Proceedings of the OAGM Workshop 2018 DOI: 10.3217/978-3-85125-603-1-14

Fast Solvers for Solving Shape Matching by Time Integration

Martin Bähr, Robert Dachsel and Michael Breuß

Abstract— **The main task in three-dimensional non-rigid shape correspondence is to retrieve similarities between two or more similar three-dimensional objects. An important building block of many methods constructed to achieve this goal is a simplified shape representation called feature descriptor, which is invariant under almost isometric transformations. A recent feature descriptor relies on the full numerical integration of the geometric heat equation. This approach involves to solve a system of linear equations with multiple right-hand sides. To this end, it is necessary to find a fast and accurate numerical scheme in conjunction with the solution of a sparse linear system and many different right sides. In this paper we evaluate direct, iterative and model order reduction (MOR) methods and their influence to shape correspondence applications which will be validated on standard shape data sets with different resolutions.**

I. INTRODUCTION

The examination of shape correspondence is a fundamental task in computer vision, pattern recognition and geometry processing. Performing the shape correspondence process is a key component for problems such as 3D scan alignment or space-time reconstruction and is essential in various applications including shape interpolation and statistical modelling, see [21]. The fundamental task of shape correspondence is to identify an explicit relation between the elements of two or more given shapes, whereby the shape of a three-dimensional geometric object can be described by its bounding surface. In this context, a challenging setting is that of non-rigid shape correspondence, where the shapes are almost isometric. Almost isometric shapes lead to a large variety of possible deformations such as poses of human or animal bodies.

An important solution strategy is to achieve a pointwise non-rigid shape correspondence using so called descriptor based methods. For this, a feature descriptor has to be developed which characterizes each element on the shape regarding its geometric relation. An interesting type of descriptors is based on the spectral decomposition of the Laplace-Beltrami operator, see e.g. [17], [20]. However, these methods rely on the expansion of eigenfunctions of the Laplace-Beltrami operator, which is for instance used to approximate the solution of the geometric heat equation, cf. [2]. A recent alternative compared to eigenfunction expansion methods is based on the full numerical integration of the underlying partial differential equations (PDEs), cf. [3]. Experiments based on time integration methods confirm a substantially higher accuracy compared to spectral methods in many cases.

Institute for Mathematics, Brandenburg Technical University, Platz der Deutschen Einheit 1, 03046, Cottbus, Germany {baehr,dachsel,breuss}@b-tu.de

Application of time integration methods leads to a new non-negligible challenge – solving a system of linear equations with multiple right-hand sides. Dealing with large sparse systems implicates two main issues, the accuracy of the solution and the computational efficiency of the numerical solver. Generally, direct and iterative methods are the most common solvers to compute the solution. However, almost always it remains an open question, which of both is the best choice to solve the underlying problem. Direct methods for solving the same system for different right sides, are fast and offer an extremely high accuracy. However, this type of solvers may use substantial memory and appears to be rather impractical for shapes with many thousands of points. In contrast, iterative methods are naturally not tweaked for extremely high accuracy but are very fast in computing approximate solutions. They require less memory space and are thus inherently more attractive candidates for this application. Nevertheless, the runtime of iterative methods depends on the data, size, sparsity and required accuracy and makes a tool that is not straightforward to use.

Let us mention, that the number of the right-hand sides of the underlying system are directly related to the number of points of the regarded shapes. Therefore, the increase of the size of the point cloud defining a shape leads to an extreme rise of the computational effort. Due to this fact, an alternative approach is the use of model order reduction. Such techniques can be used to approximate the full system by a significantly reduced model, that is much faster to solve than the original system. In this work, we consider the modal coordinate reduction (MCR), which involves the use of projection matrices to approximate the geometric heat equation. The accuracy of MCR for a given problem depends on the number and structure of equations. In case of shape matching we compare the correspondence of several shapes. Therefore, a correct matching depends on a good numerical quality of the physical process on each of the regarded shapes. For this reason, the application of the MCR method could lead to a more sensitive result with respect to the quality of the matching.

Our Contributions. In this work, we address the mentioned challenges by investigating numerical methods for computing feature descriptors based on time integration methods. To this end, we will compare direct, iterative and MCR methods in terms of accuracy and computational efficiency for shape matching by application of the classic feature descriptor defined via the geometric heat equation.

II. THEORETICAL BACKGROUND

In this section, we introduce the basic facts that are necessary to define the shape matching framework.

A. Almost Isometric Shapes

The shape of a three-dimensional geometric object can be described by its bounding surface. This is a two-dimensional curved object, embedded into a three-dimensional Euclidean space. In this paper, we model shapes as compact two-dimensional Riemannian manifolds $\mathscr{M}$, equipped with metric tensor $g \in \mathbb{R}^{2\times 2}$.

In this setting, two shapes $\mathscr{M}$ and $\tilde{\mathscr{M}}$ may be considered as isometric if there is a map T that unfolds one surface onto the other by preserving the intrinsic distance. From the mathematical point of view there exists a smooth homeomorphism $T : \mathscr{M} \to \tilde{\mathscr{M}}$ with

$$d_{\mathscr{M}}(x_1,x_2) = d_{\tilde{\mathscr{M}}}(T(x_1),T(x_2)) \quad \forall x_1,x_2 \in \mathscr{M} \tag{1}$$

where $d_{\mathscr{M}}(x_1,x_2)$ is the intrinsic distance. The intrinsic distance can be thought as the shortest curve along the surface $\mathscr{M}$ connecting x_1 and x_2.

