

INAUGURALDISSERTATION

ZUR
ERLANGUNG DER DOKTORWÜRDE
DER
NATURWISSENSCHAFTLICH-MATHEMATISCHEN
GESAMTFAKULTÄT
DER
RUPRECHT - KARLS - UNIVERSITÄT
HEIDELBERG

MULTI-MODAL PARTIAL
SURFACE MATCHING
FOR INTRA-OPERATIVE REGISTRATION

VORGELEGT VON
THIAGO RAMOS DOS SANTOS, MSC.COMPSC.
AUS FLORIANÓPOLIS - SANTA CATARINA - BRASILIEN

2012

THEMA

**MULTI-MODAL PARTIAL
SURFACE MATCHING
FOR INTRA-OPERATIVE REGISTRATION**

AUTHOR: THIAGO RAMOS DOS SANTOS, MSC.COMPSC.

BETREUER: CHRISTOPH SCHNÖRR, PROF. DR.
HANS-PETER MEINZER, PROF. DR.

ZUSAMMENFASSUNG

Die Übertragung präoperativer Daten auf die intraoperative Situation am Patienten ist ein wichtiger Bestandteil jeder computergestützten Intervention. Diese *intraoperative Registrierung* wird gewöhnlich marker- oder bildbasiert durchgeführt. Andere Ansätze nutzen die intraoperative Erfassung der Organoberfläche mittels 3D-Tiefenmessung, welche eine Registrierung zur präoperativ akquirierten Oberfläche ermöglicht. Dieser Ansatz ist jedoch nicht trivial, da hier Rauschen, Verzerrungen und Verformungen auftreten sowie häufig nur teilweise überlappende, beinahe flache Oberflächen vorliegen. Wegen dieser Schwierigkeiten wurde intraoperative Oberflächenregistrierung bis jetzt nur lokal angewandt, während die globale Registrierung noch manuell durchgeführt wird.

In dieser Arbeit werden zwei verschiedene Ansätze zur automatischen, markerlosen Oberflächenregistrierung für intraoperative Szenarien vorgestellt, wobei insbesondere o.g. Probleme berücksichtigt werden. Für den ersten Ansatz werden Oberflächen als Graph repräsentiert und Korrespondenzen zwischen ihnen mittels Graph-Matching identifiziert. Da das Graph-Matching Problem bekanntlich NP-hart ist, wurde ein iteratives Verfahren basierend auf der Bewertung der Knotenähnlichkeiten gewählt und dadurch in ein lineares Zuweisungsproblem überführt. In dem zweiten Ansatz werden Korrespondenzen durch die Auswahl von zwei räumlichen Konfigurationen von Landmarken bestimmt, die in Bezug auf ihre Fehlermetrik besser zugeordnet werden können. Diese Fehlermetrik berücksichtigt nicht nur den Zuordnungsfehler, sondern auch ein neues Maß für die Verlässlichkeit von räumlichen Konfigurationen. Die Registrierung wird dann mit Hilfe eines Greedy Optimisierungsalgorithmus gelöst.

Beide Ansätze wurden in etlichen Experimenten, welche die intraoperativen Bedingungen simulierten, evaluiert. Es konnte gezeigt werden, dass der Graph-Matching-Ansatz insbesondere für die Registrierung von kleinen Teiloberflächen geeignet ist, während der punkt-basierte Ansatz bei hohem Rauschen robust und akkurat war. Neben dem signifikanten Beitrag zur partiellen Oberflächenregistrierung ist diese Arbeit von besondere Bedeutung zur Erreichung von vollautomatischer, markerloser und intraoperativer Registrierung für computerassistierte Assistenzsysteme.

ABSTRACT

An important task for computer-assisted surgical interventions is the alignment of pre- and intra-operative spaces allowing the transfer of pre-operative information to the current patient situation, known as *intra-operative registration*. Registration is usually performed by using markers or image-based techniques. Another approach is the intra-operative acquisition of organ surfaces by 3D range scanners, which are then matched to pre-operatively generated surfaces. However, this approach is not trivial, as methods for intra-operative surface matching must be able to deal with noise, distortions, deformations, and the availability of only partially overlapping, nearly flat surfaces. For these reasons, surface matching for intra-operative registration has so far only been used to account for displacements that occur in local scales, while the actual alignment is still performed manually.

The main contributions of this thesis are two different approaches for automatic surface matching in intra-operative environments. The focus here is the registration of surfaces acquired by different modalities, dealing with the aforementioned issues and without relying on unique landmarks. For the first approach, surfaces are converted to graph representations and correspondences between them are identified by means of graph matching. Graphs are obtained automatically by segmenting the surfaces into regions with similar properties. As the graph matching problem is known to be NP-hard, it was solved by iteratively computing node similarity scores, and converting it to a *linear assignment problem*. In the second approach, correspondences are identified by the selection of two spatial configurations of landmarks that can be better fitted to each other, according to an error metric. This error metric does not only incorporate a fitting error, but also a new measure for spatial configuration reliability. The optimization problem is solved by means of a greedy algorithm.

Evaluation of the two approaches was performed with several experiments, simulating intra-operative conditions. While the graph matching approach proved to be robust for the registration of small partial data, the point-based approach proved to be more reliable for noisy surfaces. Apart from being a significant contribution to the field of feature-less partial surface matching, this work represents a great effort towards the achievement of a fully automatic, marker-less, registration system for computer-assisted surgery guidance.

PUBLICATIONS

Some ideas, figures and tables from this work have appeared previously in the following publications:

INTERNATIONAL JOURNALS

- Maier-Hein, L., Franz, A.M., dos Santos, T.R., Schmidt, M., Fangerau, M., Meinzer, H.-P., Fitzpatrick, M. *Convergent Iterative Closest-Point Algorithm to Accomodate Anisotropic and Inhomogeneous Localization Error*. In: **IEEE Trans. Patt. Anal. and Mach. Intel. (TPAMI)**. (To appear). 2012
- dos Santos, T.R., Meinzer, H.-P., Maier-Hein, L. *Extending the Doubly Linked Face List for the Representation of 2-Pseudomanifolds and 2-Manifolds with Boundaries*. In: **Int. J. of Comp. Geometry and Apps. (IJCGA)**. 21(4):467-494. 2011
- Seitel, A., Kwon, Y., Mersmann, S., Kilgus, T., Groch, A., dos Santos, T.R., Franz, A., Nolden, M., Meinzer, H.-P., Maier-Hein, L. *MITK-ToF - Range data within MITK*. In: **Int. J. of Comp. Assisted Radiology and Surg. (JCARS)**. 2011

PEER-REVIEWED INTERNATIONAL CONFERENCES

- dos Santos, T.R., Goch, C., Franz, A., Meinzer, H.-P., Heimann, T., Maier-Hein, L. *Minimally deformed correspondences between surfaces for intra-operative registration*. In: **SPIE Med. Img.: Img. Proc.** 2012
- Maier-Hein, L., Seitel, A., Kilgus, T., Franz, A., Fangerau, M., Bartha, L., dos Santos, T.R., Mersmann, S., Groch, A., Yung, K., Meinzer, H.-P. *Registration of Partially Overlapping Surfaces for Time-of-Flight based Augmented Reality on Mobile Devices*. In: **SPIE Med. Img: Img.-Guided Proc., Robotic Interv., and Mod.** 2012
- dos Santos, T.R., Franz, A., Meinzer, H.-P., Maier-Hein, L. *Robust Multi-modal Surface Matching for Intra-operative Registration*. In: **IEEE Comp.-Based Med. Sys. (CBMS)**. 2011
- dos Santos, T.R., Seitel, A., Kilgus, T., Meinzer, H.-P., Maier-Hein, L. *Multi-modal surface comparison and its application to intra-operatively acquired range data*. In: **SPIE Med. Img.: Img. Proc.** 2011

- Seitel, A., dos Santos, T.R., Mersmann, S., Penne, J., Groch, A., Yung, K., Tetzlaff, R., Meinzer, H.-P., Maier-Hein, L. *Adaptive bilateral filter for image denoising and its application to in-vitro Time-of-Flight data*. In: **SPIE Med. Img.: Vis., Img.-Guided Proc., and Mod.** 2011
- Maier-Hein, L., dos Santos, T.R., Franz, A., Meinzer, H.-P., Fitzpatrick, J.M. *Iterative Closest Point Algorithm with Anisotropic Weighting and its Application to Fine Surface Registration*. In: **SPIE Med. Img.: Img. Proc.** 2011
- Groch, A., Seitel, A., Hempel, S., Speidel, S., Engelbrecht, R., Penne, J., Höller, K., Rohl, S., Yung, K., Bodenstedt, S., Pflaum, F., dos Santos, T.R., Mersmann, S., Meinzer, H.-P., Hornegger, J., Maier-Hein, L. *3D surface reconstruction for laparoscopic computer-assisted interventions: comparison of state-of-the-art methods*. In: **SPIE Med. Img. 2011: Vis., Img.-Guided Proc., and Mod.** 2011
- dos Santos, T.R., Seitel, A., Meinzer, H.-P., Maier-Hein, L. *Correspondences Search for Surface-Based Intra-Operative Registration*. In: **Med. Img. Comp. and Comp. Assisted Interv. (MICCAI)**. 2010
- dos Santos, T.R., Gergel, I., Meinzer, H.-P., Maier-Hein, L. *Fast Correspondences Search in Anatomical Trees*. In: **SPIE Med. Img.: Img. Proc.** 2010
- Maier-Hein, L., Schmidt, M., Franz, A., dos Santos, T.R., Seitel, A., Jähne, B., Fitzpatrick, J.M. *Accounting for anisotropic noise in fine registration of Time-of-Flight range data with high-resolution surface data*. In: **Med. Img. Comp. and Comp. Assisted Interv. (MICCAI)**. 2010
- Gergel, I., dos Santos, T.R., Tetzlaff, R., Maier-Hein, L., Meinzer, H.-P., Wegner, I. *Particle Filtering for Respiratory Motion Compensation during Navigated Bronchoscopy*. In: **SPIE Med. Img.: Vis., Img.-Guided Proc., and Mod.** 2010

PEER-REVIEWED NATIONAL CONFERENCES

- dos Santos, T.R., Seitel, A., Meinzer, H.-P., Maier-Hein, L. *Time-of-Flight Surface Denoising by Spectral Decomposition*. In: **Bildverarbeitung für die Medizin (BVM)**. 2011
- Kilgus, T., dos Santos, T.R., Seitel, A., Yung, K., Franz, A., Groch, A., Meinzer, H.-P., Maier-Hein, L. *Generation of triangle*

meshes from Time-of-Flight data for surface registration. In: **Bildverarbeitung für die Medizin (BVM).** 2011

- Maier-Hein, L., Franz, A., Fangerau, M., Schmidt, M., Seitel, A., Mersmann, S., Kilgus, T., Groch, A., Yung, K., dos Santos, T.R., Meinzer, H.-P. *Mobile augmented reality for on-patient visualization of medical images.* In: **Bildverarbeitung für die Medizin (BVM).** 2011
- dos Santos, T.R., Gergel, I., Meinzer, H.-P., Maier-Hein, L. *Graphbasierte Registrierung von Tubulären Strukturen* In: **Bildverarbeitung für die Medizin (BVM).** 2010
- Maier-Hein, L., dos Santos, T.R., Franz, A., Meinzer, H.-P. *Iterative Closest Point Algorithm in the Presence of Anisotropic Noise* In: **Bildverarbeitung für die Medizin (BVM).** 2010
- Gergel, I., dos Santos, T.R., Tetzlaff, R., Maier-Hein, L., Meinzer, H.-P. *Partikelfilterung für die Kompensation von Atembewegung während der Navigierten Bronchoskopie.* In: **Bildverarbeitung für die Medizin (BVM).** 2010
- Seitel, A., dos Santos, T.R., Mersmann, S., Penne, J., Tetzlaff, R., Delorme, S., Meinzer, H.-P., Maier-Hein, L. *Time-of-flight cameras for intra-operative surface acquisition.* In: **Bildverarbeitung für die Medizin (BVM).** 2010

ACKNOWLEDGMENTS

Life is too short to learn German.

— Richard Porson

During my PhD, I was financed by a CAPES/DAAD (Brazil/Germany) scholarship.

I would like to thank Prof. Hans-Peter Meinzer, head of the Div. of Medical and Biological Informatics at the German Cancer Research Center (DKFZ), for his support to this work, and for have given me the unique opportunity to be a member of his team. I learned a lot of lessons, which I will never forget.

Also many thanks to Prof. Christoph Schnörr, head of the Image & Pattern Analysis Group at the University of Heidelberg, for accepting the supervision of this work. Prof. Schnörr has the unique ability of making very wise observations, requiring very little explanation from my part.

Thanks to Lena Maier-Hein. Although we had different opinions on several issues, she played a very important part in this work.

Thanks a lot to Tobias Heimann, for his help on the final phase of this work, when things start getting complicated, and one thinks it will be impossible to finish.

Many many thanks to my colleagues and friends, who helped making my stay in Heidelberg much nicer than I could have imagined: Anja Groch (who also translated the abstract into German for the price of 4 cookies and a pack of sesame sticks), Diana Wald, Michael Müller, Sven Mersmann, Marco Nolden, Markus Fangerau, Johannes Kast, Markus Engel. More special thanks to Ingmar Gergel, Alexander Seitel and Matthias Baumhauer, who are not only friends, but became somehow part of my “family”.

There is a person, to whom being thankful won't be enough, my wife, Iara dos Santos. There are no words to describe her importance, her support, her patience. I hope I can make it up to you someday.

I dedicate this work to my family and friends back home. Being abroad is not easy, and in the same way I feel I am “winning” a lot being here, I feel I have been “missing” a lot not being there. It is always wonderful to know that you all have been supporting me, even when supporting just means giving a call and saying that everything is going to turn out just fine. Thanks a lot to you all, or, as they would say in German, *Dankeschön!*

CONTENTS

List of Figures	xvii
List of Tables	xxvii
List of Algorithms	xxviii
1 INTRODUCTION	1
1.1 Objectives	4
1.2 Structure of this work	4
2 OVERVIEW OF SURFACE-BASED INTRA-OPERATIVE REG- ISTRATION	7
2.1 Image segmentation	10
2.2 3D range scanners	10
2.3 Range images to surfaces	10
2.3.1 Object recognition	11
2.3.2 Surface generation	12
2.3.3 Data structures for mesh representation	12
2.4 Deformation models	13
3 STATE OF THE ART OF SURFACE MATCHING APPROACHES	15
3.1 Descriptor representation	16
3.2 Feature selection	22
3.3 Error metric	25
3.3.1 Error metrics for point correspondences	26
3.3.2 Other error metrics	29
3.4 Optimization	30
3.4.1 Rough-scale optimization	31
3.4.2 Fine-scale optimization	35
3.5 Transformation	36
3.6 Summary	37
4 SURFACE REPRESENTATION	41
4.1 The <i>Doubly Linked Face List</i>	44
4.2 Extensions to the <i>Doubly Linked Face List</i>	48
4.2.1 2-Manifolds with boundaries	49
4.2.2 2-Pseudomanifolds	52
4.2.3 The <i>Extended Doubly Linked Face List</i>	55
4.2.4 Implementation aspects and memory usage	56

4.3	Manipulation by direct face creation and removal . . .	57
4.3.1	The CREATEFACE operator	57
4.3.2	The REMOVEFACE operator	66
4.4	XDLFL initialization by posterior edge list creation . .	68
4.5	Discussion	69
5	DISTORTIONS BETWEEN MULTI-MODAL SURFACES	71
5.1	Mesh comparison framework	71
5.1.1	Establishment of correspondences	72
5.1.2	Descriptors and local distance metrics	74
5.1.3	Global distance metric	74
5.2	Assessment of the distortions between time-of-flight and computed tomographies	75
5.3	Discussion	75
6	SURFACE MATCHING	81
6.1	Region-based approach	82
6.1.1	Mesh segmentation	84
6.1.2	Graph construction	88
6.1.3	Correspondence search	88
6.1.4	Evaluation and results	98
6.1.5	Discussion	112
6.2	Point-based approach	116
6.2.1	Error metrics and feature selection in surface matching	117
6.2.2	Descriptor	121
6.2.3	Selection of features and candidate correspon- dences	124
6.2.4	A new error metric for the registration of feature-less surfaces	126
6.2.5	Correspondences search	130
6.2.6	Evaluation and results	132
6.2.7	Discussion	137
7	CONCLUSIONS	147
7.1	Summary of contributions	149
7.2	Future work	151
A	ROTATION INVARIANCE OF THE LOCAL COORDINATE SYSTEM	155
B	DERIVATION OF THE <i>average target registration error</i>	157
	BIBLIOGRAPHY	160

LIST OF FIGURES

Figure 1	Processing pipeline for surface-based intra-operative registration.	8
Figure 2	Acquisition of the surface of an object by a time-of-flight camera. (a) The object (a mug); (b) The acquired range image, with distance values mapped to a color palette ranging from green (further) to red (near); (c) Extraction of the object of interest; (d) Conversion of the range image into a 3D point cloud; (e) Triangulation of the point cloud in order to build a surface.	11
Figure 3	Projection of an object in 3D space to the 2D space of an image according to the <i>pinhole camera model</i>	12
Figure 4	Geodesic distance fields for two distinct points on different surfaces (red: high, blue: low): (a) A plane. Both points have exactly the same radial geodesic field. (b) A porcine liver surface acquired by a time-of-flight camera. As the surfaces are nearly planar, the geodesic fields of both points are similar. They are also very similar to the radial fields on the plane surface. (c) A humanoid shape. As the structure is more complex than the previous surfaces, not resembling a plane, distinct points on the surface have different geodesic fields (unless they lie close to each other).	18
Figure 5	Mean curvature plot on a porcine liver surface acquired by a time-of-flight camera. (a) Curvature computed according to the discrete approach of Meyer et al. [189] on the original surface (noisy); (b) Curvature computed according to the same approach on the smoothed surface; (c) Curvature computed according to the noise-robust approach of Cazals and Pouget [54] on the original surface. The curvature values on the smoothed surface (b) and on the original surface computed with a more robust approach (c) are highly similar.	19

Figure 6	Spin images for three points on the surface of a duck model (image found in [142]).	20
Figure 7	Matching a dog to different four-legged animals with different surface details by means of comparison of their surface skeletons (image found in [20]).	21
Figure 8	Representation of the cortical surface by spherical harmonic series of different highest frequency bands (image found in [61]). From left to right, the highest frequency band in the series decreases.	21
Figure 9	Manifold harmonic bands of two Armadillo models, and one of the Armadillo’s arm. The columns show the first, second and third bands of their manifold harmonics. As the models in rows (a) and (b) can be nearly mapped to each other by an isometry, the bands are similar, as the spaces spanned by these surfaces are similar. However, the partial surface in row (c) spans a different space, which is not isometric to the other models, thus inducing different bands, defined in its own space.	23
Figure 10	Heat diffusion field of a heat source (blue point) on different surfaces (red: high, blue: low), at different time points (125 and 500 seconds): (a) A plane. (b) A porcine liver surface acquired by a time-of-flight camera. (c) A humanoid shape. The heat diffusion is inversely proportional to the geodesic distance (Fig. 4).	24
Figure 11	Features selected from two different dog surfaces (image found in [285]). Note that features are consistently selected on both surfaces, <i>i.e.</i> , they are chosen on the same locations.	24
Figure 12	Comparison between selected feature points on two surfaces of the same object (porcine liver), as acquired by a computed tomography (left) and a time-of-flight camera (right). Features were selected according to the multi-scale approach of Ho and Gibbins [131].	25

Figure 13	Registration of several partial surfaces of a bunny using the $E_{\text{dRMS}}(\cdot)$ (see Eq. 3.6) error metric (bottom left) and after the error minimization of the initial registration with ICP (bottom right).	28
Figure 14	Correspondences obtained for a pair of hand model using the $E_{\text{iso}}(\cdot)$ (see Eq. 3.7) error metric for registration under isometric deformations (image found in [89]).	29
Figure 15	A set of initial sparse correspondences (a) and its expansion in a dense correspondence set (b) (image found in [281]).	36
Figure 16	The internal organization of the DLFL during the representation of a tetrahedron. The DLFL is composed of a doubly linked face list, a vertex list and an edge list. As an example, the links of vertex v_1 and edge e_6 are shown. This figure is based on the illustration found in the work of AKLEMAN AND CHEN [3].	44
Figure 17	The INSERTEDGE and DELETEEDGE operations. An edge is inserted between the corners represented by vertex v_1 , in face f_1 at the extremity of edge e_2 , and by vertex v_2 , in face f_1 at the extremity of edge e_4 , causing face f_1 to be divided in two. The removal of the new edge (e) causes both faces to be merged into a single face again.	46
Figure 18	Operation sequence for the creation of the tetrahedron shown in Fig. 16. Here we only show the vertices in the faces instead of the corners in order to keep the figure simple.	47
Figure 19	Step 6 of Fig. 18 using a different corner for the edge insertion.	48
Figure 20	Extending the boundaries (shown in gray) with a new face using the extension for boundary representation. In the end, the boundary label has to be manually removed from the new face, because the INSERTEDGE operator does not make any assumptions as to whether a face is a boundary or not. For the sake of simplicity, only faces f_5 and f_6 are shown explicitly. For the same reasons, instead of explicitly writing the corners as arguments for the operations, only the vertices are indicated.	51

Figure 21	Subdivision scheme where the mesh converges to b-spline surfaces, except at the boundaries, where they converge to b-spline curves.	51
Figure 22	Examples of surfaces which can be represented through 2-pseudomanifolds: (a) Two cubes connected through a single edge; (b) A pinched torus; (c) The Möbius band.	52
Figure 23	A two-faced strip with a boundary (in gray). . .	54
Figure 24	A Möbius strip representation (faces in gray represent boundaries): (a) With a regular DLFL; (b) With a DLFL extended for the representation of 2-pseudomanifolds.	54
Figure 25	The boundaries after inserting and removing new faces using the CREATEFACE and REMOVEFACE operators: (a) The boundaries are merged after the insertion of a face in the middle and split after removal; (b) The boundaries are split in two after inserting a new face and merged after removal.	57
Figure 26	A face as a connection of edge sides. Face f_1 is composed of edge sides $v_1 \rightarrow v_2$, $v_2 \rightarrow v_3$, $v_3 \rightarrow v_4$ and $v_4 \rightarrow v_1$. The beginning of each one is represented by the corners (v_1, f_1, e_1) , (v_2, f_1, e_2) , (v_3, f_1, e_3) and (v_4, f_1, e_4) , respectively. Each edge is represented by the beginning of each of its edge sides. For example, $e_1 = ((v_1, f_1, e_1), (v_2, f_3, e_1))$ and $e_4 = ((v_1, f_2, e_4), (v_4, f_1, e_4))$	58
Figure 27	Creation of a face with a boundary walk equal to $(v_1, v_2, v_3, v_4, v_5, v_6)$; gray areas indicate internal regions of the mesh. (a) Initial situation, where all vertices have boundary corners, except for v_3 and v_5 , which do not have any rotation; (b) Edge side $v_1 \rightarrow v_2$ already exists in a boundary; (c) Creation of new rotation for v_3 and edge insertion; (d) Creating edge side $v_3 \rightarrow v_4$ using the previously existing boundary corner; (e) Creation of a new rotation for v_5 and edge insertion; (f) Creating edge side $v_5 \rightarrow v_6$ using the previously existing boundary corner; (g) Edge side $v_6 \rightarrow v_1$ already exists in a boundary; (h) Removal of boundary flag.	60

Figure 28	A 2-manifold interpretation of a non-manifold, where the boundary face binds the individual face loops together.	65
Figure 29	The results of the REMOVEFACE operator: (a) The original mesh; (b) After removing f_1 (it becomes a boundary face); (c) After removing f_2 (it is completely removed).	67
Figure 30	Correspondence search between a point on the test mesh and the reference mesh: (a) Due to the anisotropy in the test mesh, the correct correspondence can not be found with the regular closest point operation; (b) The correct correspondence on the underlying surface can not be found if only the mesh is considered, possibly yielding wrong interpolation of descriptors, such as curvatures.	72
Figure 31	Two consecutive refinements of the original surface (a) by the Loop subdivision scheme, which guarantees C^2 continuity for regular meshes. Colors represent the mean curvature for each vertex. Note that, after surface refinement, curvature values remain approximately the same.	74
Figure 32	Local differences of three different descriptors between surfaces representing a porcine liver acquired with CT and ToF. The differences are shown in the bottom row, while the upper rows show the descriptor values on the CT and ToF surfaces.	76
Figure 33	Local differences of three different descriptors between surfaces representing a porcine lung acquired with CT and ToF. The differences are shown in the bottom row, while the upper rows show the descriptor values on the CT and ToF surfaces.	77
Figure 34	Comparison between the descriptors' distortions between ToF and CT surfaces found by our framework (underlying surface) and by the direct computation of differences between the meshes using the Euclidean closest point (mesh).	78
Figure 35	Descriptors distortions between ToF and CT surfaces shown in relative percentual variation, for the original (noisy) and the smoothed ToF surfaces.	78

