making their performance evaluation tools for 3D keypoint extractors available online. This work was partly supported by CUNY Graduate Center's Technology Fellowship program and the NSF (STTR Award #0750485).

References

[1] Tangelder, Veltkamp, A survey of content based 3d shape retrieval methods, in: SMI '04: Proc. of the Shape Modeling International 2004, 2004, pp. 145–156.
[2] R. Gal, D. Cohen-Or, Salient geometric features for partial shape matching and similarity, ACM Transactions on Graphics 25 (1) (2006) 130–150.
[3] Y. Yang, H. Lin, Y. Zhang, Content-based 3-D model retrieval: a survey, IEEE Transactions on Systems, Man and Cybernetics 37 (6) (2007) 1081–1098.
[4] Tony Lindeberg, Scale-Space Theory in Computer Vision, Monograph 1994.
[5] T. Lindeberg, L. Bretzner, Real-time scale selection in hybrid multi-scale representations. Springer Verlag, 2003, pp.148–163.
[6] A.P. Witkin, Scale-space filtering, in: Proc. of the Eighth Intl. Joint Conf. Artificial Intelligence, 1983, pp. 1019–1022.
[7] T. Lindeberg, Scale-Space Theory in Computer Vision, Kluwer Academic Publishers, Norwell, MA, USA, 1994.
[8] D.G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2004) 91–110.
[9] M. Brown, D.G. Lowe, Automatic panoramic image stitching using invariant features, International Journal of Computer Vision 74 (1) (2007) 59–73, http://dx.doi.org/10.1007/s11263-006-0002-3.
[10] H. Bay, A. Ess, T. Tuytelaars, L.V. Gool, Speeded-up robust features (surf), Computer Vision and Image Understanding 110 (3) (2008) 346–359 (similarity Matching in Computer Vision and Multimedia).
[11] F. Mokhtarian, A.K. Mackworth, A theory of multiscale, curvature-based shape representation for planar curves, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (8) (1992) 789–805.
[12] M. Desbrun, M. Meyer, P. Schröder, A.H. Barr, Implicit fairing of irregular meshes using diffusion and curvature flow, in: SIGGRAPH '99, 1999, pp. 317–324.
[13] M. Schlattmann, P. Degener, R. Klein, Scale space based feature point detection on surfaces, Journal of WSCG 16 (2008) 1–3.
[14] J. Novatnack, K. Nishino, A. Shokoufandeh, Extracting 3D Shape Features in Discrete Scale-Space, in: 3DPVT06, 2006, pp. 946–953.
[15] M. Novotni, P. Degener, R. Klein, Correspondence Generation and Matching of 3d Shape Subparts, Tech. Rep., Universitt Bonn, 2005.
[16] U. Castellani, M. Cristani, S. Fantoni, V. Murino, Sparse points matching by combining 3d mesh saliency with statistical descriptors, Computer Graphics Forum 27 (2) (2008) 643–652.
[17] C.H. Lee, A. Varshney, D.W. Jacobs, Mesh saliency, ACM Transactions on Graphics 24 (3) (2005) 659–666.
[18] A. Zaharescu, E. Boyer, K. Varanasi, R. Horaud, Surface feature detection and description with applications to mesh matching, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, 2009, pp. 373–380.
[19] R. Unnikrishnan, M. Hebert, Multi-scale interest regions from unorganized point clouds, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPRW '08, 2008, pp. 1–8. http://dx.doi.org/10.1109/CVPRW.2008.4563030.
[20] J. Sun, M. Ovsjanikov, L. Guibas, A concise and provably informative multi-scale signature based on heat diffusion, in: Eurographics Symposium on Geometry Processing (SGP), 2009.
[21] A.M. Bronstein, M.M. Bronstein, L.J. Guibas, M. Ovsjanikov, Shape google: geometric words and expressions for invariant shape retrieval, ACM Transactions on Graphics 30 (2011) 1:1–1:20.
[22] A. Vaxman, M. Ben-Chen, C. Gotsman, A multi-resolution approach to heat kernels on discrete surfaces, ACM Transactions on Graphics (TOG) 29 (4) (2010) 121.
[23] F. Tombari, S. Salti, L. DiStefano, Performance evaluation of 3d keypoint detectors, International Journal of Computer Vision (2012) 1–23, http://dx.doi.org/10.1007/s11263-012-0545-4.
[24] H. Chen, B. Bhanu, 3d free-form object recognition in range images using local surface patches, Pattern Recognition Letters 28 (10) (2007) 1252–1262, http://dx.doi.org/10.1016/j.patrec.2007.02.009.
[25] Y. Zhong, Intrinsic shape signatures: a shape descriptor for 3d object recognition, in: 2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops), 2009, pp. 689–696. http://dx.doi.org/10.1109/ICCVW.2009.5457637.
[26] A. Mian, M. Bennamoun, R. Owens, On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes, International Journal of Computer Vision 89 (2–3) (2010) 348–361.

[27] M. Garland, P.S. Heckbert, Surface simplification using quadric error metrics, in: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., 1997, pp. 209–216.
[28] X. Li, T. Jia, H. Zhang, Expression-insensitive 3d face recognition using sparse representation, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2009.
[29] A. Scheenstra, A. Ruifrok, R. Veltkamp, A survey of 3d face recognition methods, in: Audio- and Video-Based Biometric Person Authentication, vol. 3546, 2005, pp. 891–899.
[30] A.F. Abate, M. Nappi, D. Riccio, G. Sabatino, 2d and 3d face recognition: a survey, Pattern Recognition Letters 28 (14) (2007) 1885–1906, http://dx.doi.org/10.1016/j.patrec.2006.12.018 (Image: Information and Control).
[31] B. Moghaddam, T. Jebara, A. Pentland, Bayesian face recognition, Pattern Recognition 33 (11) (2000) 1771–1782.
[32] J. Wright, A. Yang, A. Ganesh, S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2) (2009) 210–227, http://dx.doi.org/10.1109/TPAMI.2008.79.
[33] L. Sirovich, M. Kirby, Low-dimensional procedure for the characterization of human faces, Journal of the Optical Society of America A 4 (3) (1987) 519–524.
[34] A. Moreno, A. Sanchez, GavabDB: a 3D face database, in: Proc. 2nd COST Workshop on Biometrics on the Internet: Fundamentals, Advances and Applications, 2004, pp. 77–82.
[35] P.J. Flynn, FRGC Database v2.0, 2003. <http://bbs.bee-biometrics.org/>.
[36] M.H. Mahoor, M. Abdel-Mottaleb, Face recognition based on 3d ridge images obtained from range data, Pattern Recognition 42 (2009) 445–451.
[37] S. Berretti, A. Del Bimbo, P. Pala, 3d face recognition by modeling the arrangement of concave and convex regions, in: Adaptive Multimedia Retrieval: User, Context, and Feedback, vol. 4398, 2007, pp. 108–118.
[38] P.N. Belhumeur, J.a.P. Hespanha, D.J. Kriegman, Eigenfaces vs. fisherfaces: recognition using class specific linear projection, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (1997) 711–720.
[39] A. Moreno, A. Sanchez, J. Velez, J. Diaz, Face recognition using 3d local geometrical features: Pca vs. svm, in: Image and Signal Processing and Analysis, 2005. ISPA 2005. Proc. of the 4th International Symposium on, 2005, pp. 185 – 190. doi:10.1109/ISPA.2005.195407.