The notion of purely isometric shapes appears too restrictive. Many shapes include an additional small elastic deformation. These occur by either the transformation itself, noisy datasets or by transferring the continuous shape into a numerical framework. This acts as a "small" distortion for the intrinsic distance. We call two shapes $\mathscr{M}$ and $\tilde{\mathscr{M}}$ almost isometric, if there exists a transformation $S : \mathscr{M} \to \tilde{\mathscr{M}}$ with

$$d_{\mathscr{M}}(x_1,x_2) \approx d_{\tilde{\mathscr{M}}}(S(x_1),S(x_2)) \quad \forall x_1,x_2 \in \mathscr{M} \tag{2}$$

B. The Heat Equation on Shapes

The heat equation that yields a useful descriptor involves the Laplace operator when considering the Euclidean plane. In order to respect the curvature of a manifold in 3D, techniques from differential geometry are employed [5]. The resulting Laplace-Beltrami operator is defined on a smooth manifold $\mathscr{M}$. In this context, let us recall that for a given parameterisation of a two-dimensional manifold, the Laplace-Beltrami operator applied to a scalar function $u : \mathscr{M} \to \mathbb{R}$ can be expressed in local coordinates:

$$\Delta_{\mathscr{M}} u = \frac{1}{\sqrt{|g|}} \sum_{i,j=1}^{2} \partial_i \left(\sqrt{|g|} g^{ij} \partial_j u \right) \tag{3}$$

where g^{ij} are the components of the inverse of the metric tensor and $|g|$ is its determinant.

The *geometric heat equation* describes how heat would propagate along a surface $\mathscr{M}$ and can be formulated as

$$\begin{cases} \partial_t u(x,t) = \Delta_{\mathscr{M}} u(x,t) & x \in \mathscr{M}, t \in I \\ u(x,0) = \exp\left(-\frac{d_{\mathscr{M}}(x-x_i)^2}{2\sigma^2}\right) & \\ \langle \nabla_{\mathscr{M}} u, \mathbf{n} \rangle = 0 & x \in \partial\mathscr{M} \end{cases} \tag{4}$$

where the initial condition $u(x,0)$ is a given by a highly peaked Gaussian heat distribution. In this context σ^2 is the variance parameter and $x_i \in \mathscr{M}$ is the centre of the Gaussian distribution. Many shapes appear as a closed manifold with $\partial\mathscr{M} = \emptyset$, where boundary conditions do not appear. For the case $\mathscr{M}$ has boundaries, we additionally require u to satisfy homogeneous Neumann boundary conditions, where $\mathbf{n}$ is the normal vector to the boundary.

C. The Feature Descriptor and Shape Correspondence

a) Feature Descriptor: Considering the surface itself is unsuitable for many shape analysis tasks. A simplified representation is needed which is often called a feature descriptor. In this context, the feature descriptor stores the geometry of the surface at a certain local region. We restrict the spatial component of $u(x,t)$ to

$$f_{x_i}(t) = u(x,t)_{|x=x_i} \text{ with } u(x,0) = \exp\left(-\frac{d_{\mathscr{M}}(x-x_i)^2}{2\sigma^2}\right) \tag{5}$$

and call the f_{x_i} the feature descriptors at the location $x_i \in \mathscr{M}$. Let us note that there exists a physical interpretation of the feature descriptors. The heat based feature descriptor describes the rate of heat transferred away from the considered point x_i. Since we used an intrinsic approach, let us note, the feature descriptor can not distinguish between intrinsic symmetry groups.

b) Shape Correspondence: To compare the feature descriptors for different locations $x_i \in \mathscr{M}$ and $\tilde{x}_j \in \tilde{\mathscr{M}}$ on respective shapes $\mathscr{M}$ and $\tilde{\mathscr{M}}$, we simply define a distance $d_f(x_i,\tilde{x}_j)$ using the L_1 norm

$$d_f(x_i,\tilde{x}_j) = \int_I |f_{x_i} - f_{\tilde{x}_j}| \, \mathrm{d}t \tag{6}$$

The tuple of locations $(x_i,\tilde{x}_j)$ with the smallest feature distance should belong together. This condition can be written as a minimisation problem for all locations:

$$(x_i,\tilde{x}_j) = \underset{\tilde{x}_k \in \tilde{\mathscr{M}}}{\arg\min}\, d_f(x_i,\tilde{x}_k) \tag{7}$$

By using $\tilde{x}_j = S(x_i) = x_i$, the map S can pointwise be restored for all x_i. Let us mention, that without further alignment the restored map S is neither injective nor surjective because the minimisation condition is not unique.

III. DISCRETISATION ASPECTS

As indicated we will construct a feature descriptor by direct discretisation of the geometric heat equation. In order to approximate the equation on a shape we have to take care of three aspects. First, a discrete approximation of our continuous and closed surface as well as of the time domain is needed. Second, a suitable discrete Laplace-Beltrami operator has to be defined. Third, quadrature formulas need to be used to approximate the time integration.

We start with the integrated form of the geometric heat equation in (4) over time and space

$$\int_I \int_{\mathscr{M}} \partial_t u(x,t) \, \mathrm{d}x \, \mathrm{d}t = \int_I \int_{\mathscr{M}} \Delta_{\mathscr{M}} u(x,t) \, \mathrm{d}x \, \mathrm{d}t \tag{8}$$

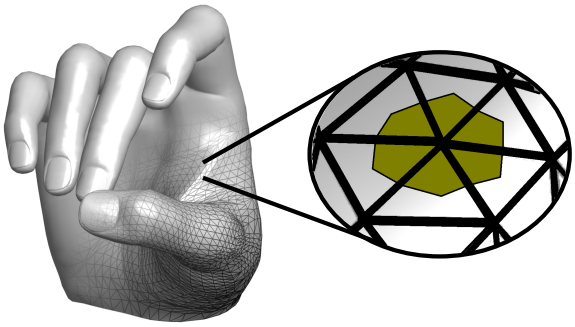

Fig. 1. Continuous and discrete representation of a shape. The discrete approximation of a shape is given by non-uniform and linear triangles. The construction of volume cells is done by using the barycentric area around a vertex.

A. Discrete Space and Time Domain

A suitable surface representation is given by a triangular mesh, cf. Figure 1. In more detail, a triangulated surface is given by the tuple $\mathscr{M}_d = (P,T)$. The point cloud $P := \{x_1,\dots,x_N\}$ contains the finite number of vertices (given as coordinate points) a shape consists of. The entire mesh can be formed by connecting the vertices x_i so that one obtains two-dimensional triangular cells. Therefore, the set of linear triangles T contains the neighborhood relations between vertices forming a triangle.