Figure 36	The region-based approach for surface matching. In order to match two surfaces, they are segmented and graph representations are created. These graphs are then matched in order to establish correspondences between them. . .	82
Figure 37	Examples of a part-type (left; taken from [168]) and surface-type mesh segmentation (right). The surface-type segmentation was computed by creating same-sized pentagons above the parametrized surface.	84
Figure 38	Segmentation results of the total-variation based method [78], on the mean curvature. (a) The input mesh; (b) The mean curvature; (c) Results of the segmentation using three different labels; (d) The respective adjacency graph representing the segmentation. Figs. (a-c) were taken from [78].	85
Figure 39	The shape-index and the respective shapes (based on the illustration found in [155]) . . .	86
Figure 40	The segmentation of a mesh and a sub-mesh, which was extracted from the marked region. The regions are colored according to a sequential graph coloring method. Note that the sub-mesh segmentation is equal to the segmentation of its corresponding part in the reference mesh (except for the boundaries of the submesh). . .	87
Figure 41	Graphical representation of the correspondences configuration between two graphs. Every node represents a correspondences configuration C_i , while the arrows represent configurations that are related by a single node correspondence operation. The nodes are sorted in increasing order of assignment energy ($E(C)$). As our optimization algorithm only moves towards maximal positive energy variation, convergence is guaranteed. Starting at configuration C_0 , configuration C_6 will be the final one. However, the global maximum (C_8), may only be reached with moves towards negative energy variation.	97
Figure 42	Matching of anatomical trees.	100

Figure 43	Tree properties: (a) diameter; (b) internal distance to the root; (c) node level in the tree; (d) maximum and minimum angle between the edges that connect the children; (e) angle between the (virtual) line connecting the root node with the node under consideration and the first branch.	101
Figure 44	The six reference surfaces used for the evaluation of the proposed method. All meshes (with exception of the liver), are available at the AIM@SHAPE Shape Repository (http://shapes.aim-at-shape.net) and were decimated to approximately 10000 faces.	103
Figure 45	Number of nodes in the graphs of the reference meshes as a function of the submesh size.	104
Figure 46	Results of the surface matching experiment described in study 1. For each reference mesh, the mean percentage of correct matches and incorrect matches after the initial alignment averaged over 500 samples is shown as a function of the submesh size. <i>Correct ICP</i> represents the quality of the final match (after ICP).	105
Figure 47	Results of the surface matching experiment described in study 2. The mesh segmentation was performed using curvature classes as opposed to shape index and curvedness. For each reference mesh, the mean percentage of correct matches and incorrect matches after the initial alignment averaged over 500 samples is shown as a function of the submesh size. <i>Correct ICP</i> represents the quality of the final match (after ICP).	107
Figure 48	Results of the experiment described in study 3, where the proposed correspondence search method was evaluated with noisy data. The mean percentage (averaged over 50 samples) of submesh vertices that were assigned to their corresponding vertices in the reference mesh after the final iteration of the ICP is shown for all liver meshes as a function the submesh size.	108

Figure 49	Results for the evaluation of the post-processing algorithm using greedy optimization with simulated data. In this evaluation, 100 attributed small-world graphs with 200 nodes were randomly generated. From these graphs, 50 submeshes were extracted. These submeshes were included into other randomly generated graph, so that both graphs have approximately the same size. Additionally, structural and attribute errors were applied. Both graphs were matched and the results before and after post-processing were compared with the ground truth. <i>Intersection size</i> denote the size of the intersection between the graphs, <i>i.e.</i> , the size of the extracted subgraph. The results were averaged for each intersection size. <i>Correct assignments</i> are given by the percentage of nodes in the second graph that were correctly assigned to its corresponding one in the original graph, or were correctly left unassigned if there were no corresponding node.	110
Figure 50	Results of the evaluation of post-processing using greedy optimization with real scanned data. Shown in the images are the correspondences found with <i>SoftAssign</i> (top row) and after post-processing (bottom row). Also the assignment energy ($E(C)$) before and after post-processing is shown. <i>Ops</i> denotes the amount of assignment operations (insertion or removal) that were performed during post-processing. . .	111
Figure 51	Processing pipeline for our surface-based intra-operative registration method.	117
Figure 52	Comparison of the registration of a set of source and target landmarks, selected close to each other (configuration 1) and from more unique locations (configuration 2), thus forming a more widespread and non-planar spatial landmark configuration. In cases (a) and (b), both right and wrong registrations deliver small registration errors, while in cases (c) and (d), only the correct one delivers a small error.	120
Figure 53	Spherical support volume structure used for the descriptor presented in [257].	121

Figure 54	The computation of <i>target registration error</i> (TRE) [102] for surface matching may be misleading, as the TRE assumes that the correct correspondences are previously known. In (a), the correct registration returns a large TRE, as the distance to the target point (red dot) to the source landmarks is large. Minimizing the target registration error misguides the registration towards the target point, as shown in (b), where the distances between target point and source landmarks are smaller.	129
Figure 55	Simulated ToF data (blue) and its corresponding CT surface (green) used in the evaluation. Top-left: Human head; Top-right: Human liver; Bottom: Human knee.	133
Figure 56	Comparison between the descriptor presented here (blue) and the one presented by Tombari et al. [257] (green). Shown are the mean distances between corresponding ground-truth markers for different descriptor similarity thresholds.	134
Figure 57	Evaluation setting using a physical phantom of the human liver. In (a), the phantom is positioned in a respiratory motion simulator [178]. In (b), a rectangular region of interest is selected on the ToF scan. The respiratory simulation is shown in (c) (expiration) and (d) (inspiration). The green lines next to the top right corners indicate the expiration position of the device, while the red one indicates the inspiration position. The distance between the green and red lines is 2.4 cm. Note that the original motion simulator was modified with polystyrene in order to fill empty spaces and increase the robustness of the setting.	135
Figure 58	Error and computation times for the registration experiment with phantom and porcine liver surfaces in the inspiration state (deformed), using a clustering radius of 20 mm. Note that the computation time axes have a logarithmic scale. . .	139

Figure 59	Correspondences found between the surfaces acquired by a time-of-flight (ToF) camera and a computed tomography (CT) of a liver phantom (a) and a porcine liver (b). The ToF surfaces are in the inspiration state (deformed).	139
Figure 60	Correspondences between simulated ToF surface and its originating one. Mean error after registration based on ground truth information: 4.29 mm.	140
Figure 61	Correspondences between simulated ToF surface, with low resolution and therefore higher noise, and its originating one. Mean error after registration based on ground truth information: 9.49 mm.	140
Figure 62	Correspondences between the surfaces of the faces of three different persons.	141

LIST OF TABLES

Table 1	Overview of the state-of-the-art surface matching approaches. Refer to Secs. 3.1, 3.3, 3.4 and 3.5 for a description of the methods and of the abbreviations.	39
Table 2	Results for the anatomical tree matching evaluation. <i>Nodes</i> show the number of nodes in the two corresponding trees starting with bigger ones. <i>Assignable</i> refers to the number of nodes in the smaller tree that have a corresponding node in the bigger tree (according to the ground truth assignment). <i>Unassignable</i> refers to the number of nodes in the smaller trees that do not have a corresponding one. In this case, <i>correct</i> shows the number of nodes that were correctly left unassigned, while <i>wrong</i> shows the total of nodes that were assigned to any other one. The times shown are the durations of topological and properties similarity scoring, assignment and reduction of false matches. The averages are given in percentages of the total in their categories (<i>assignable</i> or <i>unassignable</i>), except for the totals in the categories themselves, which are given in percentages of the smaller tree. . .	102
Table 3	Results of the experiments. The table shows the errors between ground-truth markers for each experiment, averaged for the entire descriptor tolerance interval. All values are given in mm. We show the errors for both the computation of the registration with and without the incorporation of the configuration error (Sec. 6.2.4.3) in the error metric. Exp. indicates that the surfaces were in the expiration state (not deformed), while Insp. denotes the inspiration state. . . .	138

LIST OF ALGORITHMS

4.1	The <code>ISBOUNDARY</code> operator.	50
4.2	The <code>CREATEROTATION</code> operator.	53
4.3	The <code>DELETEROTATION</code> operator.	53
4.4	Procedure for the construction of an operations sequence graph and identification of the shortest path.	64
4.5	The <code>CREATEFACE</code> operator.	65
4.6	The <code>REMOVEFACE</code> operator.	67
6.1	Post-processing of an initial correspondence set C_0 between two input graphs G_A and G_B . The method is performed until the assignment energy variation is smaller than a given threshold value t	96

INTRODUCTION

Before everything else, getting ready is the secret of success.

— Henry Ford

Due to their complexity and risks involved, many surgical procedures require extremely careful planning. Procedures such as the ablation of liver tumors, where a needle is used, which needs to be inserted directly in the center of the tumor in order to cauterize it from the inside to the outside, or liver resection, which describes the removal of a part of the liver with the potential purpose of eliminating cancerous tissue, require very cautious consideration. In the case of tumor ablation, the needle must be inserted through several tissues including the skin of the patient in order to reach the tumor without harming any crucial organ, such as the lungs, or hitting solid structures, such as bones, which cannot be punctured. In the case of a liver resection, a part of the liver must be removed without damaging any vital arteries or veins and at the same time leaving enough organ volume behind to guarantee the physiological functions of the liver.

Surgical planning often involves the acquisition of three-dimensional (3D) medical images, such as computed tomographies (CT) or magnetic resonance imaging (MRI), which provide the surgeon with an overview of the patient's anatomy and potential pathologies. Given this information, the surgeon is able to identify a target structure, such as a tumor, and plan a path towards that target, along which a needle can safely be inserted. Alternatively, the surgeon can plan a cut to isolate a certain section of a partially dysfunctional organs, as is done during liver resection.

Unfortunately, a careful surgery plan does not always coincide with a successful surgery itself. Without assisting techniques, the implementation of the surgery plan is not trivial, as the patient's condition might change significantly from the moment when images used for planning are acquired to the moment when the surgery is performed: Among others, organ displacement, breathing, and heart beats, are some of the issues that must be dealt with during surgery. In this complex setting, the surgeon must still be able to adapt the surgery plan to the altered situation. Therefore, computer-assisted interventions are increasingly gaining significance in the surgical routine. Nowadays, several computer-based systems exist to assist surgeons in the

real-time localization and visualization of organs, regions of interest and structures of risk, and to guide them towards target structures (see [246, 96, 206, 277, 88, 27, 273] for more extensive reviews on those systems).

Critical for surgery guidance is the computation of an alignment between the (virtual) space where the surgery planning was performed and the space where surgery is taking place. This task is called *registration*. Most existing commercial systems focus on assisting interventions near rigid structures, like bones, where they can rely on static anatomical landmarks or attach fiducial markers for the alignment of the spaces (e.g. BrainLab VectorVision[®]). In this scenario, the relative position between guidance landmarks does not change and, assuming that the patient has been immobilized, registration between planning and surgery must only be performed once. However, researchers, have drawn their attention to a more complicated problem: computer guidance for soft-tissue interventions. In contrast to rigid structures, soft tissues are continuously deformed due to respiratory motion, heart beats, and external forces (e.g. surgical manipulation). In this case, anatomical landmarks are usually not clear or not reliable due to deformations. Even if one resorts to fiducial markers as artificial landmarks, their position relative to each other is constantly changing. Computer-assisted guidance in soft-tissue interventions therefore remains very challenging.

Several authors have proposed the use of optically or magnetically tracked fiducial markers for the compensation of deformations during soft tissue interventions [152, 160, 194, 169, 170, 161, 180, 28, 195]. These fiducials are either placed above the skin or inside the organ of interest (needles), and are continuously tracked in order to determine their position. Deformation models are employed in order to estimate the location of the targets¹ based on the movement of the fiducials. The main drawback of the use of fiducial markers as surgical guidance aids is that the spatial configuration of the markers around the target is crucial for the minimization of target registration errors [102, 101, 73]. Markers placed above the skin are usually too far from the targets, being less reliable for target localization. While needles can be used for a more reliable target localization, they are of a more invasive nature. After the attachment of the markers, 3D medical image acquisition must be repeated in order to establish the position of the anatomical structures in relation to the markers, exposing the patient to extra radiation, in the case of computed tomographies, for example. Furthermore, fiducial markers are not particularly suit-

¹ We denote as *target* any pre-defined point or region of interest whose position is to be found during surgery.

able for open body surgery, as it would require the placement of the patient in an 3D image acquisition device with the body open and with the entire surgical apparatus. Nevertheless Weber et al. [266] and Markert et al. [184] presented some efforts for attaching markers on the liver surface, but never performed a clinical evaluation. Maier-Hein et al. [179] investigated the optimal combination of skin markers and needles in order to maximize accuracy and minimize invasiveness.

Image-based techniques have also been employed for guidance [133, 267, 23, 130, 269, 98, 163, 198]. These techniques usually involve the segmentation of organs, targets and instruments from real-time imaging modalities (*e.g.* ultrasound), or the direct image-to-image registration between pre- and intra-operative data. Image-guided techniques for the robust and accurate compensation of soft-tissue motion are still subject of ongoing research [177]. Imaging modalities such as ultrasound require direct body contact with the patient and are usually very noisy, while intra-operative X-ray and tomography devices (C-arms) expose the patient to additional radiation and require the isolation of the operation room during image acquisition. The combination of image-guided techniques with the use of fiducial markers has also been investigated [227, 151, 201, 237]. Here, the main task is the identification and extraction of the markers from the images, which constitutes a less complex problem than directly identifying organs, anatomical landmarks, and instruments.

Surface-based intra-operative registration is an attractive alternative for intra-operative guidance, as common surface scanners (1) Do not require contact with the patient, thus also being suitable for open body surgery; (2) Do not expose the patient to radiation; (3) Do not require isolation of the operation room during acquisition; (4) Have moderate to high acquisition rates; and (5) Do not require any kind of markers. Having surface models from segmented pre-operative data, registration is obtained by matching intra-operatively acquired surfaces (*e.g.* surface of an organ or the skin) to pre-operatively acquired ones. As in the case of the use of fiducial markers, deformation models are required in order to correctly estimate the location of targets inside a particular volume based on the deformation of its surface. The use of surface matching techniques for intra-operative registration purposes is already being investigated [128, 21, 52, 65, 111, 51, 212, 31, 66, 91]. These authors have focused their attention to fine alignment algorithms known to require an initial alignment in order to converge. They evaluated it for intra-operative purposes and proposed variations in order to make it more robust to initial alignments and to surface deformations. However,

initial alignments are still performed manually or with the assistance of fiducial markers, in some cases requiring two sensors being placed in the operation room: one for surface acquisition and another for marker tracking.

So far, the automatic establishment of correspondences between pre- and intra-operatively acquired surfaces remains elusive, as surface matching in intra-operative settings poses a challenging problem.

1.1 OBJECTIVES

The main objective of this work is to develop and evaluate a surface matching approach for the purpose of intra-operative registration. This approach should feature the following properties:

- *Automatic*: To compute a registration between two input surfaces, the presented approach is supposed to find a set of correspondences between the input surfaces without making any assumptions about their initial position in space. The surface matching approach should work without requiring any manual interaction by the user for the selection of an initial set of correspondences between the surfaces.
- *Robust, accurate and fast performance*: It is expected that the presented approach is able to deal with the different types of surfaces, extent of deformations, and noise levels encountered in intra-operative registration scenarios, being robust to different parameter settings, accurate enough for initializing fine registration procedures, and fast enough for intra-operative purposes.
- *Generic applicability*: The registration approach should be easily configurable for different intra-operative registration scenarios and organs of interest.

1.2 STRUCTURE OF THIS WORK

This work is organized as follows: Chap. 2 gives a more detailed overview of the entire pipeline for surface-based intra-operative registration, pointing out the challenges and problems that must be overcome by the registration procedure presented here. Technologies, techniques, and methods, related to the intra-operative registration pipeline, are also presented. In this chapter, the reader is informed about surface acquisition methods and pre- and post-processing stages required for surface-based registration.

Chap. 3 describes the current state of the art for surface matching. Its main components and processing stages are presented and their applicability in intra-operative registration are discussed. As such, this chapter forms the basis for the work presented in the following chapters.

Chap. 4 presents an efficient data structure for the representation of surfaces as polygonal meshes. This data structure allows for fast adjacency queries among surface elements, and is able to track surface boundaries and other types of surface abnormalities. Efficient representation of surfaces is essential for the fast computation of surface properties, which are used to find correspondences between surfaces.

Chap. 5 presents a framework for the comparison of surfaces of different modalities based on surface properties, such as curvatures. As intra-operative registration involves the registration of surfaces acquired by different equipments, there are intrinsic deformations between the surfaces due to the different acquisition principles. These deformations may have a great influence on the matching procedure, as surface properties on the same anatomical locations may have different values. In addition, the differences between surfaces acquired by time-of-flight (ToF) cameras and computed tomographies (CT) are evaluated.

Chap. 6 describes two new surface matching approaches, aiming to fulfill the requirements of intra-operative registration. These approaches are evaluated with experiments designed to be as similar as possible to the real-life challenges that have to be dealt with during intra-operative registration.

Chap. 7 finishes this work with some concluding remarks, a summary of the contributions, and an outlook for future work.

OVERVIEW OF SURFACE-BASED INTRA-OPERATIVE REGISTRATION

I believe that if you show people the problems and you show them the solutions they will be moved to act.

— Bill Gates

Surface-based intra-operative registration refers to the computation of a transformation that maps a pre-operative coordinate system, acquired by imaging techniques such as computed tomography (CT) or magnetic resonance imaging (MRI), to an intra-operative coordinate system, using the patient's current morphological (shape) information during surgery. Intra-operative registration is used to align and adapt structures, targets, trajectories, and other data previously generated during surgery planning to the patient's situation during surgery, in order to allow for adequate surgery guidance. The term surgery guidance refers to a way of communicating the current status of the surgery in relation to what was planned to the surgeon. In other words this means showing information that helps the surgeon to follow the surgery plan. It may thus involve continuous intra-operative registration and tracking of surgical instruments. For more details on the history of surgical guidance and on the role of registration in the guidance process we kindly refer the reader to [100].

Procedures based on surface-based intra-operative registration are structured as follows (Fig. 1): First, the patient is submitted to a volume scan, such as a CT or MRI, which generates a 3D image (volume). Given this volume, the physician is able to create different models for the different objects (*e.g.* organs, vessels, tumors) by means of image segmentation algorithms (Sec. 2.1). These models are the basis for the generation of surfaces of interest, which will be used for the subsequent registration, but also for the construction of a surgery plan, which determines the course and goals of the surgery (*e.g.* a trajectory for needle insertion that avoids structures of risk, such as the lungs, or a liver resection that avoids injury of major arteries and vessels). Surgery plans are usually built with specialized software. Once surgery is about to start, the remaining task is the adaption of pre-operatively generated information to the patient's current situation. This implies a non-trivial problem, as the organ of interest deforms due to patient movements, breathing, or surgical manipulations. This

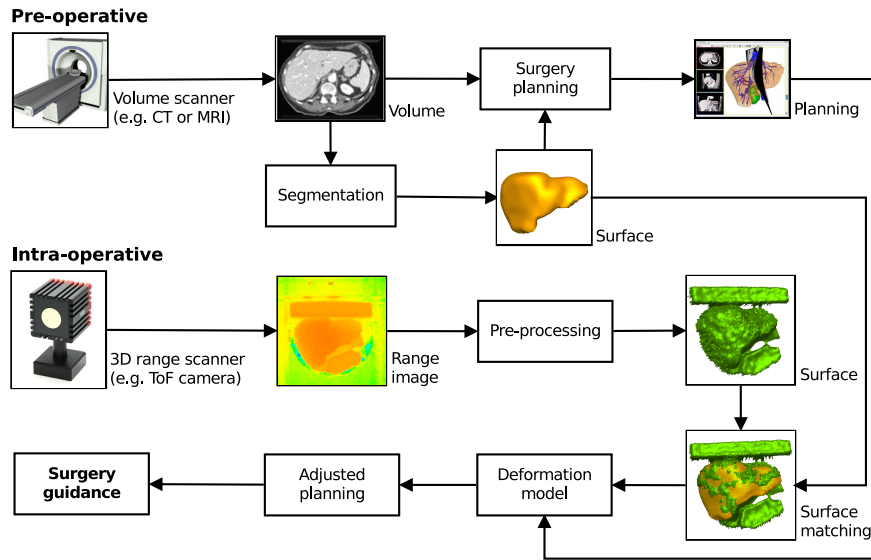


Figure 1: Processing pipeline for surface-based intra-operative registration.

task is accomplished by the acquisition of intra-operative data and by matching it to pre-operative data, in order to establish a spatial relation between the two. With regards to this work, the information used for registration is morphological data represented by surfaces.

Surfaces are acquired intra-operatively by means of 3D range scanners (Sec. 2.2), which generate range images (also known as depth images) with each pixel representing a distance value. Range images are converted into surfaces by a sequence of pre-processing stages (Sec. 2.3), aiming to identify the objects of interest and to compensate for possible image distortions and errors intrinsic to the image acquisition technology. The spatial relation between the pre- and intra-operative situations is established by means of surface matching, which is the main topic of this work, and is reviewed in detail in Chap. 3. After a spatial relation is established, the surgery plan can be adapted to the patient’s current situation for the purpose of guidance. This is achieved by means of a physics-based deformation model, which extrapolates the transformation obtained for the surfaces to the organ’s entire volume.

When designing a method for the automatic establishment of correspondences between pre- and intra-operatively acquired surfaces, the following issues must be taken into consideration, which turns surface matching for intra-operative registration purposes in a challenging and non-trivial problem, as it imposes several challenges:

1. **Partial surfaces:** As 3D scanners do not have a complete view of the region of interest, and there are also many other objects and

structures in the environment, we must deal with a partially overlapping surfaces matching problem. Furthermore, due to the different acquisition principles, holes on one surface may not be present on the other one.

2. **Noise:** Depending on the acquisition rates, surfaces generated by 3D scanners can be noisy. The higher the acquisition rate, the higher the amount of noise on the surfaces.
3. **Distortions:** Due to the different acquisition principles of intra- and pre-operatively generated surfaces (multi-modality), and the different systematic errors that are inherent to these principles, distortions on local and global scales can occur.
4. **Non-rigidity:** As the organ of interest may be undergoing deformation due to respiratory motion or from the surgical intervention, the surface is subject to complex deformations. In this case, the spatial configuration of a set of points on one surface and the configuration of their corresponding ones on the other surface might be different.
5. **Lack of structure:** Usually, surfaces of interest do not present a clear structure or articulation (*e.g.* a clear subdivision of the human body into arms, legs and head) which could be used as reliable landmarks. Matching of surfaces with a clear subdivision into several parts poses a less complex problem, as it resumes to identifying these parts and establishing correspondences between them.
6. **Lack of landmarks:** Surfaces of interest for intra-operative registration often do not present prominent regions or locations, which could be used as reliable landmarks. Surfaces acquired intra-operatively are mostly nearly flat, such as the partial surface of a liver, for example.
7. **Speed:** The entire matching process must be completed in reasonable space of time, *i.e.*, ideally within a few seconds, although within a couple of minutes is acceptable.

Due to these issues, surface matching for intra-operative registration has been used so far only for accounting for displacements that occur in local scales, by means of variations of the *iterative closest point* (ICP) algorithm [128, 21, 52, 65, 111, 51, 212, 31, 66, 91], whereas the actual registration between the intra- and pre-operative spaces occurs manually. Several surface-based navigation systems for craniomaxillo-facial surgery based on ICP already exist, which have been extensively evaluated in the clinical context [185, 176, 167, 35].

In the following sections, we review the stages involved in surface-based intra-operative registration, highlighting relevant work on these topics.