Further, we sub-divide the meshed surface and the time axis as follows:

$$\mathscr{M}_d = \bigcup_{i=1}^{N} \Omega_i \quad \text{and} \quad I = \bigcup_{i=1}^{M} I_k \tag{9}$$

where Ω_i is the barycentric cell volume surrounding the i-th vertex. For time, let $I_k = [t_{k-1}, t_k]$ and $t_0 = 0$, where the time increment $\tau = t_k - t_{k-1}$ for all $k \in \{1,\dots,M\}$ is uniformly chosen.

B. Finite Volume Approach

Now we restrict the integrated geometric heat equation to Ω_i and I_k. For values $x \in \Omega_i$ and $t \in I_k$ we have

$$\int_{I_k}\int_{\Omega_i} \partial_t u(x,t)\,\mathrm{d}x\,\mathrm{d}t = \int_{I_k}\int_{\Omega_i} \Delta_{\mathscr{M}}\, u(x,t)\,\mathrm{d}x\,\mathrm{d}t \tag{10}$$

Further we use the definition of the cell average

$$u_i(t) = u(\bar{x}_i,t) = \frac{1}{|\Omega_i|}\int_{\Omega_i} u(x,t)\,\mathrm{d}x \tag{11}$$

where $|\Omega_i|$ is the surface area of the i-th cell. Now, we apply the divergence theorem to substitute the volume integral on the right-hand side into a line integral over the boundary of the volume cell to define the averaged Laplacian of the i-th cell

$$Lu_i(t) = \frac{1}{|\Omega_i|}\int_{\Omega_i} \Delta_{\mathscr{M}}\, u(x,t)\,\mathrm{d}x \tag{12}$$

A function defined on all cells is represented by an N-dimensional vector $\mathbf{u}(t) = (u_1(t),\dots,u_N(t))^\top$. Using the mean value expression for equation (10), we obtain a system of integrated ODEs:

$$\int_{I_k} \partial_t \mathbf{u}(t)\,\mathrm{d}t = \int_{I_k} L\mathbf{u}(t)\,\mathrm{d}t \tag{13}$$

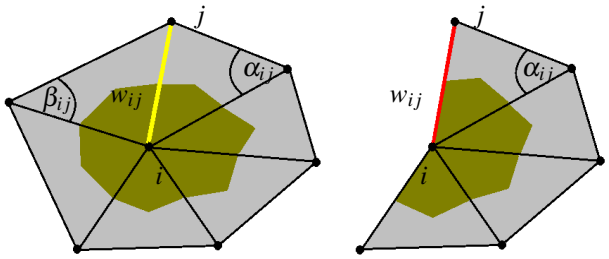

Fig. 2. The cotangent weight scheme as discretisation of the Laplace-Beltrami operator. Left: interior edge Right: boundary edge

C. Discrete Spatial Operator

Many schemes have been proposed to estimate the Laplace-Beltrami operator for a triangular meshed surface [13], [15]. A commonly used method is the cotangent weight scheme introduced in [11]. As a result, for a function $\mathbf{u}$ defined on a triangular mesh the discrete Laplace-Beltrami operator $L \in \mathbb{R}^{N\times N}$ reduces to the following sparse matrix representation

$$L = D^{-1}W \tag{14}$$

The symmetric weight matrix W reads component-wise

$$W_{ij} = \begin{cases} -\sum\limits_{j\in N_i} w_{ij} & \text{if } i = j \\ w_{ij} & \text{if } i \neq j \text{ and } j \in N_i \\ 0 & \text{else} \end{cases} \tag{15}$$

where N_i denotes the set of points adjacent to x_i. The weights w_{ij} of the edge (i,j) can be classified into interior E_i and boundary edges E_b respectively

$$w_{ij} = \begin{cases} \frac{\cot\alpha_{ij}+\cot\beta_{ij}}{2} & \text{if } (i,j) \in E_i \\ \frac{\cot\alpha_{ij}}{2} & \text{if } (i,j) \in E_b \end{cases} \tag{16}$$

as shown in Figure 2. Furthermore, α_{ij} and β_{ij} denote the two angles opposite to the edge (i,j), for details we refer to the mentioned source. The Matrix

$$D = \mathrm{diag}\,(|\Omega_1|,\dots,|\Omega_i|,\dots,|\Omega_N|) \tag{17}$$

contains the local cell areas. Let us note that L is not symmetric after computing the matrix product.

D. Discrete Time Integration

Discrete time integration of ODEs can be done by using standard numerical methods through time step methods. Common time integration schemes are the explicit Euler method, the implicit Euler method and the trapezoidal rule known as Crank-Nicolson method. The Crank-Nicolson method is also an implicit method, however it is second-order in time and will only produce slightly more computational

cost than the implicit Euler method. For this reason, we consider only the explicit Euler method and the Crank-Nicolson method.

a) Explicit Euler Method: As a first step we apply the fundamental lemma of calculus for the left-hand-side of (13)

$$\int_{I_k} \partial_t \mathbf{u}(t)\,\mathrm{d}t = \int_{t_{k-1}}^{t_k} \partial_t \mathbf{u}(t)\,\mathrm{d}t = \mathbf{u}(t_k) - \mathbf{u}(t_{k-1}) \tag{18}$$

Finally we approximate the integral on the right-hand side by using the rectangle method

$$\int_{t_{k-1}}^{t_k} L\mathbf{u}(t)\,\mathrm{d}t \approx \tau L\mathbf{u}(t_{k-1}) \tag{19}$$

and by using the notation $\mathbf{u}(t_k) = \mathbf{u}^k$ we obtain

$$\mathbf{u}^k = (I + \tau L)\,\mathbf{u}^{k-1} \tag{20}$$

where $k \in \{1,\dots,M\}$ and $\mathbf{u}^0 = u_0$. Due to the fact that the values of u^{k-1} are known, we can easily compute the corresponding values u^k at time k by simple matrix-vector multiplication. This explicit scheme is known to be just conditionally stable, see [19]. The stability requirement yields a limitation on the size of the time step τ.