2.1 IMAGE SEGMENTATION

Image segmentation means the partitioning of an image in multiple segments. These segments should make sense in the context of a particular application. In medical imaging, segmentation focuses on the identification of regions and boundaries of organs, tumors, and other anatomical structures, in order to enable the quantification of an organ's specific measurements (*e.g.* volume, blood irrigation, etc). Segmentation allows for the computation of surfaces representing the objects of interest. These surfaces can be used for the purpose of registration, which lies at the basis this work.

For more details on medical image segmentation methods, we refer the reader to [208, 272, 127].

2.2 3D RANGE SCANNERS

Three-dimensional (3D) imaging systems (or range scanners) are devices that measure the distance between the device itself and the objects in its field-of-view. The result is usually an image, referred to as range or depth image, in which each pixel represents a distance value (Fig. 2b). The most widely used 3D imaging systems for surface data are *stereoscopic imaging* [229, 108], *structured light* [225], and *laser range scanning* [197]. A very promising new technology for range scanning are *time-of-flight* (ToF) cameras [156, 71], which are able to simultaneously generate a light intensity image and a range image at high acquisition frequency rates. The drawback of the ToF cameras are the potentially high noise levels, however. ToF cameras are employed in this work for the purpose of evaluation.

For more comprehensive reviews on the topic of 3D range scanning please refer to [36, 164].

2.3 RANGE IMAGES TO SURFACES

Here, we present the required pre-processing steps for the conversion of a range image into a surface representation (polygonal mesh). As intra-operative environments are usually filled with different objects, which may appear in the acquired range images, imposing extra complexity to the surface matching procedure, we present a brief

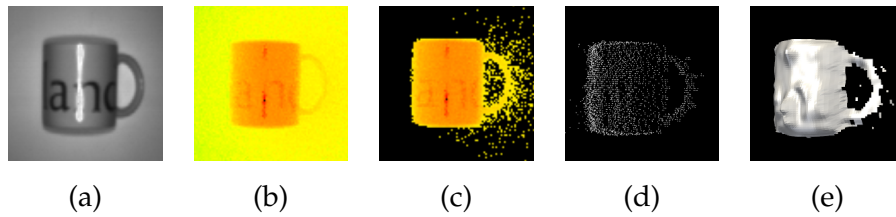


Figure 2: Acquisition of the surface of an object by a time-of-flight camera. (a) The object (a mug); (b) The acquired range image, with distance values mapped to a color palette ranging from green (further) to red (near); (c) Extraction of the object of interest; (d) Conversion of the range image into a 3D point cloud; (e) Triangulation of the point cloud in order to build a surface.

overview of object recognition in range images in Sec. 2.3.1, which aims to isolate the object of interest (region or organ) from other objects. In Sec. 2.3.2 we describe the conversion of range data in a surface representation, focusing on triangle meshes, which are one of the most commonly used discrete representations for surfaces. Finally, Sec. 2.3.3 presents data structures for the representation of meshes. These data structures allow the efficient computation of mesh properties, which are required for surface matching.

2.3.1 Object recognition

Object recognition deals with the problem of identifying a particular object in an image. It has been subject of research for more than four decades [276]. Major problems in the field of object recognition, are the detection of objects from different camera points-of-view, lighting conditions and partial occlusion by other objects in the scene [193]. The different approaches for object recognition are usually classified as follows: *Geometry-based approaches* [193], which employ the geometric properties of the object of interest, and its projection on 2D planes; *Appearance-based approaches* [219], which attempt to capture the appearance variations of the object; and *Feature-based approaches* [230], which focus on finding points-of-interest that are used to characterize the desired object. Object recognition is closely related to image segmentation (see Sec. 2.1), and, in fact, many object recognition approaches employ certain segmentation techniques.

In the case of range images, the use of appearance-based approaches is somewhat limited, as the appearance of a particular object in the image varies according to its distance to the camera. However, some scanners, such as the time-of-flight camera, can simultaneously acquire a light intensity image, which can be used for appearance-

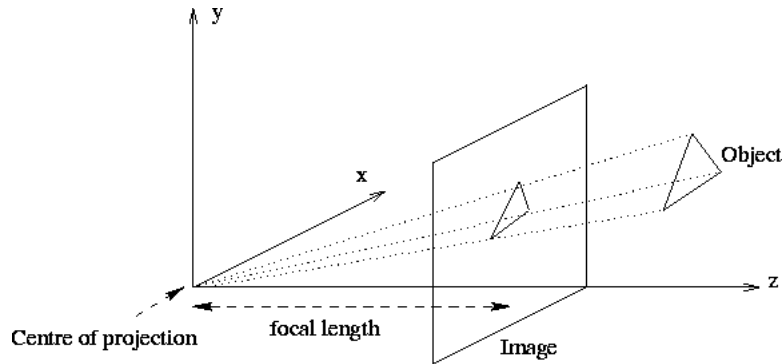


Figure 3: Projection of an object in 3D space to the 2D space of an image according to the *pinhole camera model*.

based detection. Several authors have focused on the specific problem of detecting objects in range images, as can be found in [171, 222, 83].

2.3.2 *Surface generation*

In order to construct a surface from a range image, the first step is to convert the image into a spatial representation: each pixel in the range image space is converted to a 3D point (Fig. 2d). This conversion is performed according to the *pinhole camera model* (Fig. 3), which allows the computation of the original 3D position of the object based on its projection onto the image plane [125]. Furthermore, several image distortions may occur due to lens effects and measurement principles, which must be compensated for correct geometry computation [125, 136, 283]. The second step in the conversion process is the establishment of neighborhoods for each point, representing these neighborhoods by edges, in order to form meshes (Fig. 2e). Triangle meshes are the most widely employed discrete representation for surfaces, although meshes composed by other polygons are also possible [153]. Kilgus et al. [154] investigated and compared several alternatives for the generation of triangle meshes from point sets generated by time-of-flight cameras.

2.3.3 *Data structures for mesh representation*

The representation of meshes using appropriate data structures is advantageous for several reasons, including the accelerated retrieval of adjacency and incidence information, the efficient object traversal, and the maintenance of topological consistency during manipulation. The use of efficient data structures may be critical for achieving the

required performance in the case of surface matching, where the computation of neighborhoods and surface properties (*e.g.* curvatures) becomes necessary. Also desirable for a mesh representation data structure is the ability to represent the boundaries of the meshes, as objects scanned by range scanners have open boundaries. The detection and representation of mesh anomalies, such as *non-manifoldness*¹, is also crucial. Most widely employed data structures for mesh representation are the *winged-edge* [24, 26], the *quad-edge* [122] and the *doubly-linked face list* [58, 3], for the representation of perfect manifolds. The *half-edge* representation [183] can also be used for representation of boundaries. Further details on data structures for mesh representation can be found in Chap. 4.

2.4 DEFORMATION MODELS

Deformation models are employed to extrapolate deformation information on one part of the organ to the rest of its entire volume. These models are usually based on physically coherent models, such as the *finite element method* (FEM) [117], in order to compute extrapolations that are plausible with the physical reality. Literature on deformation models can be found, for example, for the kidneys [199], the liver [52, 72] and the brain [290]. A model for the computation of internal abdominal motion based on the skin deformation was presented in [135].

¹ A mesh is said to be a *manifold* if, for every point, the surface is locally equivalent to an open disk. If this is not the case, the mesh is said to be a *non-manifold*.

STATE OF THE ART OF SURFACE MATCHING
APPROACHES

A lot of people in our industry haven't had very diverse experiences. So they don't have enough dots to connect, and they end up with very linear solutions without a broad perspective on the problem. The broader one's understanding of the human experience, the better design we will have.

— Steve Jobs

Surface matching is a large field and it has been studied for several years. The primary goal of surface matching is to compute a mapping from one surface onto another, and it can be regarded as an optimization problem. Because of the many existing approaches for matching surfaces, classifying them into meaningful groups is not trivial. Audette et al. [22] addressed this issue by first decomposing the surface matching procedure into its main stages, followed by classification of the methods according to how each stage was approached. Similar to Audette et al. [22], we also consider three different stages: *descriptor representation* (Sec. 3.1), *optimization* (Sec. 3.4) and *transformation* (Sec. 3.5). However, another component plays a major role in registration, and it is relevant to analyze it on its own: the *error metric* (Sec. 3.3), which is attempted to be minimized during optimization. Another important stage, commonly used in surface matching for reducing the search-space and increasing the chances of finding a correct match is the *selection of surface features* (Sec. 3.2).

In the *descriptor representation* stage, global or local surface information is extracted, which can be used for surface comparison purposes during the optimization stage. This information is called a *surface descriptor*. In order to make comparisons possible, a distance metric between two descriptor instances must be defined alongside the descriptor itself. The *error metric* is used to determine how two surfaces fit to each other in any given stage of the matching process. *Optimization* denotes the process of finding a match between the input surfaces, such that the error metric is minimized. After the establishment of correspondences, a mapping between the input surfaces is computed on the *transformation* stage. Note, however, that the computation of a transformation does not necessarily occur only after the optimization stage is finished. In several methods, transformations are computed

during the optimization process in order to identify correspondence sets with minimal error.

Generally speaking, given two surfaces represented by point samples, a source surface $S = \{s_i\}$ and a target surface $T = \{t_j\}$, where s_i and t_j denote the surface samples for source and target surfaces, respectively, their descriptors D_S and D_T , and an error metric $E(\cdot)$, the goal of surface matching is to find a mapping $\Phi : S \rightarrow T$ represented by a transformation operator $A : S \rightarrow T$, so that:

$$\Phi = \arg \min_A E(S, T, D_S, D_T, A) \quad (3.1)$$

3.1 DESCRIPTOR REPRESENTATION

Descriptors are pieces of information extracted from surfaces in order to make the comparison between two surfaces, or two surface parts, possible. They can be local or global, and can be used to subdivide the surfaces into regions with similar properties. Descriptors must be comparable to each other and, along with the descriptor itself, a distance metric between two descriptor entities has to be defined as well. For intra-operative registration purposes, the descriptor must be able to provide robust characterization even in the presence of noise. Another aspect to be considered is the discriminative power of the descriptor: If the descriptor is too sensitive, a small variation on the surfaces will result in a high distance value between the descriptors. As surface-based intra-operative registration must deal with surfaces acquired from different sensors and, therefore, different acquisition principles, variations on the surface representations for the same anatomical location should be expected (see Chap. 5). In this case, an over-sensitive descriptor would be inadequate. However, as scanned surfaces are usually nearly flat, since they have only a partial view of the object-of-interest, a certain degree of sensitivity is required in order to effectively discriminate the different regions of the surface. Since obtaining a perfect balance in discriminative power is very difficult, the optimization procedure must be able to deal with this issue. Furthermore, as we deal with partial surfaces, the descriptors should be able to represent information in local scales, instead of global ones. In addition, the descriptor should be invariant with respect to the occurring transformation class (see Sec. 3.5).

The most basic form of a surface descriptor is the surface *geometry* itself, *i.e.*, its points, edges, and faces. Geometry was used to find the set of four congruent points that, when mapped to each other, delivers an alignment with the maximal intersection area [2]. This method is robust to noise and very effective for rigid registra-

tion. Geometry was also used to search for a mapping between two surfaces, which minimizes some kind of deformation error between them [285, 281]. Although these approaches are very effective, they are time-consuming and usually not applicable for surfaces without prominent features (see Sec. 3.4). Another class of methods that employs geometry as a descriptor are the variants of the *iterative closest point* algorithm (ICP) [34], which iteratively finds pairs of closest points and computes a transformation that maps these points onto each other (see Sec. 3.4.2.1 for more details). This class of methods are employed for the fine alignment of surfaces as well as for accounting for displacements that occur in local scales, as an initial alignment must be provided in order to ensure convergence to the global minimum distance. Other methods for the fine alignment of surfaces based on pure geometric information were presented by Eckstein et al. [93], Papazov and Burschka [202].

Based on geometry, a global form of shape description is the *principal component analysis* (PCA) [234], which computes the principal axes of shape variation, using the eigenvectors associated with the largest eigenvalues of the second order moments covariance matrix. Matching two shapes using PCA implies aligning their principal axes. As mentioned before, PCA can usually only be used for aligning entire surfaces, as the principal axes extracted from a partial surface are incompatible to the ones extracted from the whole surface. Furthermore, PCA does not provide directions of the principal axes, thus matching solely based on PCA can be ambiguous.

Bronstein et al. [44], Eckstein et al. [93], Papazov and Burschka [202], Sahillioğlu and Yemez [223] employed *geodesic distances* for matching. A geodesic is defined as a curve which realizes the shortest distance between any two points lying on a general metric space Jost [144]. In the case of surfaces (2-dimensional embeddings), the geodesic distance between two points lying on a surface denotes the length of the shortest line above the surface that connects these two points (Fig. 4). Only relying on geodesic distances for matching, can be inconclusive in the intra-operative case, as surfaces are mostly nearly planar, and every point has similar geodesic fields (Fig. 4b), differently from surfaces that have a more complex structure (Fig. 4c). Furthermore, geodesic distances are not robust to noise. Distances are usually longer on noisy surfaces because of the high frequency variations on the surface.

A slightly improved form of point (local) descriptors, which considers the geometry of its neighbors but not the global geometry, are the principal curvatures and other curvature-based quantities (Fig. 5): *mean curvature*, *Gaussian curvature* [189], *shape-index*, and *curvedness*

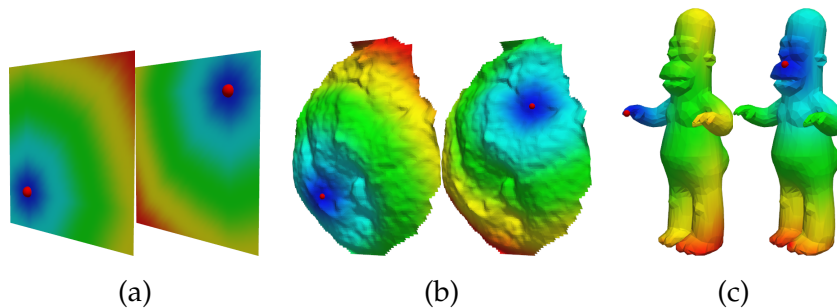


Figure 4: Geodesic distance fields for two distinct points on different surfaces (red: high, blue: low): (a) A plane. Both points have exactly the same radial geodesic field. (b) A porcine liver surface acquired by a time-of-flight camera. As the surfaces are nearly planar, the geodesic fields of both points are similar. They are also very similar to the radial fields on the plane surface. (c) A humanoid shape. As the structure is more complex than the previous surfaces, not resembling a plane, distinct points on the surface have different geodesic fields (unless they lie close to each other).

[155]. These measures are related to the way a surface bends in a particular point, *i.e.*, how the surface differs from a plane at a particular point. As the computation of curvatures on discrete surfaces is not robust to noise, Cazals and Pouget [54] proposed a method for the computation of differential properties by fitting a smooth polynomial to the local neighborhood, followed by computing the curvatures of this polynomial (Fig. 5c). Curvatures have been used as descriptors for many years. Kehtarnavaz and Mohan [148] segmented the surfaces in patches of homogeneous curvature and employed graph matching to obtain correspondences between them. Other methods employing curvature as a measure for data likelihood were presented by Zeng et al. [281, 282], Windheuser et al. [271]. The problem, however, is there may be many points on one surface that have the same curvature values as another point on the other surface [115]. Curvatures are also used to select features on the surface, in order to reduce the search-space for correspondence search and to increase the chance of finding correct matches by using only more prominent and distinguishable points (see Sec. 3.2).

In order to increase the discriminative power and robustness of the curvature descriptor, Gatzke et al. [113] subdivided the geodesic circle around a particular point in bins, and computed the average curvature for each bin. The descriptor itself is represented by a vector, where a curvature average is stored at each vector's position, and each position represents the curvature value of a particular bin. Two descriptor entities are compared by means of the L_2 norm. John-

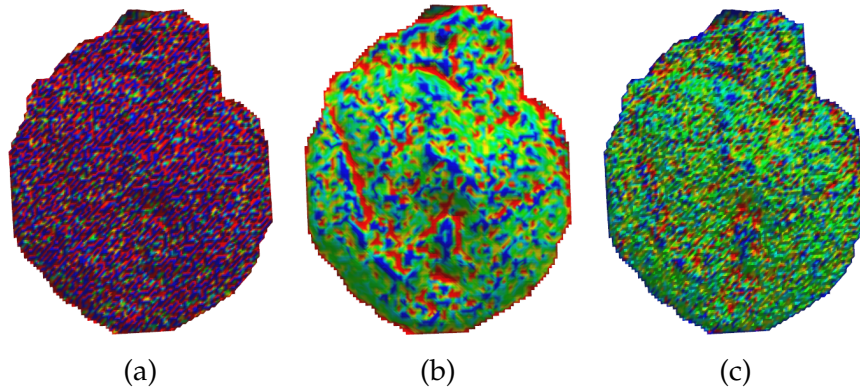


Figure 5: Mean curvature plot on a porcine liver surface acquired by a time-of-flight camera. (a) Curvature computed according to the discrete approach of Meyer et al. [189] on the original surface (noisy); (b) Curvature computed according to the same approach on the smoothed surface; (c) Curvature computed according to the noise-robust approach of Cazals and Pouget [54] on the original surface. The curvature values on the smoothed surface (b) and on the original surface computed with a more robust approach (c) are highly similar.

son and Hebert [141, 142] presented the *spin-images* (Fig. 6), one of the most well-known descriptor for shape matching. Using the same idea of binning, they computed a 2D histogram of points that are contained inside a spherical volume by means of a plane rotating around the normal of a particular point. The descriptor proposed by Frome et al. [105], known as *3D shape context*, extends the idea of spin-images for a 3D spherical volume defined around a point, thus partitioning the sphere in bins by inserting subdivisions in the radial, azimuth, and elevation directions. However, in order to consistently index the 3D bins to the 2D vector that represents the descriptor, they rely on the determination of a local coordinate system for each point, which must be repeatable across surfaces. The same applies to the fingerprint-like descriptor presented by Sun and Abidi [249], Sun et al. [250], which is obtained by the projection of geodesic circles around a point to its tangent plane. Tombari et al. [257, 258] improved the robustness of the computation of local coordinate systems, and presented the *signature of histograms* descriptor, which computes histograms of angles between normals, instead of computing histograms of point positions. The *MeshHog* descriptor [279] also applies an improved computation procedure of the local coordinate system, and computes histograms of mean curvature gradients. The robustness and repeatability of several descriptors based on the computation of local coordinate systems was investigated by Petrelli and Di Stefano

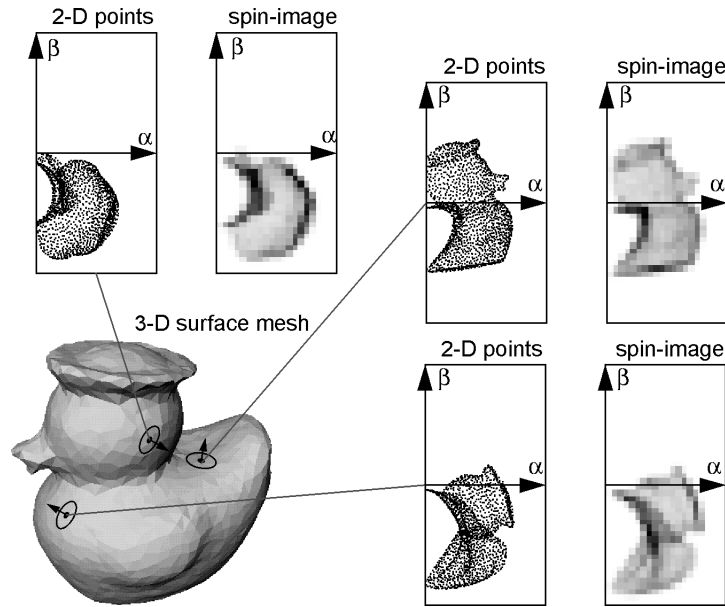


Figure 6: Spin images for three points on the surface of a duck model (image found in [142]).

[207], who performed several experiments with meshes of different point densities and noise levels.

For the global description of closed, articulated surfaces, skeletonization methods have been employed (Fig 7) [19, 288, 251, 50]. Skeletons naturally incorporate the notion and the representation of parts and articulations. They are represented by trees, and matched by minimizing the differences between properties of branches and nodes. Global shape representations were also presented by Toldo et al. [256], Bronstein et al. [46], who computed a geometric vocabulary by clustering the descriptor space, which could be any local descriptor in this case. Each point descriptor is then represented in the vocabulary using vector quantization. The global descriptor is computed as the histogram of quantized local descriptors.

Gelfand et al. [115] computed the volume of the intersection between a sphere centered on a point and the surface for local shape description. This descriptor is called *integral volume descriptor*. Pottmann et al. [211] performed an analysis of robustness and stability of different integral computation methods, focusing on volume descriptors.

In a more elaborate manner of extracting surface information, authors represent surfaces as a series of functions of different frequencies (bands), defined on the sphere, in the same way as a Fourier series, but in 2D instead of 1D [190]. This technique is called *spherical harmonics*, and it can be employed for both the local and the global

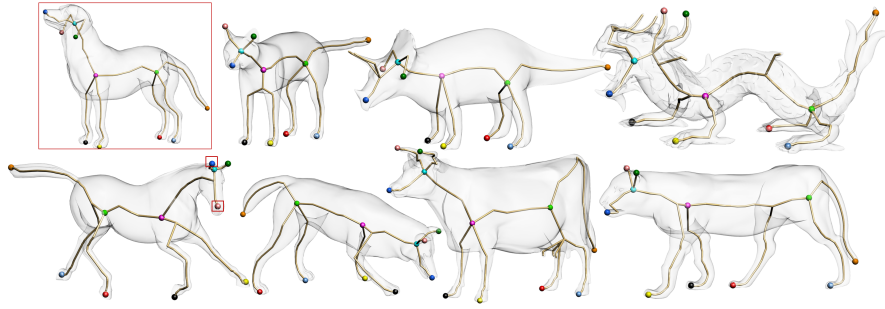


Figure 7: Matching a dog to different four-legged animals with different surface details by means of comparison of their surface skeletons (image found in [20]).

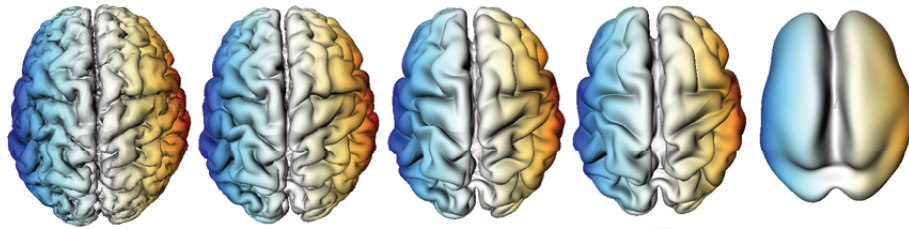


Figure 8: Representation of the cortical surface by spherical harmonic series of different highest frequency bands (image found in [61]). From left to right, the highest frequency band in the series decreases.

description of shapes. The global descriptors presented by Kazhdan et al. [147], Funkhouser et al. [107] store the amplitude of spherical harmonic coefficients within each frequency for shape retrieval in databases. Frome et al. [105], Funkhouser and Shilane [106] adapted this idea for local surface description, by constraining the spherical harmonics into local support volumes. Spherical harmonics have been extensively researched for the description of cortical surfaces (Fig. 8) [61, 62, 150]. An advantage of this representation is its robustness to noise when using lower frequency bands only, as noise is generally represented by higher frequencies.

More recently, attention has been drawn to the *manifold harmonics* [254, 260, 215, 286, 85], which are a generalization of the spherical harmonics for arbitrary manifolds. Importantly, the spherical harmonics are related to the manifold harmonics: the manifold harmonics series on a sphere is exactly the basis of the spherical harmonics series [260]. The advantage of the manifold harmonics representation is that it is invariant to isometric deformations¹, even very large ones, as the fre-

¹ Quoting from Beardon [29, p. 89]:

A map $f : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is an *isometry* if it preserves distances; that is, if for all x and y , $\|f(x) - f(y)\| = \|x - y\|$.

quency series is not defined on a sphere, like the spherical harmonics, but on the space spanned by the surface itself (Fig. 9a-b). Reuter et al. [214], Rustamov [221] employed manifold harmonic bands for noise-robust and isometry-invariant registration and database retrieval of complete models. However, because manifold harmonics are defined in the surface space, instead of a common space, it is not applicable to the registration of partial surfaces, since the manifold harmonic bands are incompatible and only make sense in the surface space itself (Fig. 9c). Furthermore, their computation is very expensive, as it involves the computation of eigenvalues and eigenvectors of large Laplacian matrices.

Based on the manifold harmonics, Sun et al. [248] presented the *heat kernel signatures* (HKS), which model the amount of heat that is transferred between two points in a given amount of time, assuming one of the points as a heat source. As this descriptor is defined between two particular points, similar to the geodesic distance, and not on a global scale, like the manifold harmonics, it can be used for the registration of partial models. Dey et al. [84], Zobel et al. [291] improved the HKS for better registration and retrieval of partial and incomplete models. Although the computation of heat diffusion is robust noise, in contrast to geodesic distances, the diffusion in time occurs inversely proportional to the geodesic distance (Fig. 10). This implies in higher amount of heat transfers occurring in shorter distances in shorter time. Thus points on nearly planar or planar surfaces have approximately the same heat diffusion fields, as they have similar geodesic distance fields. For the same reasons as mentioned for the geodesic distance fields, heat-based descriptors can be ambiguous for matching surfaces of interest of intra-operative registration. In addition, as HKS is based on manifold harmonics, their computation is also very expensive.

Comparisons and surveys on descriptor representations were presented by Tangelder and Veltkamp [253], Bustos et al. [49], Iyer et al. [139], Bronstein et al. [45], Heider et al. [126].

3.2 FEATURE SELECTION

Feature selection is the process of identifying surface points that are unique on the surfaces, *i.e.*, points that can be used as reliable landmarks for the matching process. For instance, such points can lie on the extremities of a surface (Fig. 3.2). Feature selection is performed

Isometric deformations on surfaces means that distances are preserved in a geodesic sense. In the case of surfaces, being isometry-invariant means being invariant to initial shape alignment, translation, rotation, scaling, and non-rigid bending [274].

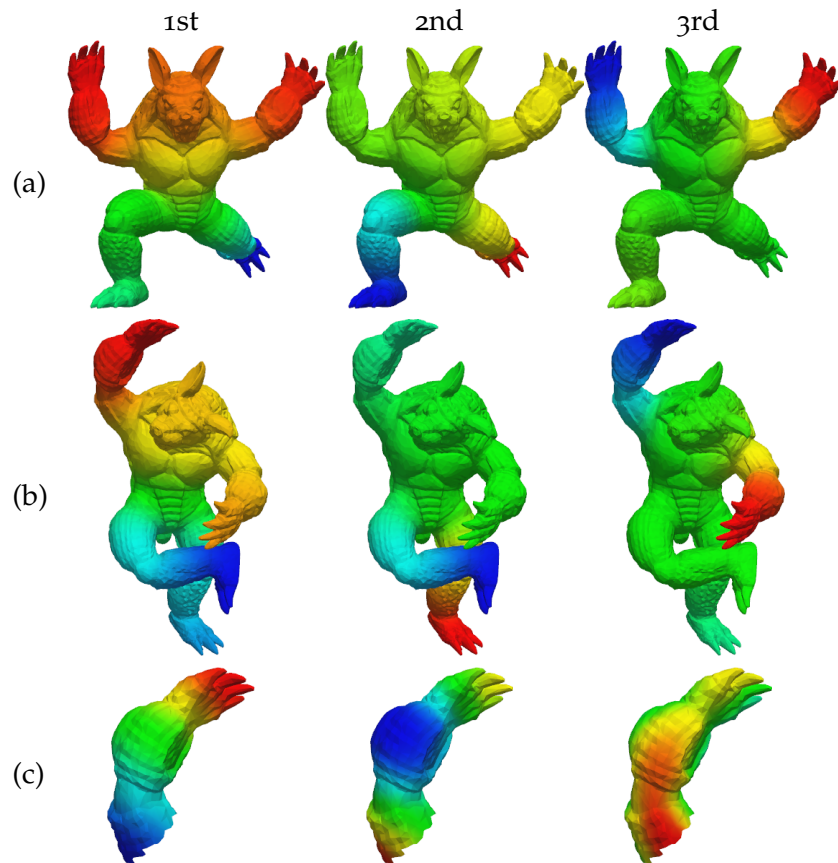


Figure 9: Manifold harmonic bands of two Armadillo models, and one of the Armadillo's arm. The columns show the first, second and third bands of their manifold harmonics. As the models in rows (a) and (b) can be nearly mapped to each other by an isometry, the bands are similar, as the spaces spanned by these surfaces are similar. However, the partial surface in row (c) spans a different space, which is not isometric to the other models, thus inducing different bands, defined in its own space.

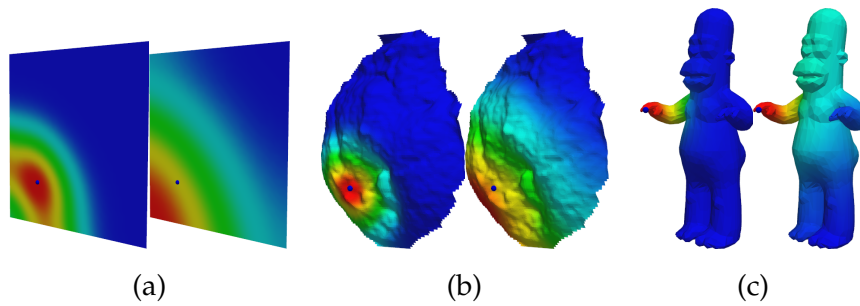


Figure 10: Heat diffusion field of a heat source (blue point) on different surfaces (red: high, blue: low), at different time points (125 and 500 seconds): (a) A plane. (b) A porcine liver surface acquired by a time-of-flight camera. (c) A humanoid shape. The heat diffusion is inversely proportional to the geodesic distance (Fig. 4).

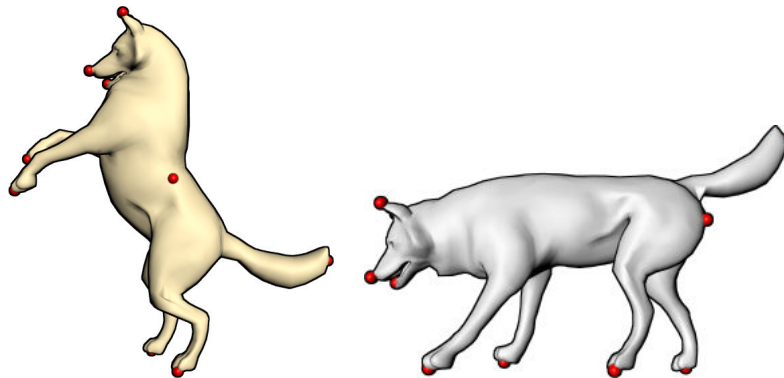


Figure 11: Features selected from two different dog surfaces (image found in [285]). Note that features are consistently selected on both surfaces, *i.e.*, they are chosen on the same locations.

mainly for two purposes: First, to speed-up the registration by only using a subset of the data. Second, to increase the probability of finding a correct match between surfaces by using only prominent and reliable landmarks, and not other more common points, which are more likely of having many similar ones. It is important to note, however, that, if feature selection is used, and correspondences shall be found only among the selected features, points lying on the same locations on both surfaces must be selected. If points lying on different locations are selected, finding a correct match between the surfaces is impossible. Consistent feature selection across surfaces is crucial for the successful registration of surfaces by methods relying on feature selection.

Methods for feature selection include: Random selection [191, 105]; Selection based on surface curvature [275, 103, 279]; Saliency [166, 109,

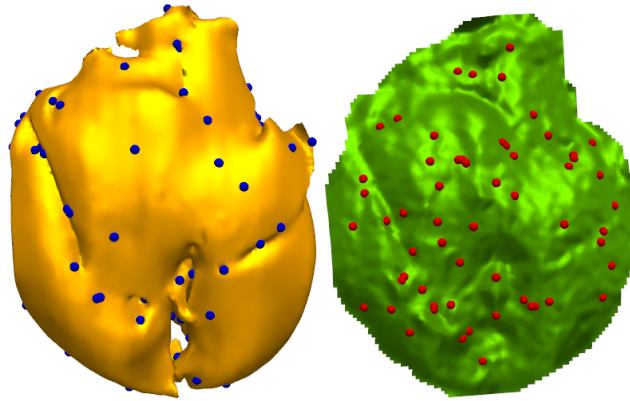


Figure 12: Comparison between selected feature points on two surfaces of the same object (porcine liver), as acquired by a computed tomography (left) and a time-of-flight camera (right). Features were selected according to the multi-scale approach of Ho and Gibbins [131].

53]; Persistence across different scales [204, 115, 196, 131]; Number of similar points on the other surface [232]; Maximums of heat propagation [248, 114]; Tangential discontinuity [265]; Analysis of gradients [116, 238].

The consistent selection of features on surfaces of interest for intra-operative registration is not trivial. Because of the noise, lack of structure (no articulations), and lack of prominent and reliable landmarks (the surfaces are nearly flat), the consistent selection of features across pre- and intra-operative surfaces is usually not possible (Fig. 12). Methods for surface matching that rely on feature selection, are likely to fail in intra-operative registration scenarios.

3.3 ERROR METRIC

Error metrics are used to determine how well two surfaces match to each other in any given state of the optimization process. Error metrics can employ, among others, geometric information, descriptor similarities, or distortion measurements, in order to estimate an error value for the current optimization state. In Sec. 3.3.1 we focus on error metrics that are used to establish correspondences between points, while in Sec. 3.3.2 we show some error metrics for establishing correspondences between other kinds of non-point-based descriptors, such as regions and skeletons.

3.3.1 Error metrics for point correspondences

Let us assume two surfaces represented by discrete samples (points): The source surface $S = \{s_i\}$, and the target surface $T = \{t_j\}$. The goal is to find a set of correspondences $C \subset S \times T$ between source and target surfaces that delivers the best alignment with respect to an error metric when these correspondences are used to compute a transformation that aligns both surfaces. The set C is defined by its characteristic function $\sigma_C : S \rightarrow T$, which is partial and injective, *i.e.*, not all points have correspondences, but the ones that do have a correspondence, have a single one.

The most basic form of error metric is the Euclidean distance $d_{\text{Eucl}}(\cdot, \cdot)$ between source and target points. Using the closest point for finding correspondences, the correspondence set can be obtained as follows:

$$\begin{aligned} C = \{ & (s_i, t_j) : t_j = \arg \min_{t_k \in T} d_{\text{Eucl}}(s_i, t_k)^2 \\ & \wedge \forall s_k \in S [(s_k, t_j) \notin C] \\ & \wedge \forall t_k \in T [(s_i, t_k) \notin C] \} \end{aligned} \quad (3.2)$$

If the surfaces are aligned closely enough, the closest point is a simple way of finding correspondences. In practice, however, it is very unlikely to find cases where the closest point could deliver a correct correspondence set. Nevertheless, there exists an entire class of iterative algorithms, derived from the *iterative closest point* (ICP) algorithm [34] (see Sec. 3.4.2.1), that minimizes the global squared Euclidean distance:

$$E_{\text{ICP}}(S, T) = \sum_{s_i \in S} d_{\text{Eucl}}(s_i, \arg \min_{t_j \in T} d_{\text{Eucl}}(s_i, t_j))^2 \quad (3.3)$$

Note that the error metrics presented so far do not incorporate any kind of similarity between points, relying only on geometric information².

Let us assume the function $q : S \times T \rightarrow \mathbb{R}$ that measures the incompatibility between a point on the source surface and one on the target surface, based on their descriptor distances, for example. A simple error metric that incorporates these incompatibility values is defined as the *linear assignment problems* (LAP) [47], which computes the global compatibility error of a correspondence set:

$$E_{\text{LAP}}(C) = \sum_{(s_i, t_j) \in C} q(s_i, t_j) \quad (3.4)$$

² Note that there are variants of the ICP algorithms that do incorporate other kind of similarity metrics. See Sec. 3.4.2.1 for more references on this topic.

The optimization of $E_{LAP}(\cdot)$ delivers the correspondence set with minimal global incompatibility. In contrast to $E_{ICP}(\cdot, \cdot)$, $E_{LAP}(\cdot)$ does not incorporate any kind of geometric information, solely relying on incompatibility measurements between points. A more sophisticated error metric is defined as the *quadratic assignment problem* (QAP) [47], which not only incorporates a first order incompatibility measure such as $q(\cdot)$, but also includes a second order regularization (smoothing) term, which measures the incompatibility between assignments:

$$E_{QAP}(C) = \sum_{(s_i, t_j) \in C} \sum_{(s_k, t_l) \in C} q_2(s_i, s_k, t_j, t_l) + \sum_{(s_i, t_j) \in C} q(s_i, t_j) \quad (3.5)$$

where $q_2 : S \times S \times T \times T \rightarrow \mathbb{R}$. Incompatibility between assignments can be measured, for example, as the difference between the distances of source points and their corresponding ones ($d_{Eucl}(s_i, s_k) - d_{Eucl}(t_j, t_l)$). This means if s_i is assigned to t_j and s_k to t_l , we expect the Euclidean distance between s_i and s_k to be equal to the distance between t_j and t_l , in a rigid case. Employing $E_{QAP}(\cdot)$ for registration problems is a more robust compared to $E_{LAP}(\cdot)$, as, like $E_{LAP}(\cdot)$, it is independent of the initial position of the surfaces, but permits the incorporation of both geometric and descriptor similarities. Unfortunately, Sahni and Gonzalez [224] showed that QAP is NP-hard, and even finding a nearly optimal solution, within some constant factor, cannot be performed in polynomial time.

Gelfand et al. [115] showed that a pure second order, Euclidean distance based error is robust enough to rigidly match partially overlapping surfaces (Fig. 13). They employed the *distance root mean squared* (dRMS) error, which is similar to the second order definition presented above, comparing all internal pairwise distances between corresponding points:

$$E_{dRMS}(C) = \frac{1}{|C|^2} \sum_{(s_i, t_j) \in C} \sum_{(s_k, t_l) \in C} (d_{Eucl}(s_i, s_k) - d_{Eucl}(t_j, t_l))^2 \quad (3.6)$$

The solution of the registration problem using this metric is a QAP and, therefore, NP-hard (see Sec. 3.4 for a discussion on the optimization procedures). Funkhouser and Shilane [106] employed a full QAP error metric for rigid registration, incorporating not only the second order term based on distances, but also the first order term, based on point dissimilarity measures. Chang and Zwicker [55] applied the same QAP error metric, but for the problem of matching shapes undergoing isometric deformations. They identified surface patches that were subjected to the same rigid transformation, and solved the registration as a labeling problem.

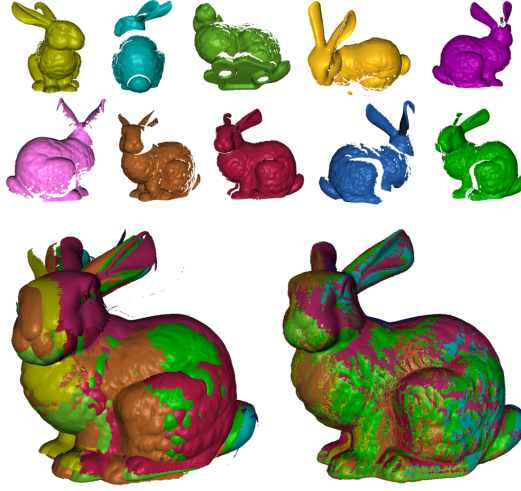


Figure 13: Registration of several partial surfaces of a bunny using the $E_{dRMS}(\cdot)$ (see Eq. 3.6) error metric (bottom left) and after the error minimization of the initial registration with ICP (bottom right).

For the registration of shapes undergoing isometric deformations, the adaptation of the QAP error metric can be performed in a straightforward manner, by replacing the Euclidean distance with the geodesic distance (Fig. 14). Sahillioğlu and Yemez [223] employed only the second order term, directly replacing the Euclidean distances in the $E_{dRMS}(\cdot)$ formulation with the geodesic distance, while Dubrovina and Kimmel [89], Wang et al. [263] formulated their error metrics as a full QAP error, as follows:

$$E_{iso}(C) = w \sum_{(s_i, t_j) \in C} \sum_{(s_k, t_l) \in C} (d_{geo}(s_i, s_k) - d_{geo}(t_j, t_l)) + \sum_{(s_i, t_j) \in C} q(s_i, t_j) \quad (3.7)$$

where $d_{geo}(\cdot, \cdot)$ denotes the geodesic distance between two points on the same surface, and w a weighting scalar to balance the influence of the terms. Raviv et al. [213] used the same error metric formulation, but instead of using geodesic distances, they employed heat diffusion as a distance metric. As explained in Sec. 3.1, heat diffusion is inversely proportional to the geodesic distance, with the advantage of being robust to noise.

Lipman and Funkhouser [173], Zeng et al. [281, 282] employed the deformation error between source and target surfaces, when they are both conformally flattened onto a common canonical 2D domain, as their error metric. This flattening can be uniquely determined by fix-

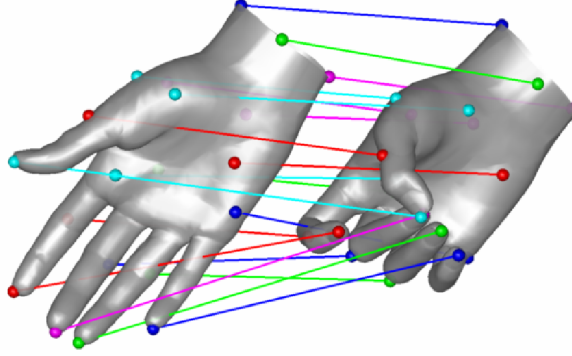


Figure 14: Correspondences obtained for a pair of hand model using the $E_{\text{iso}}(\cdot)$ (see Eq. 3.7) error metric for registration under isometric deformations (image found in [89]).

ing any three points on the surface. Zeng et al. [281, 282] posed this flattening error as a third order assignment problem, as follows:

$$E_{2D_fit}(C) = \sum_{(s_i, t_j) \in C} \sum_{(s_k, t_l) \in C} \sum_{(s_m, t_n) \in C} \theta(s_i, s_k, s_m, t_j, t_l, t_n) + \sum_{(s_i, t_j) \in C} q(s_i, t_j) \quad (3.8)$$

where $\theta : S \times S \times S \times T \times T \times T \rightarrow \mathbb{R}$ denotes the mutual flattening deformation error.

Zhang et al. [285] utilized an error metric that measures elastic distortion and preservation of local differential properties for an as-rigid-as-possible deformable mapping [174, 159] of the entire surface S onto surface T , given a set of k correspondences. Windheuser et al. [271] employed an error metric defined by the elasticity theory as commonly known in physics, which measures stretch and bending error [63, 79]. They posed the error as a linear error metric, computing it for pairs of triangles.

3.3.2 Other error metrics

The measurement of errors not related to point correspondences and relations between point correspondences can also be found in the literature. However, they are not as commonly used as the ones presented in the previous section (Sec. 3.3.1). As in the previous section, we assume two discretely sampled surfaces $S = \{s_i\}$ and $T = \{t_j\}$, the source and target surfaces, respectively.

The most common metric for measuring the global error between two surfaces is the *Hausdorff distance* [218], which is based on the Euclidean distance. This metric computes the maximum of the distances from a point in any of the surfaces to the nearest point on the other surface. In its discrete form, the Hausdorff distance is defined as follows:

$$E_{\text{Hausd}}(S, T) = \max\{\max_{s_i \in S} \min_{t_j \in T} d_{\text{Eucl}}(s_i, t_j), \max_{t_j \in T} \min_{s_i \in S} d_{\text{Eucl}}(s_i, t_j)\} \quad (3.9)$$

Note that the Hausdorff distance is symmetric, *i.e.*, $E_{\text{Hausd}}(S, T) = E_{\text{Hausd}}(T, S)$. Charpiat et al. [57], Eckstein et al. [93] proposed the *pseudo-Hausdorff distance*, which has the advantage of being differentiable with respect to the position of the points of the two meshes [93]. The pseudo-Hausdorff distance converges to the Hausdorff distance with increasing sampling of the surfaces [57]. A global distance metric for isometrically deformed surfaces is the *Gromov–Hausdorff distance* [187], which measures how far two surfaces are from being isometric. The Gromov-Hausdorff distance, in its discrete form, is defined as follows:

$$E_{\text{Gro-Hausd}}(S, T) = \inf_{f, g} E_{\text{Hausd}}(f(S), g(T)) \quad (3.10)$$

where $f : S \rightarrow Z$ and $g : T \rightarrow Z$ are isometric embeddings into the metric space Z . Note the similarity with the error metric presented by Lipman and Funkhouser [173], Zeng et al. [281, 282] (see Sec. 3.3.1, Eq. 3.8), where they measure the deformation error of the embeddings of both surfaces onto a 2D plane.

Aiger et al. [2] used the size of the intersection region between two surfaces as a similarity metric, measured as the amount of points on one surface that are close enough (smaller than a given threshold) to a point on the other. Au et al. [20] adopted the amount of votes casted by all reasonably possible combinations of fitting two skeletons to each other as similarity values for skeleton nodes.

3.4 OPTIMIZATION

Two classes of optimization methods are known in surface matching: The first aims to find a set of correspondences that roughly aligns two shapes and is called *rough-scale optimization* (Sec. 3.4.1). This kind of optimization is usually automatic, *i.e.*, it makes no assumptions about the initial position of the surfaces in space. Most rough-scale optimization methods deliver a sparse set of correspondences. The second class of optimization contains the methods that account for small misalignment and for deformations in the local scale. This class

of optimization is therefore called *fine-scale optimization* (Sec. 3.4.2). It contains the popular *point-based iterative optimization* algorithms (Sec. 3.4.2.1), such as the *iterative closest point* (ICP) [34] and its variants.

An extended bibliography on correspondences search for surface matching problems was presented by van Kaick et al. [261].

3.4.1 *Rough-scale optimization*

One of the simplest solution for obtaining an automatic matching between two surfaces is to minimize the *linear assignment problem* (LAP) error (Eq. 3.4 on page 26) using the distance between descriptors (see Sec. 3.1) as a measure of dissimilarity between points. Minimizing the LAP error metric is equivalent to obtaining a set of point correspondences between source and target surfaces, so that every point on the source surface is assigned to its most similar point on the target surface, in a way that the sum of the distances between the descriptors of assigned points becomes the minimum. As can be seen in Eq. 3.4, the error metric minimized for the solution of the LAP does not contain any information about the relative position of the surfaces, which is irrelevant for this problem. The LAP is one of the oldest and most studied problems in combinatorial optimization [48] and there are several algorithms to solve it in polynomial time. Under certain conditions, it can even be solved in linear time (see [48] for a review on these algorithms).

However, solving the surface matching problem as a LAP may be error-prone for two reasons: (1) As LAP only considers descriptor similarities, and usually there are multiple correspondence configurations with compatible descriptors, finding the correct match may be difficult, as the problem becomes very ambiguous; (2) As LAP does not incorporate any regularization term, ensuring that points in a particular neighborhood on the source surface will be assigned to points that also belong to a common neighborhood on the target surface, it may result in a lack of geometric consistency among correspondences. Furthermore, it is important to note that the formulation of the surface matching problem as a LAP finds a global matching between the possible correspondences. This means that a correspondence will be assigned for every point on the source surface, assuming that the source surface has less points than the target one, and that no feature selection was performed. If the surfaces represent areas that are only partially overlapping, solving a global assignment problem cannot, in any manner, deliver a correct set of correspondences, as the points that do not belong to the overlapping area should be left unassigned. For partial surface matching, in the case of LAP, one can resort to the

k-cardinality assignment problem (CAD) [80]: Given an integer *k*, one wants to find *k* correspondences from source to target surface, so that the sum of distances between the descriptors of corresponding points becomes the minimum.

The *quadratic assignment problem* (QAP) error (Eq. 3.5 on page 27) incorporates a regularization term for neighborhood consistency of the correspondences. However, solving a QAP is known to be NP-hard [224]. Still, there are two common methods for solving QAP (see [203, 42] for more details): The first class of methods performs a combinatorial analysis based on search procedures, such as *greedy search* or *branch-and-bound* [165]. Although a full combinatorial analysis of the correspondences space guarantees that the global minimum is found, it may be untreatable to search the entire space. Therefore, many authors resort to measures that constrain and reduce the search-space. The second class of methods is known as *probabilistic relaxation* [245], where the constraints of the QAP are relaxed to allow fuzzy correspondences, casting the problem as a convex continuous minimization problem. Although the problem becomes treatable due to relaxation, it is prone to follow into a local minimum. In the rest of this section, we review the solutions found in recent publications on surface matching.