b) Crank-Nicolson Method: If we apply the trapezoidal rule at the integral of the right-hand-side of (13)

$$\int_{t_{k-1}}^{t_k} L\mathbf{u}(t)\,\mathrm{d}t \approx \frac{\tau}{2}\left(L\mathbf{u}(t_k) + L\mathbf{u}(t_{k-1})\right) \tag{21}$$

we obtain finally

$$\left(I - \tfrac{\tau}{2}L\right)\mathbf{u}^k = \left(I + \tfrac{\tau}{2}L\right)\mathbf{u}^{k-1} \tag{22}$$

To compute the values u^k at time k it requires solving a system of linear equations as well as a matrix-vector multiplication in each time step. Therefore, it is numerically more intensive than the explicit Euler method, however it has second-order accuracy in time. The considerable advantage of an implicit scheme is the numerical stability independently of the time step size τ, cf. [19]. However, the Crank-Nicolson method is sensitive for problems with discontinuous initial conditions.

IV. Numerical Solvers

An essential key requirement for a correct shape matching is a sufficient accuracy of the computed numerical solution. However, the geometric heat equation has to be solved for each point and on each shape for a fixed time interval $t \in (0, t_M]$. Consequently, the computational costs are directly related to the number of points of the regarded shapes. This fact suggests that one may forego high accuracy in exchange for a faster computational time. Therefore, a qualitative analysis of numerical methods for the geometric heat equation in context to shape matching is absolutely essential.

As seen in the last section, the temporal integration can either be done explicitly or implicitly. For both approaches there exist several numerical solvers, which have different advantages in terms of computational effort and accuracy of the computed solution. In the following, we give a short overview.

A. Explicit Methods

Explicit schemes are simple iterative schemes of the form $u^k = (I + \tau L)u^{k-1}$ such as (20). The typical time step restriction has a rather small upper bound and makes these methods unsuitable for shape matching. An alternative is the usage of the Fast Explicit Diffusion (FED) or Fast Semi-Iterative (FSI) scheme, which is well-known in image processing. For a detailed presentation of FED or FSI we refer to [6], [7]. The core idea behind FSI is to consider an explicit scheme and interleave time steps that significantly violate the upper stability bound with small stabilising steps. To decrease numerical rounding errors and simultaneously increase the approximation quality, FSI uses cycles of varying time steps. The cyclic FSI scheme, which accelerates the explicit diffusion scheme (20), for the m-th cycle with cycle length n is given by

$$u^{m,k} = \alpha_k \cdot (I + \tau L)u^{m,k-1} + (1 - \alpha_k) \cdot u^{m,k-2} \tag{23}$$

$$n = \left\lceil \sqrt{\frac{3t_M}{\tau_{\max} \cdot C} + \frac{1}{4}} - \frac{1}{2} \right\rceil, \quad \tau = \frac{3t_M}{C \cdot n(n+1)} \tag{24}$$

$$\alpha_k = \frac{4k+2}{2k+3}, \quad u^{m,-1} = u^{m,0}, \quad k = 1,\dots,n \tag{25}$$

where C is the number of outer FSI cycles, t_M the diffusion time and $\tau_{\max}$ the theoretical upper bound for a stable explicit finite difference scheme. The FSI scheme can be applied whenever the matrix L in (20) is negative semidefinite and symmetric. The underlying matrix L is not symmetric, however by multiplication of D to equation (20) we have

$$D\mathbf{u}^k = (D + \tau W)\mathbf{u}^{k-1} \tag{26}$$

where W is symmetric and negative semidefinite.

B. Implicit Methods

Implicit schemes result in a linear system of equations (compare (22)) and lead theoretically to an unconditionally stable scheme without a time step restriction. However, solving linear equations requires significant computational effort and therefore a fast solver for large sparse linear systems of equations is necessary.

Standard methods for solving linear systems are direct and iterative solvers. Direct methods compute highly accurate solutions and are predestined for solving a system with multiple right-hand sides. In that case, the underlying matrix will be factorised one-time, and subsequently each right hand side is solved by forward and backward substitution. Due to the fact that the underlying system matrix is sparse the computational costs for the substitution step will be at most $\mathcal{O}(n^2)$, where n is number of equations. In contrast, iterative solvers can compute approximate solutions in a very fast way. A particular class of iterative solvers designed for use with large sparse linear systems is the class of *Krylov*

subspace solvers; for a detailed exposition see [18]. The main idea behind the Krylov approach is to search for an approximative solution of $A\mathbf{x} = \mathbf{b}$, with $A \in \mathbb{R}^{n\times n}$ a large regular sparse matrix and $\mathbf{b} \in \mathbb{R}^n$, in a suitable low-dimensional subspace $\mathbb{R}^l$ of $\mathbb{R}^n$ that is constructed iteratively with l being the number of iterates. The aim in the construction is thereby to have a good representation of the solution after a few iterates. Let us note that this construction is often not directly visible in the formulation of a Krylov subspace method.

We propose to employ the well-known conjugate gradient (CG) scheme of Hestenes and Stiefel [8], which is still an adequate iterative solver for problems involving sparse symmetric and positive definite matrices. Let us note, if we multiply the matrix D to equation (22), we obtain

$$\left(D - \tfrac{\tau}{2}W\right)\mathbf{u}^k = \left(D + \tfrac{\tau}{2}W\right)\mathbf{u}^{k-1} \tag{27}$$

such that $\left(D - \frac{\tau}{2}W\right)$ is symmetric positive definite. With $A = \left(D - \frac{\tau}{2}W\right)$, $b = \left(D + \frac{\tau}{2}W\right)u^{k-1}$ and $x = u^k$ the latter equation corresponds to solving a system $Ax = b$ at time k. For the CG method, one can show that the approximate solutions $\mathbf{x}_l$ (in the l-th iteration) are optimal in the sense that they minimise the so-called energy norm of the error vector. In other words, the CG method gives in the l-th iteration the best solution available in the generated subspace [10]. Since the dimension of the Krylov subspace is increased in each iteration, theoretical convergence is achieved at latest after the n-th step of the method if the sought solution is in $\mathbb{R}^n$. The CG algorithm requires in each iteration a sparse matrix-vector-multiplication. One main advantage is that a practical solution can be reached quickly after a small number l of iterations, which yields to a quick termination of the CG method and total costs of at most $\mathcal{O}(ln^2)$. However, in practice numerical rounding errors appear and one may suffer from convergence problems for very large systems. Thus, a *preconditioning* is recommended to enforce all the beneficial properties of the algorithm, along with fast convergence [1]. Moreover, it may require fine-tuning of parameters in the preconditioned conjugated gradient method (PCG) and in addition increase potentially the computational cost.

C. Model Order Reduction

The introduced explicit and implicit methods have to handle large sparse systems, whereby the computational costs depends on the point cloud size. Model Order Reduction (MOR) techniques can be used to approximate the full ODE system by a very low dimensional system, while preserving the main characteristic of the original ODE system. Existing MOR techniques [12], [14] are polynomial, projection and non-projection based methods. In this work, we apply the widely used modal coordinate reduction (MCR) method, which is a projection based approach. The concept of MCR is to transform the full model from physical coordinates in physical space to modal coordinates in modal space by using the eigenvector matrix of this system. Subsequently, it removes those modes that have less important contributions to the system responses. Generally, only a few modes have a significant effect on the system dynamics within the frequency range of interest.

After discretisation of the PDE (4) in space, we obtain a system of ODEs (compare (13)) in format

$$u'(t) = Au(t) \tag{28}$$

with a system matrix $A \in \mathbb{R}^{n\times n}$. For A being diagonalisable, there exists a matrix $S \in \mathbb{R}^{n\times n}$ with eigenvectors of A and a diagonal matrix $\Lambda \in \mathbb{R}^{n\times n}$ with the corresponding eigenvalues λ_i such that

$$A = S\Lambda S^{-1} \tag{29}$$

Inserting of (29) in (28) and the subsequent multiplication of S^{-1} leads to

$$S^{-1}u'(t) = \Lambda S^{-1}u(t) \tag{30}$$

The latter equation is the starting point for a strategic selection of eigenvalues and eigenvectors (modes). It is well-known, that the low frequencies (corresponding to small eigenvalues) dominate the dynamics of the underlying physical system. Suppose $m \ll n$ ordered eigenvalues $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_m$ are deemed of interest. Consequently, we obtain with $\widetilde{\Lambda} \in \mathbb{R}^{m\times m}$ of Λ and $\widetilde{S} \in \mathbb{R}^{n\times m}$ the reduced model of order m

$$w'(t) = \widetilde{\Lambda}w(t) \tag{31}$$

where $w = \widetilde{S^{-1}}u$. This low dimensional system is much faster to solve than the original one. Applying the Crank-Nicolson method to (31) based solely on diagonal matrices, such that implicit method can solved by only matrix-vector-multiplications.

V. EXPERIMENTAL RESULTS

In general, we perform a dense point-to-point correspondence, involving all vertices the shapes are made off. In detail, the experiments are evaluated as follows:

a) Hit Rate: The percentage Hit Rate is defined as $TP/(TP+FP)$, where TP and FP are the number of true positives and false positives, respectively.

b) Geodesic Error: For the evaluation of the correspondence quality, we followed the Princeton benchmark protocol [9]. This procedure evaluates the precision of the computed matchings x_i by determining how far are those away from the actual ground-truth correspondence x^*. Therefore, a normalised intrinsic distance $d_{\mathscr{M}}(x_i, x^*)/\sqrt{A_{\mathscr{M}}}$ on the transformed shape is introduced. Finally, we accept a matching to be true if the normalised intrinsic distance is smaller than the threshold 0.25.

c) Dataset: For experimental evaluation, we compare datasets at three different resolutions, namely small, middle and large. For the small ($N = 4344$) and medium ($N = 27894$) shapes, examples of the wolf and cat class are used, taken from the TOSCA data set [2]. The Fat Baby shapes have a large resolution ($N = 59727$) and are taken from the KIDS dataset [16]. The datasets are available in the public domain as shown in Figure 3. All shapes provide ground-truth and degenerated triangles were removed.

d) General Parameters: We set the stopping time to $t_M = 25$ and the variance parameter $\sigma = 1$ for the still free parameters of the geometric diffusion process in (4). For the implicit methods the time increment $\tau = 1$ was fixed for all experiments. All three parameters are chosen without a fine-tuning, since we are interested to figure out the differences of the numerical methods compared to accuracy and computational costs.

All experiments were done in MATLAB R2017b on a recent Desktop Computer with an Intel Xeon(R) CPU E5-2609 v3 CPU running at 6x 1.9GHz and XGB of 15.6 GB RAM. The sparse linear system for the direct method was prefactorised with the SuiteSparse package [4]. The computed eigenvalues and eigenvectors for MCR are computed by the Matlab internal function *eigs*.

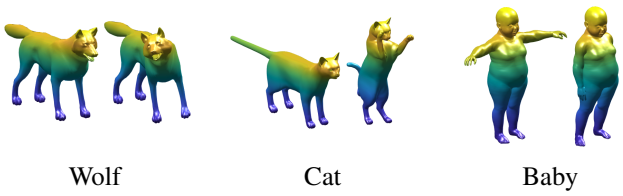

Fig. 3. For experimental evaluation, we compare shapes at three different resolutions, namely small, middle and large. These are represented by the "wolf", "cat" and "baby" , taken from the TOSCA dataset.