Branch-and-bound optimization was employed by Gelfand et al. [115], Dubrovina and Kimmel [89], Raviv et al. [213]. While Gelfand et al. [115] employed it for the minimization of the $E_{\text{dRMS}}(\cdot)$ error metric (Eq. 3.6 on page 27), based on Euclidean distance, thus only applicable to rigid registration, Dubrovina and Kimmel [89], Raviv et al. [213] minimized the $E_{\text{iso}}(\cdot)$ error metric (Eq. 3.7 on page 28), which employs geodesic distance, thus being also applicable to the registration of isometrically deformed surfaces. Branch-and-bound is based on the enumeration of all possible solutions while discarding the solutions with an error greater than the current error. In fact, branch-and-bound optimization can be most efficiently implemented as a search-tree, where the correspondences are represented as nodes while a path from a leaf to the root determines a possible solution. During the construction of the search-tree, every time a new branch is added to a node, the error of this partial solution is computed: If the error rises above the current error, the branch is pruned, and the space is not searched in this direction anymore. Otherwise, the tree keeps expanding until the solution is complete. The path with the smallest error is the correspondence set with minimum error. In order to reduce the search-space, the authors employ feature selection, feature clustering, distance consistency tests during tree construction [115].

Funkhouser and Shilane [106], Zhang et al. [285], Au et al. [19], Sahillioğlu and Yemez [223] used a *constrained greedy optimization* approach, or *exhaustive search*, which enumerates and tests the entire search-space. For faster computing, they employed several constraints to reduce the search-space before starting the optimization. Importantly, a common search-space constraining measure was adopted by all of them: the selection of very few and trustworthy features. Funkhouser and Shilane [106] performed a priority-driven search, which biases the correspondence search towards correspondences between features with smaller descriptor distances. As their method focuses on rigid registration, descriptor distances are a highly reliable measure, as descriptors at the same locations on two different rigid structures are very close to each other. Zhang et al. [285] presented a powerful method for matching non-isometric surfaces, where the set of correspondences that minimizes a deformation error is selected in a greedy optimization. However, the computation of the deformation error is very inefficient, as a large overdetermined linear system, representing the deformation for the entire surface, must be solved, in a least-squares sense. Au et al. [20] computes votes for plausible correspondences when aligning skeletons of different surfaces in all possible ways.

A common optimization technique in surface matching is known as *random sample consensus*, or RANSAC [99]. RANSAC follows the approach of iteratively selecting a given number of random samples from the correspondence search-space, computing a model (transformation) that maps these samples, and verifying how many other samples in the space also fit (consent) in the computed model. As random samples are selected in every iteration, the algorithm is non-deterministic, and produces a reasonable result only with a certain probability, which increases with increasing number of iterations. RANSAC is also known as a *voting technique*, as other samples “vote” for a given model. Aiger et al. [2] select random sets of four congruent points, aligns them rigidly, and counts the number of points that are also aligned, *i.e.*, whose distance is smaller than a given threshold. Tevs et al. [255] employed a RANSAC loop for minimizing a QAP error, but biased the randomly sampling towards sample sets with higher probability of being correct correspondences. Lipman and Funkhouser [173] projected isometric surfaces onto a 2D canonical domain, and aligned them rigidly in this domain by computing a transformation between three point correspondences selected in a RANSAC-loop, and then casted votes for further correspondences based on the distance of closely aligned points.

Projecting the surfaces onto a canonical domain, where distances can be measured as Euclidean distances instead of geodesic distances, allows the matching between them to be performed rigidly. In this case, the alignment can be performed by any robust and efficient method for rigid registration (e.g. [115, 2]). It also profits from a distance metric that can be efficiently computed - the Euclidean distance - and from reliable descriptor distances. In surface matching, popular techniques for embedding surfaces in common spaces are the *multidimensional scaling* (MDS) [39], which embeds the surfaces in a common \mathbb{R}^n space, and the *generalized MDS* [44], which embeds one surface into the space spanned by the other, thus eliminating distortions that may arise from a \mathbb{R}^n embedding. Unfortunately, these methods share a drawback: They require the repeated computation of geodesic distances for obtaining the embedding, which is not efficient. In the work of Lipman and Funkhouser [173], a more efficient technique for flattening was employed, based on the works of Pinkall et al. [209], Polthier [210].

Another common technique for solving QAPs is *relaxation*. In this approach, the binary constraints are relaxed to a fuzzy domain, posing the the problem as a convex optimization problem. In surface matching, the relaxed QAP problems are usually cast as a *graph labeling problem*, where a label (correspondence) is searched for each point on the source surface (see [110] for more details on graph labeling). Popular algorithms for solving the graph labeling problem are the *maximum-flow/minimum-cut* algorithms [43, 157, 259], which poses the search-space as a system of pipes, with widths given by similarity values. These algorithms search for sub-systems with maximum flow for determining correct labeling. Zeng et al. [281], Wang et al. [263] solved the surface matching problem as a graph labeling problem. Zeng et al. [281] employed a third order error metric (Eq. 3.8 on page 29) for computing a deformation error based on flattening, as presented by Lipman and Funkhouser [173]. In order to formulate the problem as a QAP, they considered the observation that any relaxed high-order term can be reduced to quadratic terms [40, 138] and solved it as a set of QAPs. Zeng et al. [282] presented a more efficient method for solving the high order error terms, using a *Markov random field* optimization algorithm [270], thus casting the problem as a linear program. Relaxation can also be used to minimize LAPs [94], and has been employed in surface matching by Windheuser et al. [271]. Here, the authors minimize an linear error metric based on stretching and bending energy. The solution of surface matching problems by means of relaxation and labeling algorithms is usually very computationally

expensive, and the selection of very few features is therefore mostly a requirement.

Note that, similar to LAP, the direct minimization of the QAP error is global, and does not account for partially overlapping data. If the overlapping region is sufficiently large, maximum-flow/minimum-cut algorithms for graph labeling solve this problem by incorporating a label for outliers.

3.4.2 Fine-scale optimization

Fine-scale optimization procedures are used for accounting for small misalignments and for deformations that occur in local scales. They usually assume that the surfaces have been roughly aligned or that an initial set of correspondences is given. An important class of fine-scale optimization algorithms is the *point-based iterative optimization* (Sec. 3.4.2.1), which contains the *iterative closest point* (ICP) algorithm [34] and its variants.

One of the most well known classes for optimization of local scale deformations is known as *as-rigid-as-possible* (ARAP) [14, 247, 137, 239, 202], which deforms the source surface towards a set of few correspondences (anchors) by computing a fit that preserves, as much as possible, some surface properties (e.g. normal orientations [174, 159], or face Laplacians [287]). ARAP registration is usually obtained by minimizing a large overdetermined linear system of equations in a least-squares sense.

Given a coarse set of previously known correspondences, Tevs et al. [255], Wang et al. [263], Sahillioğlu and Yemez [223] iteratively solved local QAPs around the known correspondences, until a correspondence for every point was found, resulting in a dense set of correspondences. Zeng et al. [281] employed the same strategy (Fig. 15 on the next page), but minimized instead a high-order error metric (Eq. 3.8 on page 29). Raviv et al. [213] incremented the initial sparse correspondences by a set of isometrically consistent correspondences, selecting the next correspondence candidates by a *farthest point sampling* strategy [132]. Sharma et al. [233] increments the initial correspondences set by pairs of points with consistent heat propagation, using an *expectation maximization* approach [186].

Eckstein et al. [93] fitted two roughly aligned surfaces by defining a differentiable global error metric between them (pseudo-Hausdorff distance, see Sec. 3.3.2), and solved the problem by a gradient descent approach.

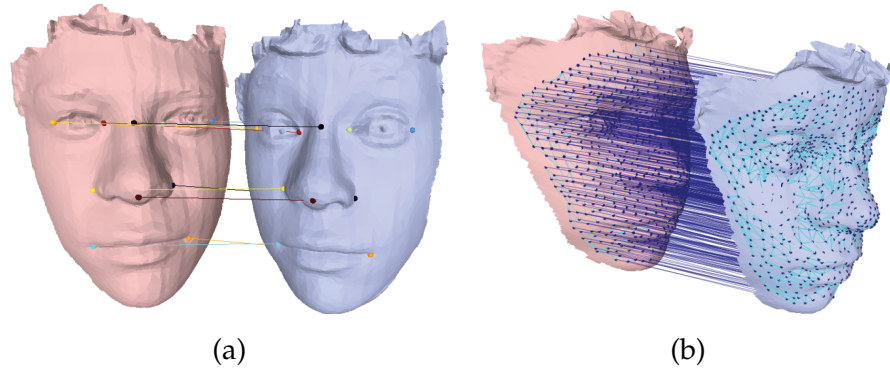


Figure 15: A set of initial sparse correspondences (a) and its expansion in a dense correspondence set (b) (image found in [281]).

3.4.2.1 Point-based iterative optimization

Point-based iterative optimization is the most well-known class for fine optimization. It is based on the *iterative closest point* (ICP) algorithm [34]. The ICP algorithm works as follows: (1) For each point on the source surface, the closest point on the target surface is found; (2) A rigid transformation that maps the pairs of closest points is computed; (3) The algorithm performs the two previous steps iteratively, until convergence is reached. It was shown that, in rigid settings, the ICP algorithm converges to at least a local minimum of the sum of squared Euclidean distances (Eq. 3.3 on page 26).

There are several variants of the ICP algorithm, dealing with: Better selection of correspondences [146, 124, 16]; Weighting of correspondences [220, 146]; Robustness to the influence of outliers [60, 140]; Adequacy of the error metric [205, 145, 95]; Non-rigid deformations [15, 68, 192].

Focusing on intra-operative registration, Cash et al. [52] presented an ICP variant that identifies and aligns minimally deformed regions of the surfaces. A non-rigid deformation based on a *finite element model* formulation is subsequently computed. Clements et al. [65, 66] presented some efforts to increase the robustness of this ICP variant, by incorporating measures of saliency.

For more details on variants of the ICP algorithm, we kindly refer the reader to [220, 200].

3.5 TRANSFORMATION

In surface matching, authors usually consider three classes of transformation: *rigid*, *affine*, and *non-rigid*.

Rigid transformations are composed of rotations and translations. The shape of the object cannot be altered by means of rigid transformations. Given a set of correspondences, the computation of a rigid transformation that aligns these correspondences, such that the sum of their squared Euclidean distance is minimal, is usually performed by solving an overdetermined linear system, in a least-square sense [134].

Affine transformations are a generalization of the rigid transformations, removing some of the constraints of the rigid transform operator. Although not all affine transformations are angle preserving, lines remain parallel [104]. A specific affine transformation of interest for surface matching is scaling, where the dimensions of the object change, but angles are preserved. Given a set of correspondences, an affine transformation between them can be computed according to the method presented by Feldmar and Ayache [97], which also minimizes the sum of squared distances.

Non-rigid transformations, also known as *free-form* transformations, are subdivided in two classes: isometric and non-isometric. Isometric transformations preserve distances (geodesic distances), while non-isometric do not. Non-rigid transformations are usually represented by *piecewise polynomials* [104], which model transformations for local surface patches while maintaining first or second order continuity between patches. This ensures a smooth mapping between the surfaces. Most well-known piecewise polynomials for transformations are the *thin-plate splines* [38] and the *B-splines* [87].

For more details on transformations, we kindly refer the reader to [104, 22].

3.6 SUMMARY

Tab. 1 provides an overview of the most recent and influential publications in surface matching. Despite of the many advances in the field, we found that most of the rough-scale optimization methods for non-rigid surface matching rely on the solution of *quadratic assignment problems* (QAPs). In the context of surface matching for intra-operative registration, this implies two main drawbacks: First, QAPs uses the difference of distances between assigned point pairs as a regularization term. As shown in Sec. 3.1, these geodesic distance profiles are not sufficiently discriminative for nearly planar surfaces, such as the ones acquired in intra-operative environments, as they are for more complex shapes, such as a human form. Second, the direct solution of a QAP using a minimization technique, such as relaxation (Sec. 3.4.1), makes the registration of partially overlapping surfaces

harder, as QAP would provides a global set of correspondences that minimizes the error.

Authors solved the first problem by the incorporation of higher order and more complex error metrics (e.g. [285, 281]). The second drawback was solved by an enumeration of the search space, for the desired amount of correspondences, and by solving the minimization problem with constrained greedy optimization procedures or voting schemes (e.g. [255, 223]). However, in these cases, the computational cost becomes so expensive that the consistent selection of features on both surfaces becomes a necessity, *i.e.*, a very few representative and unambiguous features that can be consistently selected from the same locations on both surfaces need to be selected. As discussed in Sec. 3.2, consistent feature selection on pre- and intra-operatively acquired surfaces is an issue, as these surfaces are nearly flat, and many distortions and deformations occur between them.

In conclusion, while suitable fine-scale optimization methods for intra-operative registration already exist [128, 21, 52, 65, 111, 51, 212, 31, 66, 91], the automatic alignment of surfaces of different modalities acquired pre- and intra-operatively by rough-scale optimization methods remains a challenge.

AUTHORS	DESCRIPTOR	ROUGH/FINE SCALE	OPTIMIZATION	ERROR METRIC	GLOBAL/PARTIAL MATCHING	RIGID/NON-RIGID TRANSFORMATION	FEATURE SELECTION
Gelfand et al. [115]	Integral volume descriptor	Rough	Branch-and-bound	$E_{\text{IRMS}}(\cdot)$	Partial	Rigid	Yes (on the source surface only)
Cash et al. [52]	-	Fine	Point-based iterative	Modified $E_{\text{ICP}}(\cdot)$	Partial	Non-isometric	No
Funkhouser and Shilane [106]	Spherical harmonics	Rough	Greedy	QAP	Partial	Rigid	Yes
Eckstein et al. [93]	-	Fine	Gradient descent	Pseudo-Hausdorff distance	Partial	Isometric	No
Aiger et al. [2]	-	Rough	Voting	Size of intersection	Partial	Rigid	No
Zhang et al. [285]	Gatzke et al. [113]	Both	Greedy	ARAP deformation error	Partial	Non-isometric	Yes
Tevs et al. [255]	Histogram of mean curvature	Both	RANSAC	QAP	Partial	Isometric	Yes
Lipman and Funkhouser [173]	-	Rough	Voting	Embedding deformation error	Partial	Isometric	No
Zeng et al. [281]	-	Both	Relaxation	$E_{2D_fit}(\cdot)$	Global	Non-isometric	Yes
Wang et al. [263]	HKS	Both	Relaxation	QAP	Global	Isometric	Yes
Au et al. [20]	Skeleton	Rough	Voting	Skeleton dissimilarity	Global	Non-isometric	No
Dubrovina and Kimmel [89]	Manifold harmonics	Rough	Relaxation	QAP	Global	Isometric	Yes
Sahillioglu and Yemez [223]	-	Both	Greedy	QAP	Partial	Isometric	Yes
Windheuser et al. [271]	-	Rough	Relaxation	Physics-based deformation error	Global	Isometric	Yes
Raviv et al. [213]	HKS	Both	Relaxation	QAP	Global	Isometric	Yes
Sharma et al. [233]	HKS	Fine	Expectation maximization	QAP	Global	Isometric	No

Table 1: Overview of the state-of-the-art surface matching approaches. Refer to Secs. 3.1, 3.3, 3.4 and 3.5 for a description of the methods and of the abbreviations.

SURFACE REPRESENTATION

Without geometry, life is pointless.

— Anonymous

When David Marr at MIT moved into computer vision, he generated a lot of excitement, but he hit up against the problem of knowledge representation.

He had no good representations for knowledge in his vision systems.

— Marvin Minsky

The representation of objects using boundary representation data structures is advantageous for several reasons: Accelerated retrieval of adjacency and incidence information, efficient object traversal, maintenance of topological consistency during manipulation, etc. Although very efficient and well-known data structures exist for the representation of *2-manifold* objects [24, 25, 26, 183, 122], most data structures for *non-manifold* representation [268, 123, 172, 129, 162, 216, 74, 76, 37, 235] are either verbose and inefficient [75], do not provide an operator set for topological manipulation or provide operator sets that are not intuitive.

A *2-manifold* object (mesh) is a subset of the Euclidean space, where the neighborhood of each point is homeomorphic to an open disc. Every other object that does not fulfill this property is called a *non-manifold* [74]. Non-manifold objects are very common in practical applications because of their superior expressive power. Motivations for using representations of non-manifold objects have been pointed out by several authors [268, 217, 123, 82, 56, 216]. For example: Boolean operators are closed in the non-manifold domain; sweeping or offset operations may generate parts of different dimensionalities; non-manifold topologies are required in different product development phases, such as conceptual design, analysis or manufacturing [77]; and simplification methods that employ vertex pair contraction generate non-manifold objects [112].

One approach for the representation of non-manifold objects consists of subdividing the object into simpler *2-manifold* parts and in connecting these parts via *2-pseudomanifold* components [75]. A *2-pseudomanifold* is a relaxation of the *2-manifold* definition, allowing a point to be homeomorphic not only to a single open disk, but also

to pinched open disks, which are topological spaces obtained by the identification of the centers of multiple open disks [121]. Employing the 2-pseudomanifold approach, data structures for the representation of non-manifolds can benefit from many of the advantages of 2-manifold representation data structures.

The *Doubly Linked Face List* (DLFL) is another data structure for mesh representation. The DLFL always ensures topological 2-manifold consistency and uses a minimal amount of computer memory [58, 3]. It is the implementation of a *graph rotation system*, which was shown to give a unique orientable 2-manifold. A graph rotation system consists of a list of rotations, which are cyclic permutations of edges incident to a particular vertex. For each vertex, a rotation is contained in the system.

Besides topological consistency, the DLFL supports efficient topology manipulation through its operators. Since the DLFL has a graph-based concept, its operators are derived from topological graph operations, which are extremely simple, and similar to Euler operators [183]. In fact, there are only four operators for the manipulation of its topology, and these are able to create any 2-manifold [9]. These operators are responsible for the creation and deletion of vertices, and for the insertion and removal of edges. The applicability of the DLFL and its operators was demonstrated in the construction of regular meshes [4, 5], handles [241] and rind [10] modeling, subdivision schemes [7, 8, 6] and remeshing [11, 12].

Though very intuitive, the DLFL's operators can easily lead to unexpected results, modifying the structure in unexpected ways, and degenerating faces in order to maintain topological consistency. The problem arises through the use of the `INSERTEDGE` operator. This operator inserts an edge between two face corners - representations of edge sides starting at a particular vertex in a particular face - instead of between two vertices directly. Since vertices are usually found in more than one face and many edge sides start at it, choosing the appropriate corners in order to modify the structure in the desired way may be complicated. In the development of *TopMod3D* [13], a software for mesh modeling, the user interface attempts to avoid this ambiguity by requiring the user to first select a face and then a corner. This strategy might be successful if the user is familiar with the DLFL and knows how to choose the appropriate corners. However, for the automatic generation of meshes, *e.g.* using the *Marching Cubes* algorithm [175], and for software tools in which the interaction with the data structures should be transparent, *e.g.* surgery simulation softwares, this ambiguity remains a critical issue. Furthermore, the DLFL is unable to represent boundaries or 2-pseudomanifolds. In

order to achieve 2-pseudomanifolds representations, one must recur to 2-manifold interpretations of non-manifolds [240], thereby degenerating the structure of the mesh.

In order to overcome these problems and to expand the representation capabilities of the DLFL, we present the *Extended Doubly Linked Face List* (XDLFL). The XDLFL extends the DLFL for the representation and manipulation of *2-manifolds with boundaries*¹ (Sec. 4.2.1) and for *2-pseudomanifolds* (Sec. 4.2.2) with minimal modifications to its internal structure and operators, thus avoiding substantial increases in memory usage and complexity, while profiting from its efficiency and simplicity. Some issues related to the implementation of the extensions in the DLFL and to the memory usage of the XDLFL are discussed in Sec. 4.2.4.

These extensions allow the representation and manipulation of meshes with boundaries and the exact representation of certain non-manifold cases, such as 2-manifold shells connected by single vertices or edges, or pinched surfaces, without degenerations in the structure. Proper representation of boundaries is very important in applications such as surgery simulation softwares, where meshes are cut and subsequently closed again. Appending two objects to each other may be performed very efficiently with 2-pseudomanifolds, without causing modifications to the objects' structure. The proposed extensions also allow the representation (inexactly at the boundaries) of non-orientable meshes, such as the Möbius band and any other mesh containing a subset homeomorphic to it.

In order to deal with the ambiguity of the INSERTEDGE operator, two further operators are introduced: The CREATEFACE (Sec. 4.3.1) and REMOVEFACE (Sec. 4.3.2) operators. The CREATEFACE operator creates a new face from the specification of its boundary walk. This releases the user from the obligation of having to insert every edge individually and having to pick the corners that will lead to the desired modifications. It ensures that no other face in the mesh will be modified (except the boundary faces, which will be introduced later). In addition, the orientation of the boundary walk is adjusted in order to fit into the mesh. This operator correctly manipulates the boundaries in order to append new faces to the mesh and resorts to 2-pseudomanifolds when the mesh should be expanded in places where there are no boundaries. The REMOVEFACE operator performs the inverse operation.

Though the CREATEFACE operator may be used for the initialization of the XDLFL from an arbitrary polygon set, it may manipulate

¹ A *2-manifold with boundaries* is a subset of the Euclidean space where every point is homeomorphic to an open disk or a half-open disk.

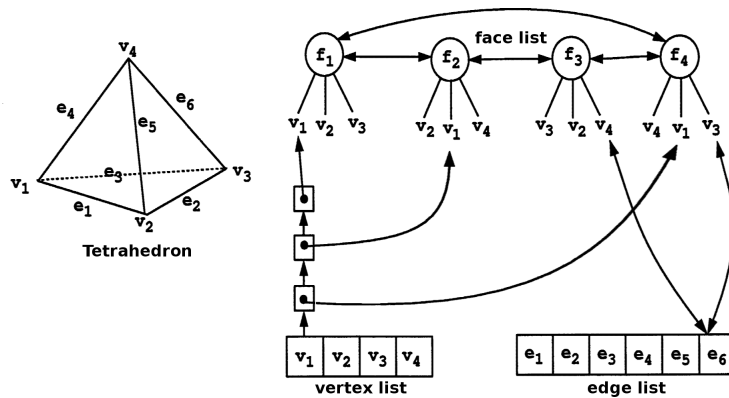


Figure 16: The internal organization of the DLFL during the representation of a tetrahedron. The DLFL is composed of a doubly linked face list, a vertex list and an edge list. As an example, the links of vertex v_1 and edge e_6 are shown. This figure is based on the illustration found in the work of AKLEMAN AND CHEN [3].

the boundaries in an unexpected way when one tries to create a 2-manifold mesh using a face creation order that generates 2-manifold interpretations of non-manifolds (Sec. 4.3.1.4). For the purpose of avoiding these misinterpretations, a safe procedure for the initialization of the XDLFL from an arbitrary polygon set is presented (Sec. 4.4).

4.1 THE doubly linked face list

The Doubly Linked Face List (DLFL) is a structure for mesh representation, which always ensures 2-manifold topology. It is composed of a face list, an edge list, and a vertex list (Fig. 16). Their nodes are referred to as *face*, *edge* and *vertex*, respectively. Each face is a 2-3 tree containing the vertices from its boundary walk. The nodes in the 2-3 trees are referred to as *corners*. Each corner represents an edge side starting at a particular vertex and has a pointer to the edge it is part of. Using the FACETRACE [3] operator, the boundary walk of a particular face is obtained. Each vertex is a list of pointers to the edge sides (corners) that start at it. Using the VERTEXTRACE [3] operator, the rotation in circular order of the edges incident to a particular vertex is obtained. Each edge is a pair of pointers to the corners that represent both its edge sides. Furthermore, the face list is a circular doubly linked list, while the vertex and edge lists are arrays.

As can be seen, the corners play an important role in the structure. They act as a bond between the elements in the DLFL. In addition, they constitute both the vertices' rotations (representing the

edge sides incident to the vertices) and the faces' boundary walks (representing the oriented edge sides of which it consists).

There are four operators for the topological manipulation of the structure:

- `CREATEVERTEX` and `DELETEVERTEX` [9], responsible for creating and deleting vertices in or from the vertex list. They are also responsible for creating and deleting new faces, since for every new vertex, a new face containing only the new vertex in its boundary walk is created (a face containing a single corner, representing a point-sphere);
- `INSERTEDGE` and `DELETEEDGE` [9], responsible for inserting and deleting edges between corners, and for modifying the boundary walk of faces. They also manipulate the vertices' rotations.

The operation used to insert an edge between two corners will be described in more details in order to show how the DLFL is manipulated and how the use of the `INSERTEDGE` operator can be ambiguous. However, we first need to explain how a corner is represented. In the work of Akleman and Chen [3], a corner is represented as a tuple composed of two consecutive edges in a face along with the vertex between them (which corresponds to two consecutive edges in the rotation of this vertex), as for example $c_1 = (e_3, e_1, v_1)$. As can be seen in Fig. 16, it is clear that this corner belongs to face f_1 , since it is the face in which both edges e_3 and e_1 have a side. In this work, we opt for a more explicit and clearer representation. It consists of a tuple composed of a vertex, the face to which the corner belongs, and the edge constituted by the side starting at that vertex. In our scheme, the same corner c_1 is represented as (v_1, f_1, e_1) . Using this representation, one can easily identify the face to which this corner belongs and which edge it is part of (as a side).

There are two further considerations regarding the corner representation scheme adopted here. Let us suppose the above mentioned corner, c_1 , and that the previous edge in the rotation of v_1 has to be found. In the scheme adopted by Akleman and Chen, this information is explicit in the corner's representation (in this case, e_3). In our scheme, it is the edge pointed by the previous corner in the face's boundary walk (in this case, the face is f_1 and the previous corner is $c_3 = (v_3, f_1, e_3)$). The second consideration regards the creation of a new vertex, which, as said before, forms a point-sphere: A new face containing a single vertex in its boundary walk (one single corner). Since there are no edges, the corner is simply represented by the vertex and the face it belongs to [242].

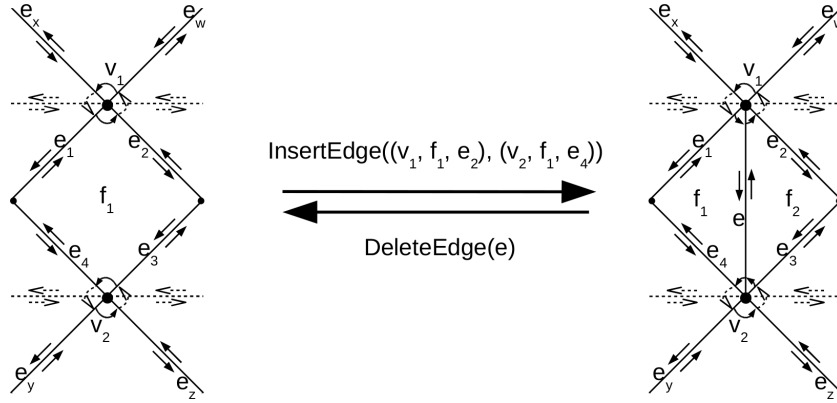


Figure 17: The INSERTEDGE and DELETEEDGE operations. An edge is inserted between the corners represented by vertex v_1 , in face f_1 at the extremity of edge e_2 , and by vertex v_2 , in face f_1 at the extremity of edge e_4 , causing face f_1 to be divided in two. The removal of the new edge (e) causes both faces to be merged into a single face again.

Supposing two corners $c_1 = (v_1, f_1, e_1)$ and $c_2 = (v_2, f_2, e_4)$, the INSERTEDGE(c_1, c_2) operator inserts a new edge e before e_1 in the rotation of v_1 , and before e_4 in the rotation of v_2 . The consistency of the faces' boundary walks will be maintained as well, splitting the boundary walk in two if the corners belong to the same face ($f_1 = f_2$), or joining them otherwise.

Fig. 17 shows the effects of applying the INSERTEDGE operator (and its inverse, the DELETEEDGE operator) to the previously defined corners c_1 and c_2 , supposing they belong to the same face ($f_1 = f_2$). The rotations of vertices v_1 and v_2 , $R(v_1) = (e_x, \dots, e_1, e_2, \dots, e_w)$ and $R(v_2) = (e_z, \dots, e_3, e_4, \dots, e_y)$, are modified to $R(v_1) = (e_x, \dots, e_1, e, e_2, \dots, e_w)$ and $R(v_2) = (e_z, \dots, e_3, e, e_4, \dots, e_y)$ respectively, in order to include the new edge e . In addition, the face's boundary walk is divided in two in order to maintain consistency.

Using the INSERTEDGE operator, we will now show how to build the tetrahedron in Fig. 16. First, all required vertices are created. This is done using the CREATEVERTEX operator, which creates a new entry in the vertex list and a new face (a point-sphere) for each vertex. After the insertion of all four vertices, the DLFL contains four vertices in the vertex list and four faces in the face list (the edge list is still empty). The edges are inserted using the newly created corners as parameters for the INSERTEDGE operator (Fig. 18). In the special case where an edge insertion occurs between two point-spheres, they are both merged into a single face [9].

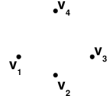
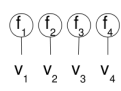
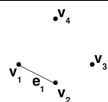
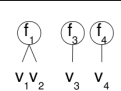
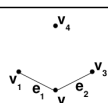
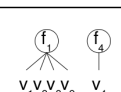
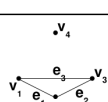
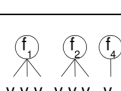
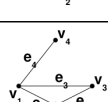
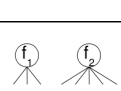
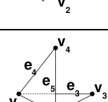
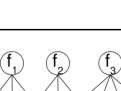
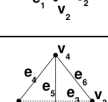
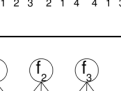
Operation	Topology	Faces	Rotations
			$R(v_1)=\{\emptyset\}$ $R(v_2)=\{\emptyset\}$ $R(v_3)=\{\emptyset\}$ $R(v_4)=\{\emptyset\}$
1. INSERTEDGE($(v_1, f_1), (v_2, f_2)$)			$R(v_1)=\{e_1\}$ $R(v_2)=\{e_1\}$ $R(v_3)=\{\emptyset\}$ $R(v_4)=\{\emptyset\}$
2. INSERTEDGE($(v_2, f_1, e_1), (v_3, f_3)$)			$R(v_1)=\{e_1\}$ $R(v_2)=\{e_1, e_2\}$ $R(v_3)=\{e_2\}$ $R(v_4)=\{\emptyset\}$
3. INSERTEDGE($(v_3, f_1, e_2), (v_1, f_1, e_1)$)			$R(v_1)=\{e_1, e_3\}$ $R(v_2)=\{e_1, e_2\}$ $R(v_3)=\{e_2, e_3\}$ $R(v_4)=\{\emptyset\}$
4. INSERTEDGE($(v_1, f_2, e_3), (v_4, f_4)$)			$R(v_1)=\{e_1, e_3, e_4\}$ $R(v_2)=\{e_1, e_2\}$ $R(v_3)=\{e_2, e_3\}$ $R(v_4)=\{e_4\}$
5. INSERTEDGE($(v_4, f_2, e_4), (v_2, f_2, e_1)$)			$R(v_1)=\{e_1, e_3, e_4\}$ $R(v_2)=\{e_1, e_2, e_5\}$ $R(v_3)=\{e_2, e_3\}$ $R(v_4)=\{e_4, e_5\}$
6. INSERTEDGE($(v_4, f_3, e_4), (v_3, f_3, e_2)$)			$R(v_1)=\{e_1, e_3, e_4\}$ $R(v_2)=\{e_1, e_2, e_5\}$ $R(v_3)=\{e_2, e_3, e_6\}$ $R(v_4)=\{e_4, e_5, e_6\}$

Figure 18: Operation sequence for the creation of the tetrahedron shown in Fig. 16. Here we only show the vertices in the faces instead of the corners in order to keep the figure simple.

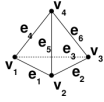

Operation	Topology	Faces	Rotations
6. INSERTEDGE($(v_4, f_2, e_5), (v_3, f_3, e_2)$)			$R(v_1)=(e_1, e_4, e_3)$ $R(v_2)=(e_1, e_2, e_3)$ $R(v_3)=(e_2, e_3, e_6)$ $R(v_4)=(e_4, e_5, e_6)$

Figure 19: Step 6 of Fig. 18 using a different corner for the edge insertion.

Two operators, the CREATEVERTEX and the INSERTEDGE, suffice for the creation of an exact representation of the tetrahedron (four vertices, six edges and four faces). This exact representation, as shown in Fig. 18, was only possible because we were able to correctly manipulate the vertices' rotations. This means that the edge insertions occurred between the corners that led to the desired tetrahedral topology.

In the case of the last edge insertion in Fig. 18, if corner (v_4, f_3, e_4) had been selected instead of corner (v_4, f_2, e_5) , the results would have been different (Fig. 19). This is the case because these two corners, although representing edge sides starting at the same vertex, represent different positions in the vertex's rotation (and belong to different faces). Inserting the new edge at another position in the rotation of vertex v_4 would result in a face containing three closed loops in its boundary walk (the boundary walk passes three times over v_4).

Such topological degenerations are very likely to occur in meshes represented by DLFLs, because the user is forced to choose between corners instead of vertices when inserting new edges. Having to choose between corners gives the user many options for inserting an edge between two particular vertices, while the selection of a single incorrect corner combination will lead to undesired topological modifications. Furthermore, the greater the number of edges incident to a particular vertex, the greater the number of corners, making it more difficult to choose as the mesh grows.

Other degenerations occur when trying to represent non-manifolds through the DLFL. Since the DLFL is not able to represent such topologies, a 2-manifold interpretation of the non-manifold takes place [242]. Such cases are depicted in more detail in Section 4.2.2.

4.2 EXTENSIONS TO THE *doubly linked face list*

We now show how to extend the DLFL for the representation of boundaries (Sec. 4.2.1) and 2-pseudomanifolds (Sec. 4.2.2), and present the definition of the new data structure achieved through them (Sec. 4.2.3): The *Extended Doubly Linked Face List* (XDLFL). These

extensions and the new operators accompanying them will be used later for the implementation of the `CREATEFACE` and `REMOVEFACE` operators. They will be responsible for the correct manipulation of the extensions, making the handling of the whole DLFL more intuitive for users.

4.2.1 *2-Manifolds with boundaries*

Because of the fact that the DLFL always maintains the 2-manifold property, for any new edge created, there will always be some face containing the same edge in the opposite direction in order to maintain the surface closed and the mesh orientable. For example, in Fig. 18 (step two), a new copy of v_2 was created in the same face after the insertion of an edge between v_2 and v_3 in order to have the same edge in the opposite direction and to maintain the circularity of the face's boundary walk. For the same purpose, an entirely new face containing the edges in the opposite direction was created after the face was closed through the insertion of an edge between v_3 and v_1 (step three). This new face was automatically created by the `INSERTEDGE` operator in order to maintain topological consistency of the surface's boundaries. In step five, one can see how this same face was expanded in order to include the newly created boundaries.

Since the DLFL automatically adds faces in order to maintain topological consistency, if one wants to expand the mesh without the intentionally created faces being modified, the correct corners to be chosen are the ones contained in the automatically created faces. The DLFL operators only modify the faces containing the operand corners. In Fig. 18, this rule was followed in order to obtain the tetrahedron, while in Fig. 19, a corner contained in an intentionally created face was chosen in the last operation, thus modifying it.

The faces added by the DLFL for topological consistency maintenance are called boundary faces because they are inserted in the boundaries of the mesh. It was also shown that these are the correct faces to modify if the intentionally added ones are to remain intact. The problem is that the DLFL does not keep track of those faces in order to allow the user to directly identify them. We adapted the DLFL to be able to keep track of the boundary faces. This was done by labeling those faces and adding them to a separate face list. In the following we show how the DLFL was modified in order to allow labeling of faces.

As previously stated, the DLFL is extended to include two distinct face lists: One for regular faces (simply referred to as "face list") and one for boundary faces (referred to as "boundary list"). An operator,

the `IsBOUNDARY` operator (Alg. 4.1), is responsible for labeling and unlabeling a particular face, moving it from the face list to the boundary list and *vice versa*.

Algorithm 4.1 The `IsBOUNDARY` operator.

```

1 IsBoundary(Face f, Boolean to_label)

3 if to_label
    if f is not labeled then
5     Label f
    Move f to the boundary list;
7 else if f is labeled then
    Unlabel f;
9     Move f to the face list;

```

The `INSERTEDGE` and the `DELETEEDGE` operators also have to be adapted for the new extension, since they insert and remove faces to and from the face list. We have adopted the following convention: Any time one of these operations involves a corner that belongs to a boundary face, the newly split faces or the resultant merged face will be labeled a boundary. This means that the old DLFL operators do not make any assumptions about the needs of a label (Fig. 20) because they do not have enough contextual information to make such a decision. They do not know what the final face topology should look like, because they operate on a single edge at time. It is the responsibility of the `CREATEFACE` operator (Sec. 4.3.1) to correctly manage the boundaries and labels.

The tetrahedron in Fig. 18 could be created using the extension for boundaries in order to ensure that the correct topology would be achieved. Implicitly, we always chose the corners belonging to the boundary faces in order to achieve correct representation. Four extra steps must be added to make use of the boundary representations. After closing the first face (f_1), the back face (f_2) should be labeled a boundary. One can see that only corners belonging to the boundary face are chosen in the next steps. After step five, the label has to be removed from f_2 , leaving it only for f_3 . At the end, after step six, the labels from f_3 and f_4 must be removed.

Additionally, the extension presented here can be used for boundary identification for methods that employ special treatments for the boundaries of the mesh, as do some subdivision methods [244] for example (Fig. 21).

It is also important to notice that vertices having a complete edge rotation will not belong to any boundary, for example vertex v_5 (Fig.

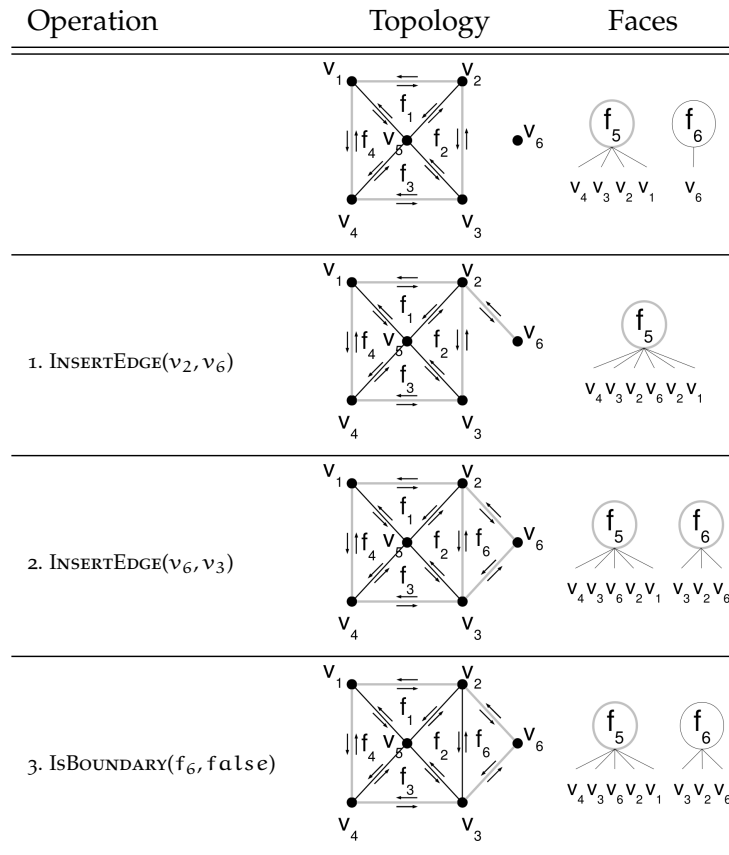


Figure 20: Extending the boundaries (shown in gray) with a new face using the extension for boundary representation. In the end, the boundary label has to be manually removed from the new face, because the INSERTEDGE operator does not make any assumptions as to whether a face is a boundary or not. For the sake of simplicity, only faces f_5 and f_6 are shown explicitly. For the same reasons, instead of explicitly writing the corners as arguments for the operations, only the vertices are indicated.

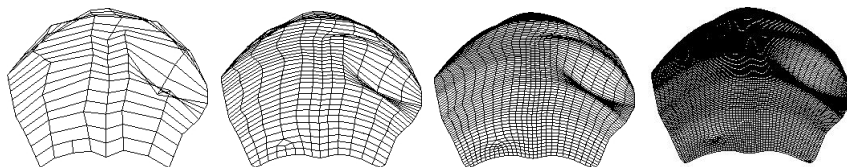


Figure 21: Subdivision scheme where the mesh converges to b-spline surfaces, except at the boundaries, where they converge to b-spline curves.

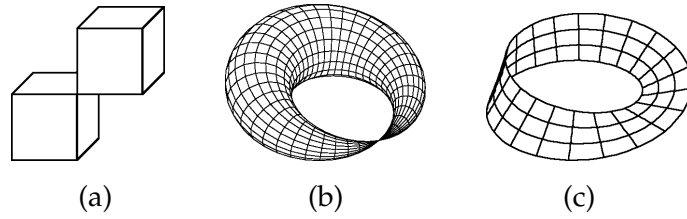


Figure 22: Examples of surfaces which can be represented through 2-pseudomanifolds: (a) Two cubes connected through a single edge; (b) A pinched torus; (c) The Möbius band.

20). Using only the `INSERTEDGE` operator, it is not possible to add a new edge in the rotation of this vertex without modifying one of the faces in which it is contained. One way to do this is to extend the DLFL for the representation of 2-pseudomanifolds.

4.2.2 2-Pseudomanifolds

A 2-pseudomanifold is a relaxation of the 2-manifold definition. A 2-manifold is a topological space in which each vertex has a neighborhood that is homeomorphic to an open disk. This means that a mesh has a 2-manifold topology when each vertex in the mesh has a unique rotation of edges. However, one can imagine two distinct topological vertices placed in the same geometrical coordinate. The result is a geometric point with two edge rotations, while each topological vertex still has only one. If those vertices can in some way be identified to each other, a pinched open disk is obtained, which is a topological space obtained from the identification of the centers of multiple open disks [121]. Thus, a 2-pseudomanifold is a topological space in which each vertex has a neighborhood which is homeomorphic to an open disk or to a pinched open disk. The extension for 2-pseudomanifolds allows the DLFL to exactly represent connected 2-manifold shells (Fig. 22a) and pinched surfaces (Fig. 22b), and to inexactly (at the boundaries) represent non-orientable meshes (Fig. 22c).

In order to obtain the 2-pseudomanifold extension for the DLFL, the structure is modified to be capable of representing pinched open disks. For this purpose, each vertex is no longer considered the center of a single rotation (open disk), but the center of multiple rotations. Thus, instead of being a list of corners, which represent the edge sides in its rotation, each vertex now becomes a list of rotations. Each rotation represents a topologically distinct open disc, centered on a particular vertex, and is a list of corners, which represent the edge

sides. In other words, the rotation assumes the functions of the original DLFL's vertex, while a vertex stands as the identified center of multiple rotations. This means that the corners will no longer point to vertices, but to the rotation they belong to, and each rotation will point to its center.

The behavior of `CREATEVERTEX` and `DELETEVERTEX` is modified: Their unique responsibility in each case is to add a new vertex to the vertex list and to remove a particular vertex from it (if it does not have any rotations). In addition, two new operators for creating and deleting rotations are introduced:

- `CREATEROTATION`: Creates, for a particular vertex, a new rotation containing a single corner, and a new face containing the same corner (Alg. 4.2). It returns the newly created corner.
- `DELETEROTATION`: Deletes a particular rotation if it has only one corner and if the face that contains the same corner is a point-sphere (Alg. 4.3).

Algorithm 4.2 The `CREATEROTATION` operator.

```

1 CreateRotation(Vertex v)
3 r ← new rotation in v
  f ← new face in the face list
5 c ← new corner into r and f
  Return c

```

Algorithm 4.3 The `DELETEROTATION` operator.

```

DeleteRotation(Rotation r)
2
  if valence(r) = 1 then
4   c ← corner in r
     if valence(c.face) = 1 then
6     Delete c.face, c and r
       Return true
8 Return false

```

Supposing one wants to create two tetrahedrons connected by a single vertex, this can be easily done using the 2-pseudomanifold operators. First, all vertices are created (with the `CREATEVERTEX` operator) and, for each vertex, a new rotation is created (with the `CREATEROTATION` operator). The first tetrahedron is constructed as shown in Fig.

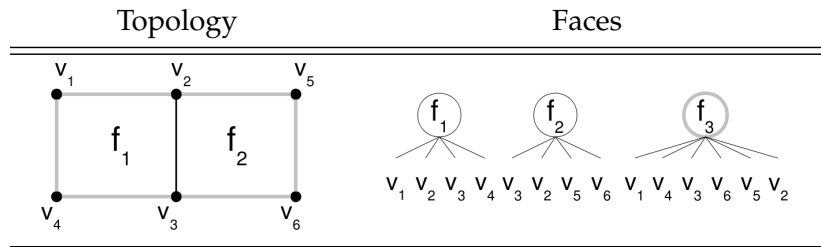


Figure 23: A two-faced strip with a boundary (in gray).

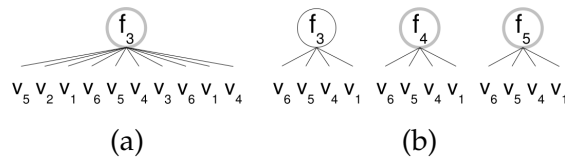


Figure 24: A Möbius strip representation (faces in gray represent boundaries): (a) With a regular DLFL; (b) With a DLFL extended for the representation of 2-pseudomanifolds.

18. A new rotation is created for the vertex that is intended to connect both tetrahedrons. The second tetrahedron is constructed in the same way as the first one, using the corner created with the new rotation and the remaining vertices. A similar procedure can be used to create the pinched torus. For the construction of the cubes connected by an edge, it is only necessary for one of the edge extremities to receive a new rotation, since the same edge can be reinserted between an old extremity and the new corner.

The Möbius strip representation is here exemplified using a three-faced polygon. A Möbius strip is a non-orientable surface with one single boundary. In Fig. 22c it can be seen that if one picks any point on the boundary and follows it in any direction, the point picked will be reached again. Let us suppose the situation in Fig. 23, and that one wants to add a new face (v_6, v_5, v_4, v_1) , which is obtained by adding the edges (v_5, v_4) and (v_1, v_6) . Those edges are crossed, forming a Möbius strip, and the new face is non-orientable since the edge side (v_4, v_1) already exists (in face f_1). Inserting the edges directly, using the corners in the boundary face, leads to a single face containing many loops (Fig. 24a) and no separation between internal faces and boundaries. If, before inserting the edges, a new rotation for vertex v_1 is created and then the edges are inserted (using the corner contained in the new rotation created for v_1), the exact representation of the desired face is obtained (Fig. 24b). However, the representation of the boundary is inexact, since two boundary faces are obtained.

Other non-orientable surfaces, such as the *Klein bottle*, may be obtained in the same way. In this case, however, a boundary will exist, whereas the Klein bottle is a non-orientable surface without boundaries.

4.2.3 The Extended Doubly Linked Face List

Adapting the definition of the Doubly Linked Face List to include the extensions to the data structure, in the following we present the definition of the *Extended Doubly Linked Face List* (XDLFL).

DEFINITION: An *Extended Doubly Linked Face List* (XDLFL) is a 4-tuple $L = (\mathcal{F}, \mathcal{B}, \mathcal{V}, \mathcal{E})$. The *face list* \mathcal{F} is a doubly linked list in which each node is referred to as a *face* and represents the boundary walk of a particular face. The *boundary list* \mathcal{B} is also a doubly linked list in which each node is referred to as *boundary* and represents the boundary walk of a particular mesh boundary. Each element in the boundary walk of a particular face or boundary is referred to as a *corner* and represents an edge side starting at a particular vertex. The *vertex list* \mathcal{V} is a doubly linked list of vertices v in which each is a list of rotations. Each *rotation*, representing a cyclic permutation of edges incident to a particular vertex v , is a doubly linked list of pointers to the corners that represent the edge sides starting at v . The *edge list* \mathcal{E} is a doubly linked list in which each node points to the two corners that represent both edge sides of the same edge.