A. Numerical Evaluation

a) Experiment - Wolf: First of all we analyse the wolf shape with a point cloud size of only $N = 4344$ points. In case of an explicit method (20) we get the time step restriction $\tau_{\max} \approx 0.0085$s, which corresponds to around 2900 iterations. The CPU time (in seconds) of the explicit method with around 360s offers unacceptable running costs and is consequently quite inefficient. In contrast, the FSI scheme for (26) takes control over the individual time steps. Due to the fact, that the final stopping time is fix we can only specify the number of cycles C. Increasing C, whereby the cycle length n becomes smaller, improves the accuracy of FSI compared to the geodesic error of the standard explicit method, see Figure 4. Already for $C = 2$ we can achieve respectable results with an additional dramatic speed up of the explicit

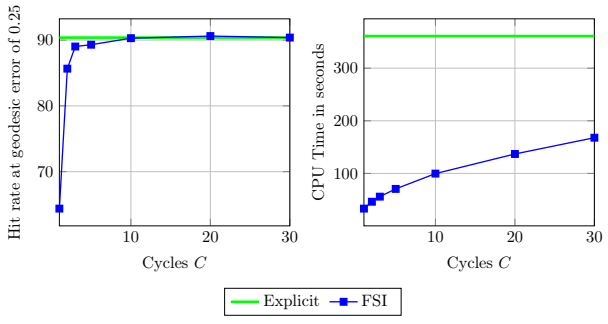

Fig. 4. Results on the dataset wolf. We compare the geodesic error up to 0.25 (left) and the performance time (right) between the explicit method and the FSI scheme for different number of cycles C.

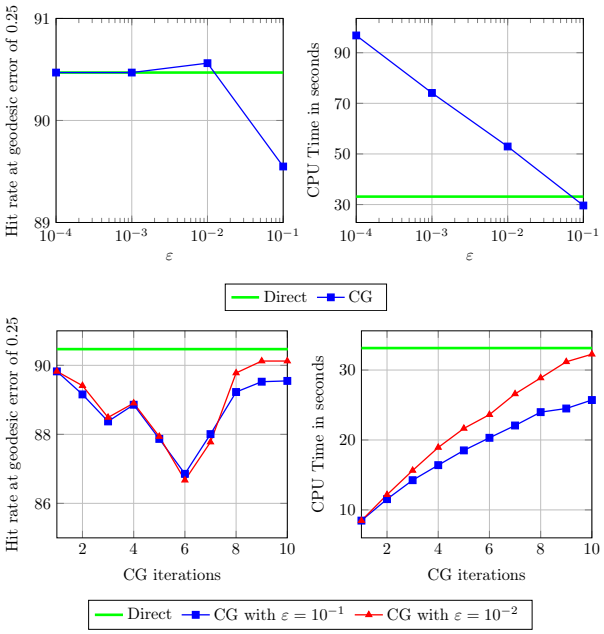

Fig. 5. Results on the dataset wolf. We compare the geodesic error up to 0.25 (left) and the performance time (right) between the direct method and the CG method for different ε (top) and different number of CG iterations l (bottom) for $\varepsilon = 10^{-1}$ and $\varepsilon = 10^{-2}$. We observe a dint-like effect (first down, then up) w.r.t. the hit rate when increasing CG iterations.

method. However, we will see that the computational costs of FSI of around 50s are still not too efficient.

The CPU time can be reduced to around 33s by using the direct method for (22) and generates the same geodesic error accuracy as the explicit method. Solving the linear system (27) with the CG method requires a stopping criterion. In general the *relative residual* $\frac{\|\mathbf{b}-A\mathbf{x}\|_2}{\|\mathbf{b}\|_2} \leq \varepsilon$ will be used. Increasing ε leads to a faster CG approximation, however the accuracy remained almost unchanged also for the relatively large value $\varepsilon = 10^{-1}$ cf. Figure 5. For this reason we tested CG for $\varepsilon = 10^{-1}, 10^{-2}$, and different number l of CG iterations, see also Figure 5. In this case, the reduction of l compared to the geodesic error accuracy leads to slight oscillations, which are still acceptable, yet with a fast CPU time of around 10s.

Finally, we explore the MCR technique. The whole MCR process includes the computation of eigenvalues, eigenvectors and the subsequent solving of the resulting reduced system. For this experiment we increase the number of used ordered modes, starting from $N_{max} = 1$ and end up to $N_{max} = 3000$. The evaluation in Figure 6 shows that the accuracy of the geodesic error depends on the number of used modes. For a larger amount of modes the accuracy increases significantly. However, it is remarkable that the results for the small spectrum $N_{max} \approx 10$ are similar to the large spectrum $N_{max} \approx 1000$. Only from a certain size ($N_{max} \approx 2000$) upwards we receive an acceptable matching result, however in unacceptable running time. Nevertheless, the CPU time by using a smaller number of modes is unbeatable. For $N_{max} = 100$ modes the approximative solution is computed

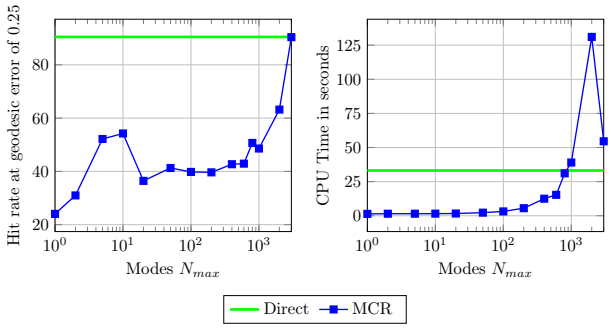

Fig. 6. Results on the dataset wolf. We compare the geodesic error up to 0.25 (left) and the performance time (right) between the direct method and the MCR technique for different number of modes $N_{\max}$.