In addition to the modifications concerning the extensions previously shown, the definition of the XDLFL differs from the DLFL inasmuch as doubly linked lists of vertices and edges are used in the place of arrays. Using doubly linked lists allows constant time insertion and removal of vertex and edge nodes. Furthermore, in the DLFL definition, the face nodes are 2-3 trees. The XDLFL definition does not include any specification. Instead of 2-3 trees, lists could be used, which would allow valence checks and cofacial tests in constant time. Using lists for the face nodes would make the INSERTEDGE and DELETEEDGE operator to execute in linear time (it is logarithmic using 2-3 trees), which is usually not a problem, since most meshes are composed of faces with a maximum of three or four vertices in their boundary walks.

4.2.4 Implementation aspects and memory usage

In order to implement the XDLFL, some minor changes must be made to the original internal structure and operators of the DLFL. Two modifications must be made to the internal structure, as stated in the definition of the XDLFL:

- A fourth doubly-linked list, the boundary list, is added to the structure for the separation of regular faces and boundary faces;
- The vertices are no longer a list of pointers to corners but a list of pointers to rotations, which are in turn a list of pointers to corners. In the XDLFL the vertices assume the role of centers of different edge rotations.

With respect to the operators, the modifications are as follows:

- `CREATEVERTEX` and `DELETEVERTEX`: They are no longer responsible for the creation and removal of point-spheres. This role is assumed by the `CREATEROTATION` operator. In the XDLFL, these operators only insert or remove entries in the vertex list;
- `INSERTEDGE` and `REMOVEEDGE`: As these operators act directly on corners of which the concept was not modified in the XDLFL, their implementation is not changed, with the exception of a few modifications for boundary labeling consistency as mentioned in Section 4.2.1. Inserting and removing edges between boundary components or between boundary and non-boundary components is not problem either, since the boundaries are represented by boundary faces, which are regular DLFL faces.

Kettner [149] analyzed the memory usage of mesh representation data structures by computing the number of pointers required in the main component of the structures. In the case of the *Winged-Edge* [24, 25, 26], *Half-Edge* [183] and *Quad-Edge* [122] data structures (the main component is the edge), the memory usage is 8 pointers, 10 pointers and 8 pointers (plus 12 bits), respectively. In the DLFL, the main component is the corner, which binds faces, edges and vertices. For the DLFL's corner, 6 pointers are required: A double link to the face containing it, a double link to edge, the half-edge of which is represented by that corner, and a double link to the vertex it represents.

In the case of the XDLFL, the corner continues to require 6 pointers, however, it no longer points to the vertex but to the rotation containing it (see Sec. 4.2.3). Furthermore, the rotation should be doubly linked to the vertex that represents its center. Even when including

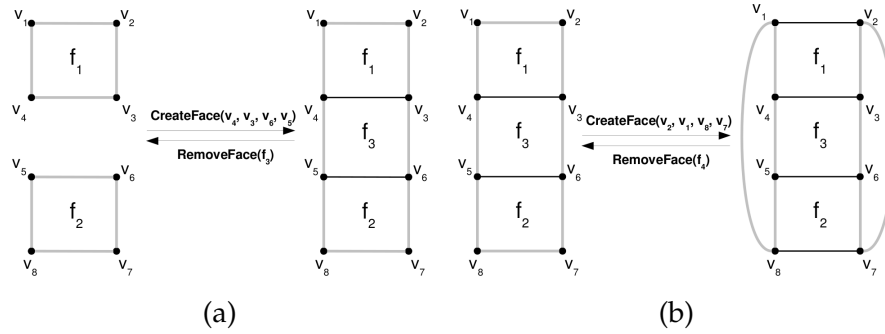


Figure 25: The boundaries after inserting and removing new faces using the `CREATEFACE` and `REMOVEFACE` operators: (a) The boundaries are merged after the insertion of a face in the middle and split after removal; (b) The boundaries are split in two after inserting a new face and merged after removal.

the extensions presented here, the XDLFL data structure continues to have only 6 pointers per corner, thus keeping memory usage smaller compared to other well-known data structures.

4.3 MANIPULATION BY DIRECT FACE CREATION AND REMOVAL

We present here two operators responsible for the direct insertion and removal of entire faces into the XDLFL: The `CREATEFACE` (Sec. 4.3.1) and `REMOVEFACE` (Sec. 4.3.2) operators. The `CREATEFACE` operator uses the extensions presented in the previous sections to expand the mesh without modifying the previously inserted faces (except for the boundaries). It correctly manipulates the boundaries by automatically labeling back faces, joining boundaries (Fig. 25a) and splitting them (Fig. 25b). In cases where the rotation of a particular vertex is completely contained in the mesh (it does not belong to a boundary), 2-pseudomanifolds are used in order to ensure that there are no modifications to the other faces. The `DELETEFACE` operator performs the inverse operation.

4.3.1 The `CREATEFACE` operator

This operator takes as arguments, in correct order, the vertices which will form the new face's boundary walk. The only precondition is that the vertices must already exist in the structure (they are created using the `CREATEVERTEX` operator). Rotations are automatically created by this operator, so there is no need for the user to manually create them.

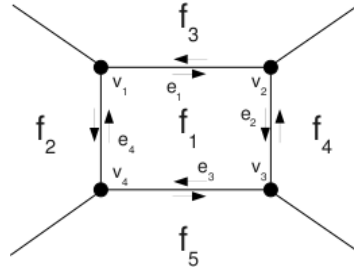


Figure 26: A face as a connection of edge sides. Face f_1 is composed of edge sides $v_1 \rightarrow v_2$, $v_2 \rightarrow v_3$, $v_3 \rightarrow v_4$ and $v_4 \rightarrow v_1$. The beginning of each one is represented by the corners (v_1, f_1, e_1) , (v_2, f_1, e_2) , (v_3, f_1, e_3) and (v_4, f_1, e_4) , respectively. Each edge is represented by the beginning of each of its edge sides. For example, $e_1 = ((v_1, f_1, e_1), (v_2, f_3, e_1))$ and $e_4 = ((v_1, f_2, e_4), (v_4, f_1, e_4))$.

In addition, the orientation of the boundary walk is automatically adjusted in order to better fit into the mesh.

In the following we explain how a new face is created. In order to ensure that the previously existing faces are not modified, only corners that belong to the boundaries are used. If a particular vertex does not have any boundary corner (because its rotations are completely included in the XDLFL or it has no rotations) a new rotation is created. The possibility of a particular vertex with more than one boundary corner may also arise. The feasibility of using each available boundary corner is tested through an operations sequence graph. We choose to execute the path that applies the fewest modifications to the XDLFL, which is always the shortest path.

In the next subsections, we will develop a set of *operation* data structures that make it possible to evaluate a sequence of XDLFL operators without actually modifying the XDLFL. These structures will then be used as nodes in the operations sequence graph. The method for operations sequence graph creation and identification of the shortest path is also presented.

4.3.1.1 Operations for face creation

A face is a cyclic connection of oriented edge sides (Fig. 26). One end of an edge side is connected to the beginning of the next, until the end of the last one is connected to the beginning of the first one, forming a cycle. The beginning of each edge side is represented by a corner: It represents a particular vertex in a particular face and the edge side starting at it. The end of an edge side is represented by the subsequent corner in the face's boundary walk, meaning that the end of each edge side is also the start of another one and *vice versa*.

In order to identify the possible operator combinations during face creation, we will demonstrate the process of a face creation. Suppose we have the situation shown in Fig. 27a and want to create a new face of which the boundary walk is $(v_1, v_2, v_3, v_4, v_5, v_6)$. Starting with v_1 , the procedure is now to look for a boundary corner representing it that already is or can be connected to v_2 . Since this edge side already exists in a boundary (Fig. 27b), nothing needs to be done. Since a first edge side is already found, we look for the possibility of connecting its end (the corner representing v_2 at the boundary) to v_3 . Because the fact that v_3 does not have any rotations, a new one must be created, making every new connection possible (Fig. 27c). Moving to the next edge side, $v_3 \rightarrow v_4$, we see that v_4 already has a boundary corner, and since this corner is not connected to v_3 , we can connect the end of the previous edge side to it (Fig. 27d). Vertex v_5 does not have any rotations, so a new one is created and the edge between v_4 and v_5 is inserted using the corner representing the end of the last edge side (Fig. 27e). For the connection of v_5 to v_6 , we see that v_6 already has a boundary corner (Fig. 27f), and since it has no connections to v_5 yet, that corner can be used. Closing the cycle, we see that the end of the previous edge side is already connected to v_1 (Fig. 27g), so nothing has to be done. Finally, the boundary flag is removed from the face containing the last edge side ($v_6 \rightarrow v_1$), delivering a new face with the desired boundary walk (Fig. 27h).

In the example shown, we were able to immediately find the correct operators for the edge insertions: The corners which were already in the boundaries, or the newly created corners for those vertices which had no rotations. But it is not always that simple. For example, suppose that v_6 has a single rotation which contains two boundary corners: One that is already connected to v_1 and one that is not. If in step f we had chosen the corner that is not connected to v_1 for the creation of an edge between v_5 and v_6 , in step g we would have seen that the rotation already has a connection to v_1 , but it is not represented by the end of the last edge side, thus making it impossible to close the face. In order to avoid such dead ends, an operations sequence graph will be used in the next section, which will allow us to exploit all possibilities.

We will explore here a way of representing a sequence of operations without actually applying it to the XDLFL. If we directly apply the operations using the first operators we can find, a dead end might be reached and some operations would have to be undone in order to test new ones, what may complicate the process and worsen its performance. We want to test the operations to make sure they will lead to the desired face, before applying modifications to the XDLFL.

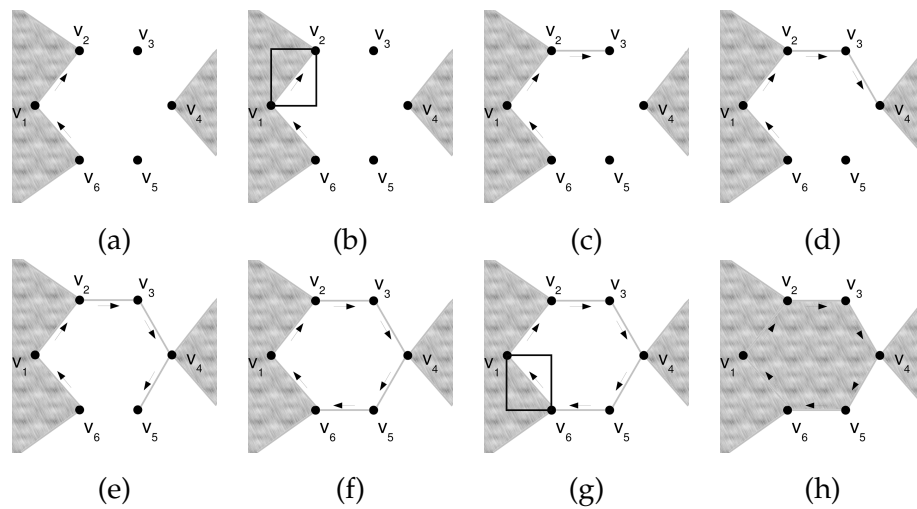


Figure 27: Creation of a face with a boundary walk equal to $(v_1, v_2, v_3, v_4, v_5, v_6)$; gray areas indicate internal regions of the mesh. (a) Initial situation, where all vertices have boundary corners, except for v_3 and v_5 , which do not have any rotation; (b) Edge side $v_1 \rightarrow v_2$ already exists in a boundary; (c) Creation of new rotation for v_3 and edge insertion; (d) Creating edge side $v_3 \rightarrow v_4$ using the previously existing boundary corner; (e) Creation of a new rotation for v_5 and edge insertion; (f) Creating edge side $v_5 \rightarrow v_6$ using the previously existing boundary corner; (g) Edge side $v_6 \rightarrow v_1$ already exists in a boundary; (h) Removal of boundary flag.

For this reason we introduce a set of *operation* data structures. These data structures represent one or more operations, and contain part (or, in some cases, all) of the operands required for their execution. The other operands are the results of the execution of the previous data structure. The results of an *operation* data structure execution are always an edge side start and end. Each *operation* has an EXECUTE operator, which takes another *operation* as argument (the previously executed one), and executes the represented operations. The following *operations* were defined for our purposes:

- *CreateRotationOperation (CR)*: Creates a new rotation for a particular vertex and labels the newly created face a boundary. It is initialized with a vertex, which will receive the new rotation. It is used only as the initial *operation* in cases where a new rotation has to be created for the very first vertex and therefore does not require the outputs of a previous *operation*. After executing, both edge side start and end outputs are set to the newly created corner, since it is a point-sphere.
- *InsertEdgeOperation (IE)*: Inserts an edge between the previous *operation's* edge side end output and a corner, passed during the initialization of this *operation*. If it is initialized with two corners, it does not require the output of a previous *operation* in order to execute. After executing, it sets the edge side start output to the corner representing the edge side in the direction of the edge insertion, and the edge side end to its subsequent in the face's boundary walk.
- *CreateRotationAndInsertEdgeOperation (CRIE)*: Inserts an edge between the previous *operation's* edge side end output and a vertex, passed during the initialization of this *operation*, after creating a new rotation for it. If it is initialized with a corner and a vertex, it does not require the output of a previous *operation* in order to execute. After executing, the outputs are set exactly as for the *InsertEdgeOperation*.
- *InsertEdgeAndRemoveBoundaryLabelOperation (IERBL)*: Inserts an edge between the previous *operation's* edge side end output and the edge side start output of the first operation in the operation sequence, and takes no initialization parameter. After that, it removes the boundary label of the face containing the edge side in the same direction of the edge insertion. It is a final node, and therefore does not set any outputs.
- *RemoveBoundaryLabelOperation (RBL)*: Removes the boundary label from the face containing the edge side outputs from the

previous *operation*. It is a final node, and therefore does not set any outputs.

- *DoNothingOperation (DN)*: Its execution method simply sets the corners passed during its initialization as outputs.

These data structures will be later used as nodes in the operations sequence graph.

In the example of Fig. 27, supposing that the boundary corners representing v_1, v_2, v_4 and v_6 are c_1, c_2, c_4 and c_6 respectively, the whole face creation operation can be represented by a sequence of *operation* data structures as follow:

$$DN(c_1, c_2) \rightarrow CRIE(v_3) \rightarrow IE(c_4) \rightarrow CRIE(v_5) \rightarrow IE(c_6) \rightarrow DN(c_6, c_1) \rightarrow RBL()$$

After the sequential execution of the previous *operation* sequence, the same results are obtained.

4.3.1.2 Operations sequence graph and shortest path

We present here a method that constructs and analyzes the whole operations sequence graph but stores only the path that is currently being tested and the shortest path for the face creation. Shortest path, refers to a sequence of previously presented *operation* data structures that will apply the fewest amount of modifications to the XDLFL after sequential execution.

In order to compute the number of modifications that will have to be applied, we sum up the number of *operations* that are going to be executed. Every time an *operation* is added to the path, a cost variable is updated depending on the number of operators it executes. So, adding a *DN* or a *RBL* does not change the cost; a *CR*, *IE* or *IERBL* increases the cost by one; and finally, a *CRIE* increases the cost by two.

Algorithm 4.4 shows the procedure which analyzes all possible paths for the creation of a new face with boundary walk $(v_0 \dots v_{N-1})$ and finds the shortest one, through a deep-first search. It tests the possibilities of connecting a particular corner c to every boundary corner representing a particular vertex v_i , which are contained in a corner set B_i . In addition, the possibility of creating a new rotation for vertex v_i is always tested, because if every other path fails, the creation of the face is ensured by connecting the newly created corners. This also ensures that if B_i is empty, because v_i has no rotations or no boundary corners, a new rotation will still be created.

The procedure executes as follow:

- Every time an edge can be inserted between the two corners c and c_i , the *IE* operation is added to the path. An edge can always be inserted if no edge exists that connects the rotations of c and c_i , or if $c = 0$, which means that a new rotation will be created for it. In the case where a new rotation will be created, it is always possible to insert an edge. The procedure is then recalled to check the possibilities of connecting c_i to v_{i+1} ;
- If no edge can be inserted because there is already an edge connecting the rotations of c and c_i , a test is done to determine whether the desired edge side is represented by c and ends at c_i , what means that c_i should be subsequent to c in its face. If this is the case, the edge side can be used and a *DN* is added to the path. The procedure is then recalled to check the possibilities of connecting c_i to v_{i+1} ;
- The possibility of connecting c to the corner of a new rotation is also tested, by adding a *CRIE* to the path and recalling the procedure with $c = 0$, signaling that a new rotation was created and that every edge insertion is possible.

If a particular path manages to get through all vertices in the desired boundary walk ($i = N$), the procedure tries to close the face. The face is closed by connecting the end of the last edge side to the beginning of the first edge side in the path. If this connection is possible and the cost is smaller than the minimal cost (the cost of the current shortest path), then the shortest path becomes the current path.

4.3.1.3 *The operator*

We present the *CREATEFACE* operator, which takes a sequence of vertices $S = (v_0 \dots v_{N-1})$ as an argument and creates a new face with those vertices in its boundary walk. This operator also tests creation of the face having S in back to front order (S^{-1}). In doing so, the user does not have to worry about the correct orientation of the faces during specification because the operator will automatically adjust it.

Algorithm 4.5 shows the *CREATEFACE* operator. It finds all boundary corners representing each vertex in S and tries to sequentially connect them using the *FINDSHORTESTPATH* procedure. The creation of new rotations is also tested. This ensures that a face will always be created even when v_0 has no boundary corners or no rotations. After that, the procedure is repeated using S^{-1} instead of S . Finally, the operations in the shortest path are executed, in order to actually modify the XDLFL.

Algorithm 4.4 Procedure for the construction of an operations sequence graph and identification of the shortest path.

```

FindShortestPath(Corner  $c$ , Corner set  $B_i$ , actual cost)
2
  if  $i = N$  then
4    CloseFace( $c$ , actual cost)
  else
6    for each  $c_i \in B_i$  do
      if an edge can be inserted between  $c$  and  $c_i$  then
8        if actual path is empty then push  $IE(c, c_i)$  into the actual
          path
          else push  $IE(c_i)$  into the actual path
10       FindShortestPath( $c_i, B_{i+1}$ , actual cost + 1)
      else if  $c_i$  follows  $c$  in  $c$ .face then
12       Push  $DN(c, c_i)$  into the actual path
          FindShortestPath( $c_i, B_{i+1}$ , actual cost)
14       Pop last operation from actual path
      if actual path is empty then push  $CRIE(c, v_i)$  into the actual
        path
16     else push  $CRIE(v_i)$  into the actual path
          FindShortestPath( $0, B_{i+1}$ , actual cost + 2)
18     Pop last operation from actual path

20
  CloseFace(Corner  $c$ , actual cost)
22
  if an edge can be inserted between  $c$  and  $c_0$  and (actual cost + 1)
    < minimal cost then
24    minimal cost  $\leftarrow$  actual cost + 1
        shortest path  $\leftarrow$  actual path
26    Push  $IERBL$ (first operation in the shortest path) into the
      minimal path
  else if  $c_0$  follows  $c$  in  $c$ .face and actual cost < minimal cost then
28    minimal cost  $\leftarrow$  actual cost
        shortest path  $\leftarrow$  actual path
30    Push  $RBL()$  into the shortest path

```

Algorithm 4.5 The CREATEFACE operator.

```

CreateFace(Vertex sequence S)
2  actual path  $\leftarrow \emptyset$ 
4  shortest path  $\leftarrow \emptyset$ 
    minimal cost  $\leftarrow 0$ 
6  for each  $v_i \in S$ , create a set  $B_i$  containing all boundary corners
      representing  $v_i$ 
    for each  $c_0 \in B_0$  do FindShortestPath( $c_0$ ,  $B_1$ ,  $\theta$ )
8  Add  $CR(v_0)$  to the actual path and execute FindShortestPath( $\theta$ ,  $B_1$ ,
    1)
    actual path  $\leftarrow \emptyset$ 
10 Repeat steps 5 and 6 for  $S^{-1}$ 
    for each operation  $o_k$  in the shortest path (sequentially) do
12  if  $k = 0$  then  $o_k.Execute()$ 
    else  $o_k.Execute(o_{k-1})$ 
  
```

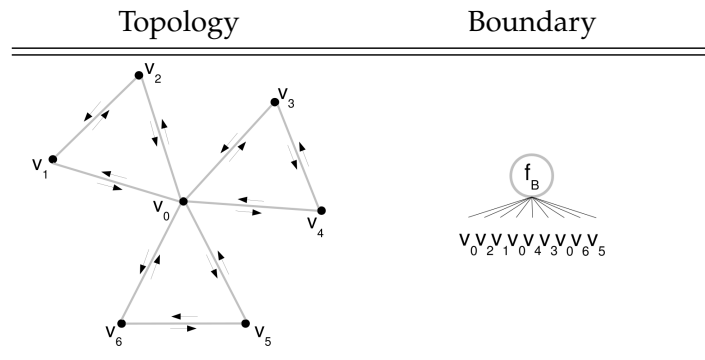


Figure 28: A 2-manifold interpretation of a non-manifold, where the boundary face binds the individual face loops together.

4.3.1.4 Unexpected behavior caused by 2-manifold interpretations of non-manifolds

Using the CREATEFACE operator, it is possible to create an arrangement where multiple faces are connected by a single vertex (Fig. 28). Such an arrangement is non-manifold, but it can be created without adding new rotations because there will always be a back face that contains the edge sides in inverse orientation and connects the loops. In Fig. 28, the boundary face acts as the back face, containing the loops of all three other faces and, consequently, including v_0 three times in its boundary walk, which is the connection point between the three face loops. Such an arrangement is a 2-manifold interpretation of a non-manifold.

Suppose the three faces shown in Fig. 28 were created as follows: `CREATEFACE((v0,v1,v2))`, `CREATEFACE((v0,v3,v4))` and `CREATEFACE((v0,v5,v6))`. Since v_0 is always present in a boundary face, no new rotations are created. Now suppose we want to add a fourth face, which is (v_0, v_4, v_5) . Although we clearly see that this face fits perfectly into the mesh, the `CREATEFACE` operator will be forced to create a new rotation for either v_5 or v_4 , because both of them are already connected to v_0 in the boundary, but to different corners. Starting at the edge side $v_0 \rightarrow v_4$, which already exists, and moving to the edge side $v_4 \rightarrow v_5$, which is created using the corners in the boundary, we reach the edge side $v_5 \rightarrow v_0$, which also previously exists. Unfortunately, the end of this edge side is the corner representing $v_0 \rightarrow v_2$, which is different from the first one ($v_0 \rightarrow v_4$), making this face impossible to close without adding a new rotation.

Initializing an entire mesh, by adding the faces in an order which creates 2-manifold interpretations of non-manifolds, will lead to unexpected behavior of the `CREATEFACE` operator and an undesired representation. This problem can be overcome by the initialization using posterior edge list creation, to be shown in Section 4.4.

4.3.2 The `REMOVEFACE` operator

The `REMOVEFACE` operator is the inverse of `CREATEFACE`: It takes a face as argument, removes it from the XDLFL (or transforms it into a boundary) and adjusts the boundary to fit the new topology. It also removes the rotations which are no longer used after the removal of the specified face.

Removing a particular face, f , consists of deleting any edge of it that has a boundary edge side. This will cause the face to merge with the boundary face and, consequently, the previous internal edges (the ones that had no edge sides in the boundary) to have a boundary edge side. If after deleting the edges, any rotation is left with a single corner, it is deleted. Furthermore, when a particular face does not have any boundary edge, it is labeled a boundary.

Fig. 29 illustrates the removal of two faces from a mesh (Fig. 29a): The removal of a face completely contained in the mesh (Fig. 29b) when labeled a boundary and the removal of a face which has boundary edges (Fig. 29c), where only these edges are removed, making the others join the boundary. The `REMOVEFACE` operator is presented in Algorithm 4.6.

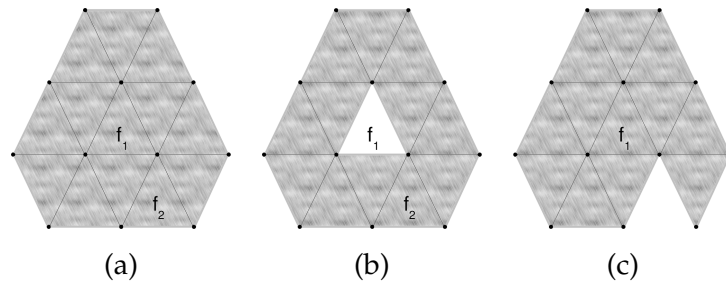


Figure 29: The results of the REMOVEFACE operator: (a) The original mesh; (b) After removing f_1 (it becomes a boundary face); (c) After removing f_2 (it is completely removed).