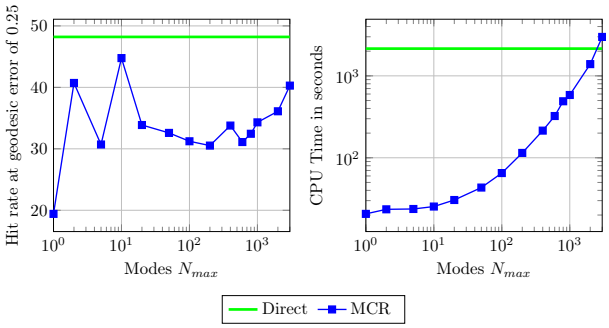

Fig. 8. Results on the dataset cat. We compare the geodesic error up to 0.25 (left) and the performance time (right) between the direct method and the MCR technique for different number of modes $N_{\max}$.

in 3s under consideration of a much less geodesic error accuracy.

b) Experiment - Cat: In the following, we consider the medium dataset cat with a cloud point size of $N = 27894$ points. Although the FIS scheme outperforms the explicit method, it is quite inefficient in consequence of the large spectrum of eigenvalues, which depends on the mesh size of the discretised shape. The mesh size is here very small, as is often the case in shape matching applications, and therefore the allowed time step $0 < \tau_{\max} \ll 1$ is also extremely small-sized. Consequently, the dramatic rise of the computational effort can not intercept the low costs of matrix-vector-multiplication.

The direct method solves the linear system in around

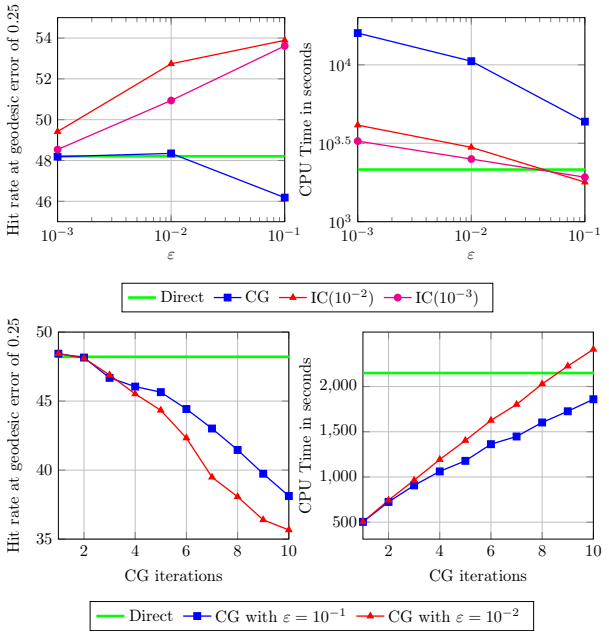

Fig. 7. Results on the dataset cat. We compare the geodesic error up to 0.25 (left) and the performance time (right) between the direct method and CG respectively IC(10^{-2}), IC(10^{-2}) for different ε (top). In addition, we present a comparison between the direct method and the CG method for the first few CG iterations l (bottom) for $\varepsilon = 10^{-1}$ and $\varepsilon = 10^{-2}$. Let us comment, that we observe here only the first part of the dint-effect, compare Figure 5, the hit rate increase of the dint will start with iteration $l = 14$.

2150s. Due to the large matrix size, we apply CG, and PCG with the incomplete Cholesky (IC) decomposition, as shown in Figure 7. For the latter decomposition often a numerical fill-in strategy IC(γ) is used, called *drop tolerance*, where the parameter $\gamma > 0$ describes a dropping criterion, cf. [18]. We have found by tests that $\gamma \in [10^{-2}, 10^{-3}]$ gives the best trade-off between accuracy and CPU time. Increasing ε leads naturally to faster computations but slightly worse results. Compared to the time performance of the direct method we achieve only a minor improvement. However, if we take a closer examination of the required iterations l, for CG and PCG with $\varepsilon = 10^{-1}$, it is conspicuous that PCG needed just about 1 iteration and CG on the other hand 20 iterations. The latter observation again inspires the idea to perform the CG method for a smaller number of iterations $l \leq 10$, which accordingly should be sufficient to gain acceptable results in fast CPU time. The results of CG for $\varepsilon = 10^{-1}, 10^{-2}$, and a number l of CG iterations is shown in Figure 7. Decreasing l reduces the time costs a lot, for instance one can save around 1500s for $l = 1$ compared to the direct method.

Application of MCR achieves the same solution behaviour as the wolf dataset, see Figure 8. Increasing the amount of used ordered modes leads to a significantly higher accuracy of the geodesic error and to more stable performance. Nevertheless, the computational costs of MCR grow exponentially (by increasing N_{max}) and a practicable value N_{max} is accordingly small. It is observable that the geodesic error for the range $N_{max} \in [20, 1000]$ is almost equal and oscillates just weakly. Even if the MCR technique for $N_{max} = 200$ looses around 35% accuracy (from 48 to 30) the simultaneous extremely short running time of around 100s is remarkable. Therefore, one may save around 95% of the computational time in relation to the direct method.

c) Experiment - Baby: Finally, we investigate the large dataset baby with a cloud point size of $N = 59727$ points. As before, we will study the shape matching correspondence in relation to the accuracy of the geodesic error and the computational effort between the direct method, the CG method (for $\varepsilon = 10^{-1}$ and $l \leq 10$) and the MCR technique.

The direct method requires around 10300s ($\approx$ 2 hours and 50 minutes) to solve 59727 linear systems on each shape for each time step. At this point it is recognizable, that large

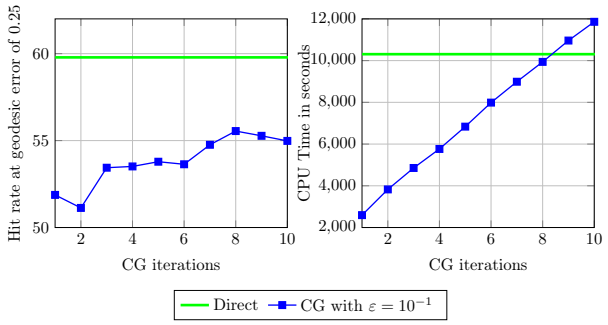

Fig. 9. Results on the dataset baby. We compare the geodesic error up to 0.25 (left) and the performance time (right) between the direct method and CG method for different number of iterations l and $\varepsilon = 10^{-1}$.

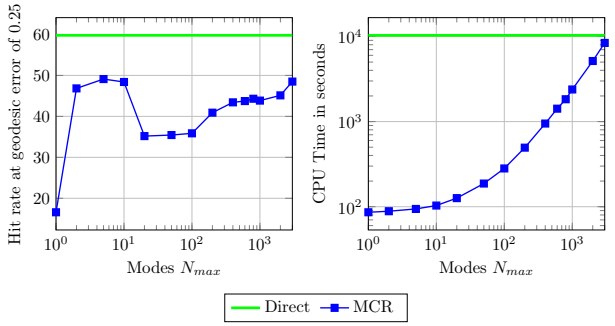

Fig. 10. Results on the dataset baby. We compare the geodesic error up to 0.25 (left) and the performance time (right) between the direct method and the MCR technique for different number of modes $N_{\max}$.

datasets produce extreme highly computational costs.

The evaluation of CG for $\varepsilon = 10^{-1}$ and $l \leq 10$ is shown in Figure 9. The percentage deviation of the accuracy of CG in relation to the direct method is approximately around 10%. In contrast, for $l = 1$, we can reduce the computational effort significantly to around 2600s, which can save more than 75% of the computational costs.

The MCR technique yields identical results as in the other two cases, compare Figure 10. The geodesic error oscillates in the range of $N_{max} = [1,20]$ and from $N_{max} = 20$ up to $N_{max} = 3000$ it converges against the solution of the direct method. Unfortunately, the convergence is very slow and a huge number of modes N_{max} is required. As before, for $N_{max} = 200$ the deviation of accuracy is around 35% (from 60 to 40), yet MCR needs around 500s. This means MCR reduce extremely the computational effort to 5%.

VI. Conclusion and Further Work

Experimental results confirm that the direct method, the CG method and the MCR technique are predestined for solving shape matching by time integration. Each of these methods has its own main advantage. The direct method is very accurate but can be inefficient. The CG method may reduce the computational costs to around 70%, whereby the percentage deviation of the accuracy in relation to the direct method is still less than 10%. The MCR technique is extremely fast and can save around 95% CPU time of the direct method, however it looses around 35% accuracy. Let us mention, that the underlying datasets are noise-free, therefore a further issue to examine is the influence of noisy data to the robustness of the numerical solvers.

The experiments show another interesting point of the MCR technique. It is remarkable that the results for a small spectrum are similar to the ones for a significantly larger spectrum. Unfortunately, the small spectrum suffers by a performance collapse at a low range of modes, roughly $N_{max} \in [1,20]$. Therefore, an interesting aspect would be to tune the small spectrum so that it becomes more stable. One may also consider other, more elaborated MOR techniques to possibly obtain a better trade-off between quality and computational effort.

References

[1] M. Bähr, M. Breuß, Y. Quéau, A. S. Boroujerdi, and J.-D. Durou, "Fast and accurate surface normal integration on non-rectangular domains," *CVM*, vol. 3, no. 2, pp. 107–129, 2017.

[2] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, *Numerical geometry of non-rigid shapes*. Springer, 2009.

[3] R. Dachsel, M. Breuß, and L. Hoeltgen, "Shape matching by time integration of partial differential equations," in *Proc. of SSVM*, 2017, pp. 669–680.

[4] T. A. Davis, "Algorithm 930: Factorize: An object-oriented linear system solver for matlab," *ACM Trans. Math. Softw.*, vol. 39, no. 4, pp. 28:1–28:18, July 2013.

[5] M. P. do Carmo, *Differential geometry of curves and surfaces*, 2nd ed. Dover Publications, 2016.

[6] S. Grewenig, J. Weickert, and A. Bruhn, "From box filtering to fast explicit diffusion," in *Proc. of DAGM*, 2010, pp. 533–542.

[7] D. Hafner, P. Ochs, J. Weickert, M. Reißel, and S. Grewenig, "Fsi schemes: Fast semi-iterative solvers for pdes and optimisation methods," in *Proc. of GCPR*, 2016, pp. 91–102.

[8] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *NIST*, vol. 6, no. 49, pp. 46–70, 1952.

[9] V. G. Kim, Y. Lipman, and T. A. Funkhouser, "Blended intrinsic maps," in *ACM (TOG)*, vol. 30, no. 4, 2011, pp. 1–12.

[10] G. Meurant, *Computer Solution of Large Linear Systems*. Elsevier Science, First Edition, 1999.

[11] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," in *In Visualization and Mathematics III*. Springer, 2002, pp. 35–57.

[12] S.-B. Nouri, "Advanced model-order reduction techniques for large-scale dynamical systems," Ph.D. dissertation, Department of Electronics, Carleton University, Canada, 2014.

[13] U. Pinkall and K. Polthier, "Computing discrete minimal surfaces and their conjugates," *Experimental Mathematics*, vol. 2, no. 1, pp. 15–36, 1993.

[14] Z.-Q. Qu, *Model Order Reduction Techniques with Applications in Finite Element Analysis*. Springer, 2004.

[15] M. Reuter, F. E. Wolter, and N. Peinecke, "Laplace-Beltrami spectra as shape-DNA of surfaces and solids," *Computer-Aided Design*, vol. 38, no. 4, pp. 342–366, 2006.

[16] E. Rodolà, S. Rota Bulo, T. Windheuser, M. Vestner, and D. Cremers, "Dense non-rigid shape correspondence using random forests," in *Proc. of CVPR*, 2014, pp. 4177–4184.

[17] R. Rustamov, "Laplace-Beltrami eigenfunctions for deformation invariant shape representation," in *Symp. Geometry Processing*, 2007, pp. 225–233.

[18] Y. Saad, *Iterative Methods For Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 2nd Edition, 2003.

[19] G. Smith, *Numerical Solution of Partial Differential Equations: Finite Difference Methods*. Clarendon Press, Oxford Applied Mathematics and Computing Science Series, 1985.

[20] J. Sun, M. Ovsjanikov, and L. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," *Computer Graphics Forum*, vol. 28, no. 5, pp. 1383–1392, 2009.

[21] O. Van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or, "A survey on shape correspondence," in *Euro STAR*, 2010, pp. 61–82.