Algorithm 4.6 The REMOVEFACE operator.

```

1 RemoveFace(Face f)
3 for every edge  $e$  in the boundary walk of  $f$  do
   if  $e$  has a boundary edge side then push it into a list B
5 if B is empty then label IsBoundary( $f$ , true)
   else
7   for each  $e$  in B do
     Let  $R_1$  and  $R_2$  be the rotations containing the corners
       representing both edge sides of  $e$ 
9   DeleteEdge( $e$ )
     if  $R_1$  is has only one corner then DeleteRotation( $R_1$ )
11  if  $R_2$  is has only one corner then DeleteRotation( $R_2$ )

```

4.4 XDLFL INITIALIZATION BY POSTERIOR EDGE LIST CREATION

We present in the following a method for the initialization of the XDLFL with an arbitrary polygonal data. This method does not suffer from 2-manifold interpretations of non-manifolds because it does not use the operators presented. Instead, it directly manipulates the structure.

This method consists of initializing the face and vertex lists and then following the rotation of each vertex in order to initialize the edge list. Afterwards the edge list is searched for the edges that have only one edge side, and boundary faces are constructed in order to add the opposite edge sides.

The method works as follow:

1. For each vertex in the polygonal data, create a XDLFL vertex;
2. For each face in the polygonal data, create a XDLFL face, and for each vertex in the boundary walk, create a corner node. Each corner node representing a vertex v is added to a temporary list of corners C_v , instead of adding it to a rotation;
3. For each vertex in the XDLFL, create a new rotation R , pick one corner c in C_v , and follow its rotation through the VERTEXTRACE operation [3], until c is reached again, moving each corner found to the new rotation and creating the edge entries (if not already created by following previous rotations). Any time the VERTEXTRACE operation cannot continue, a boundary is found, so that an edge entry is created if a single edge side and the rotation is followed in the opposite direction, starting at c again;
4. If C_v still has corners, this signifies that it has more rotations, and step 3 is repeated;
5. Once all C_v s are empty, the edge list is searched for an edge e with a single edge side. If such an edge exists, a new boundary face is created and an edge side is created in the opposite direction. The edges containing a single edge side that are connected are selected, and edge sides are added in the opposite direction in the same boundary face until e is reached again;
6. Repeat step 5 until there are no more edges with a single edge side.

4.5 DISCUSSION

The *Doubly Linked Face List* (DLFL) is a very useful data structure for the representation of 2-manifold meshes. It is compact and can be manipulated using a set of simple operators. Furthermore, it allows direct iteration over mesh elements (vertices, edges and faces) through its lists, ensuring good performance for mesh processing algorithms. However, the use of the DLFL for the implementation of practical applications is very limited, mainly because of two drawbacks: (1) Its operators may be very ambiguous in the user level, modifying the structure in unexpected ways; and (2) the DLFL is only able to represent 2-manifold objects.

For solving the first drawback, one may argue that the problem is not the representation but the user interface design. In the *TopMod3D* [13], a DLFL-based mesh modeler, ambiguities are eliminated by requiring that the user first selects a face and then a corner inside it for the edge insertion operation. Although it sounds like a very simple solution, it also has a major drawback: The user must *know* which faces and corners need to be selected in order to achieve the correct modification of the structure. In other words, the user must have knowledge about the internal data structure, which is not practical in many cases, such as with applications for surgery simulation, for example. In the medical environment, the physician (the user of the application) is usually not familiar with topological concepts and data structures, and expects that the operations are performed in a transparent, intuitive and correct manner.

The representation capabilities of the DLFL may also be a limiting issue. While other data structures allow the representation of boundaries and other non-manifold components, the DLFL always requires the 2-manifold property, sometimes creating 2-manifold interpretations of non-manifold data. The interpretation of non-manifolds as 2-manifolds is also ambiguous, as many different interpretations are possible, and may yield different structures for the same data. Furthermore, many practical applications take advantage of non-manifold representation capabilities.

In this work, we extended the DLFL for the representation of *2-manifolds with boundaries* and *2-pseudomanifolds*, which allows the representation of non-manifold objects by subdividing them into simpler 2-manifold parts. These extensions make use of some observations of the behavior of DLFL operators, and do not necessitate modifications to the original concepts. In fact, the practical integration of the extensions is rather trivial. Using these extensions, we also showed how to avoid ambiguities in the original DLFL operators and presented a

new set of intuitive operators for the manipulation of the extensions and for the unambiguous manipulation of the data structure. This new data structure has been termed the *Extended Doubly Linked Face List* (XDLFL).

The XDLFL increases very slightly the memory usage of the DLFL, maintaining it smaller than the memory usage of other well-known data structures. Furthermore, in the XDLFL, testing whether an element belongs to a boundary can be performed in constant time for vertices and edges, and with linear complexity at most for faces. Boundaries queries can be performed in constant time. Testing whether a vertex is 2-pseudomanifold can also be performed in constant time.

We strongly believe that XDLFL expands the application field of the DLFL by solving some of its internal problems and increasing its representation capabilities, being suitable for the development of real-world applications and allowing the user to interact with the structure in a transparent way.

DISTORTIONS BETWEEN MULTI-MODAL SURFACES

The least initial deviation from the truth is multiplied later a thousandfold.

— Aristotle

As pre- and intra-operative data are usually acquired by different modalities, the resulting surfaces may be subject to different sources of error. These might in turn lead to different descriptors in corresponding regions. Understanding how descriptors vary between two surfaces acquired by different modalities is therefore crucial to achieve an adequate matching. Some methods for comparing two surfaces have been proposed [64, 236]. However, they generally assume that the geometric variations between input data are isotropic, which may not be the case when they originate from different sources or in case they are affected by noise (Fig. 30a). Aspert et al. [17] used the *symmetrical Hausdorff distance* metric to compare two surfaces. This method considers the minimal distances both between source and target and between target and source, in order to identify false correspondences. This metric can potentially compensate for anisotropic variations. Nevertheless, it can only be used for the establishment of a global error metric and not for the characterization of local deviations. Furthermore, the aforementioned techniques only consider the distances to the discrete surfaces, by computing the distances to mesh triangles, instead of the underlying surfaces (Fig. 30b). They furthermore do not make use of descriptors as a means for surface comparison, but geometric vertex distances.

We present here: (1) a framework for the local and global comparison between two surfaces taking into account anisotropic variations and underlying surfaces instead of their discretizations (Sec. 5.1); (2) This framework is used to analyze the differences between descriptors of time-of-flight (ToF) surfaces and corresponding computed tomography (CT) surfaces, acquired *in vitro* (Sec. 5.2).

5.1 MESH COMPARISON FRAMEWORK

Given two pre-aligned meshes as input, the reference mesh (R) and the test mesh (T), the proposed framework works as follows: First, correspondences are established between T and R taking into account

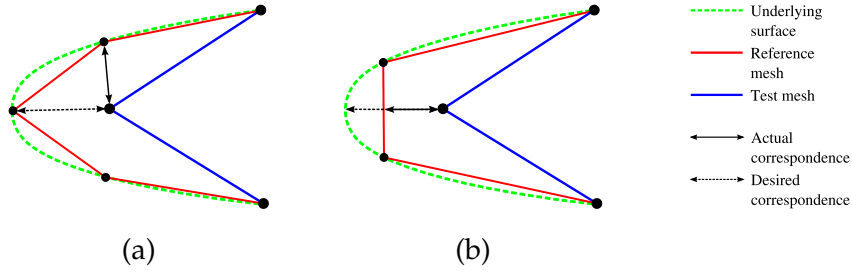


Figure 30: Correspondence search between a point on the test mesh and the reference mesh: (a) Due to the anisotropy in the test mesh, the correct correspondence can not be found with the regular closest point operation; (b) The correct correspondence on the underlying surface can not be found if only the mesh is considered, possibly yielding wrong interpolation of descriptors, such as curvatures.

anisotropies as well as the underlying surface (Sec. 5.1.1); Second, descriptors are computed per vertex, and local distance metrics are used for assessing descriptor variations between corresponding vertices (Sec. 5.1.2); Finally, a global distance metric is computed based on the local distances (Sec. 5.1.3).

An important prerequisite of the proposed framework is the regularity of the reference mesh (i.e., all vertices have six neighbors). Mesh regularity is required for the achievement of various interpolation properties during the establishment of correspondences (Sec. 5.1.1). While several methods have been proposed for mesh regularization, in this work the approach presented by Vidal et al. [262] was applied. This approach does not degrade the fidelity to the initial mesh and preserves relevant features. These are desired properties when comparing two meshes representing the same object.

5.1.1 Establishment of correspondences

Initial correspondences are established by the computation of distances between the vertices of the test and reference surfaces. The closest point operator (C) is employed for setting these correspondences:

$$C(p_i, R) = \arg \min_{q_j \in R} d(p_i, q_j) \quad (5.1)$$

where $d(p_i, q_j)$ denotes an arbitrary distance measure between the vertices $p_i \in T$ and $q_j \in R$. In order to account for anisotropies in

the data, we have adopted an anisotropic variation of the Euclidean distance measure, defined as follows:

$$d(p_i, q_j) = \|W_{p_i q_j}(p_i - q_j)\| \quad (5.2)$$

where $W_{p_i q_j}$ denotes the weighting matrix that represents the anisotropic representation error of p_i and q_j [182]. The weighting matrix $W_{p_i q_j}$ is computed as follows:

$$W_{p_i q_j} = \frac{1}{\sqrt{\Sigma_{p_i} + \Sigma_{q_j}}} \quad (5.3)$$

where Σ_{p_i} and Σ_{q_j} are the covariance matrices representing the localization errors of vertices p_i and q_j . Maier-Hein et al. [181] showed how to define these covariance matrices for points lying on surfaces acquired by time-of-flight cameras. The positions of these points are not very precise in the direction of the acquisition beams, but they show high lateral accuracy.

Although the introduction of anisotropic weighting in the closest point operator is suitable for finding the correct vertex correspondences, it is not able to deliver the correct correspondences to the underlying surface (Fig. 30b). In order to compute these correspondences, the underlying surface is interpolated by increasing the mesh resolution. As we want to compute differences between second order descriptors, such as curvatures, linear interpolation over triangles yields incorrect results. For the correct estimation of these descriptors, an interpolation scheme that guarantees continuity of second order derivatives is required (C^2 continuity). Assuming that the underlying surface can be locally approximated by cubic equations, the mesh resolution is increased by applying mesh subdivision schemes. These include Loop [243] or Butterfly [92], which guarantee C^2 continuity for regular meshes (Fig. 31). Note that these subdivision schemes can be computed locally for the region around a particular vertex.

After locating the closest vertex $q_j \in R$ to a particular vertex $p_i \in T$, the correct correspondence between p_i and the underlying surface of R can be found by: (1) Iteratively refining the local region around q_j (L_j) through the application of a subdivision scheme, and (2) by searching for the closest vertex to p_i in L_j . This procedure is repeated until the geometric distance between the closest vertex and its neighbors is smaller than a pre-defined threshold. Note that, for speed-up reasons, all regions L_j can be pre-computed. After finding the closest vertex on the underlying surface represented by mesh R , its descriptors need to be computed. These descriptors may also be pre-computed.

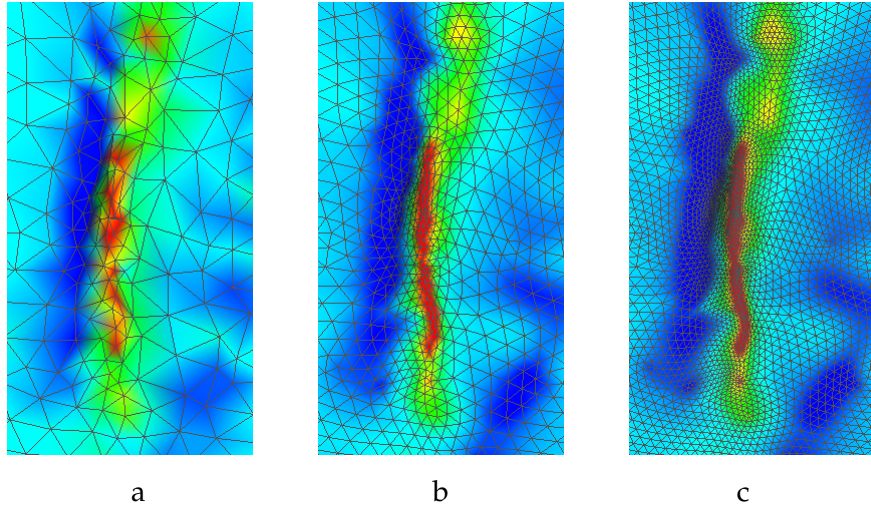


Figure 31: Two consecutive refinements of the original surface (a) by the Loop subdivision scheme, which guarantees C^2 continuity for regular meshes. Colors represent the mean curvature for each vertex. Note that, after surface refinement, curvature values remain approximately the same.

5.1.2 Descriptors and local distance metrics

Based on the previously established correspondences, distance metrics are employed for the comparison of different vertex descriptors. Such descriptors include *geometric position*, *vertex normal*, *mean curvature* [189], *Gaussian curvature* [189], and *curvedness* [155]. The framework is independent of the descriptors chosen. However the local distance metrics must be chosen in accordance. For example, regarding the computation of distances between normals, an appropriate metric would be the angle between them. In case of our our descriptors, we currently define the local distance metric as the angle for the normals, and the Euclidean distance for all others.

5.1.3 Global distance metric

Several statistical metrics were computed as means for evaluating the global deviation between test and reference meshes based on the set of their local distances. They include mean, root mean square, median, first and third quantiles, maximal and minimal values, and the standard deviation. Similarly, descriptor variation statistics can be computed for particular regions defined above the meshes. In addition, a visualization of the local distances above the test surface is generated for a better understanding of the variations. Furthermore,

box-and-whisker diagrams are presented as means for visualization of the global variations.

5.2 ASSESSMENT OF THE DISTORTIONS BETWEEN TIME-OF-FLIGHT AND COMPUTED TOMOGRAPHIES

The evaluation was performed *in vitro* with ToF surfaces and their corresponding CT surfaces obtained from three porcine organs (liver, heart and lungs) [228]. Note, however, that the ToF and CT acquisitions were performed sequentially in order to avoid any kind of deformations caused by external forces. ToF range data is subject to noise as well as to systematic errors, such as the wiggling error and the intensity and temperature dependent distance errors [156]. In order to compensate for these errors, the range images were previously denoised [228]. ToF and CT surfaces were pre-aligned using landmarks.

Figs. 32 (liver) and 33 (lung) show the results of the local differences for three different descriptors, along with their values plotted over CT and ToF surfaces. Although the meshes have slightly different shapes, as the meshes were generated from different modalities, our framework was able to correctly measure their variations.

Fig. 34 shows a comparison between our framework and a standard method for mesh difference computation. In the latter the closest point operator does not account for anisotropies (Euclidean closest point) and distances are computed directly to the mesh instead of the underlying surface. The maximal differences in all cases are visibly smaller when using our framework, as it manages to compute the correct correspondences between the surfaces.

Fig. 35 shows the error for the descriptors computed on the ToF surface in relative percentual variation from the CT surface, for both the original (noisy) and smooth ToF surfaces. Regarding the mean curvature of smooth surfaces, most errors lie between 10% and 30% of the corresponding CT surface. The Gaussian curvature showed to be a little more robust, with most errors ranging from 3% to 10%. Regarding curvedness, most errors lie between 5% and 20%.

5.3 DISCUSSION

The presented framework for surface comparison (Sec. 5) was successfully applied to assess the quality of surfaces generated from ToF range data in relation to CT surfaces. According to our results, computation of differences between surfaces requires not only global measures, such as the commonly used root-mean-square (RMS) distance,

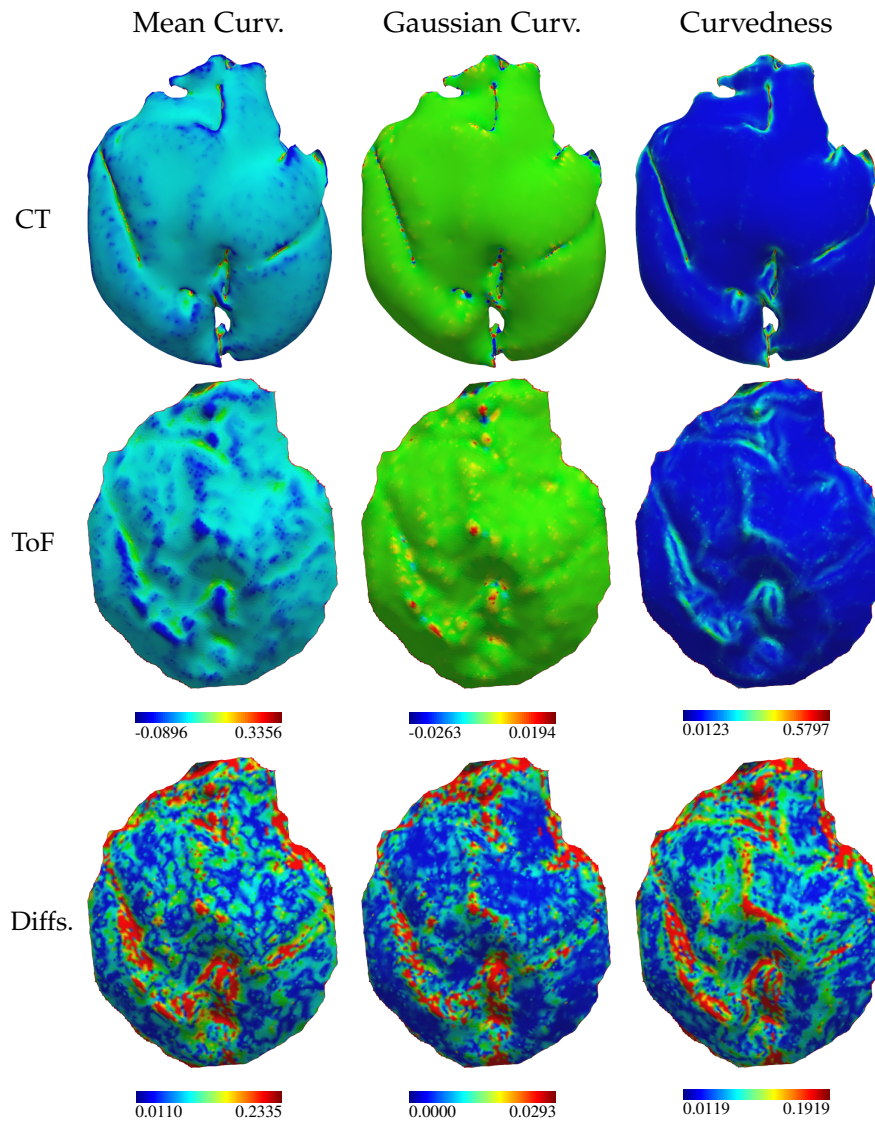


Figure 32: Local differences of three different descriptors between surfaces representing a porcine liver acquired with CT and ToF. The differences are shown in the bottom row, while the upper rows show the descriptor values on the CT and ToF surfaces.

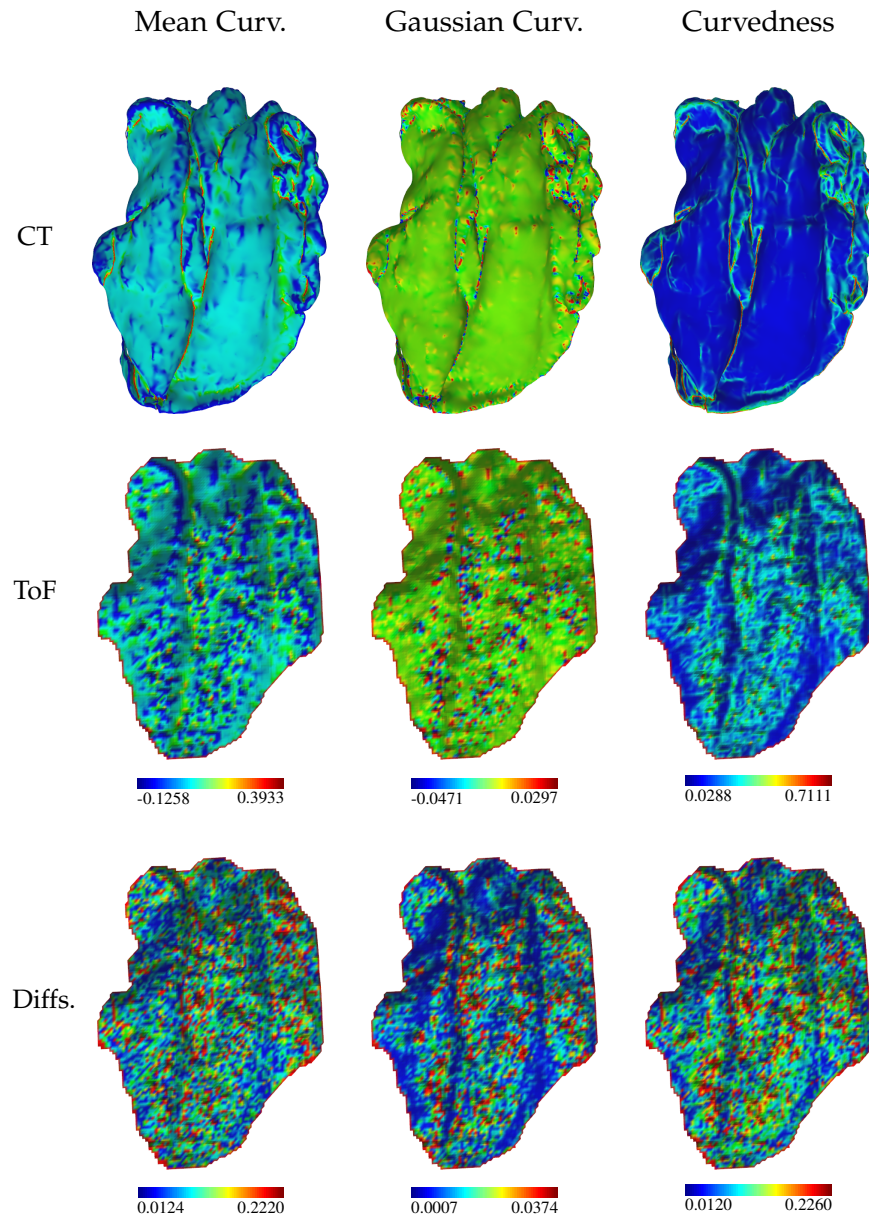


Figure 33: Local differences of three different descriptors between surfaces representing a porcine lung acquired with CT and ToF. The differences are shown in the bottom row, while the upper rows show the descriptor values on the CT and ToF surfaces.

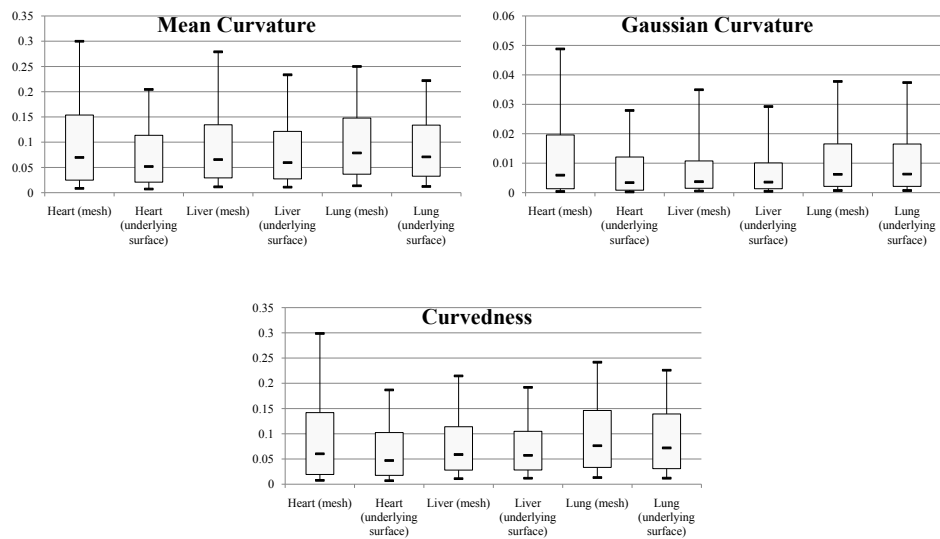


Figure 34: Comparison between the descriptors' distortions between ToF and CT surfaces found by our framework (underlying surface) and by the direct computation of differences between the meshes using the Euclidean closest point (mesh).

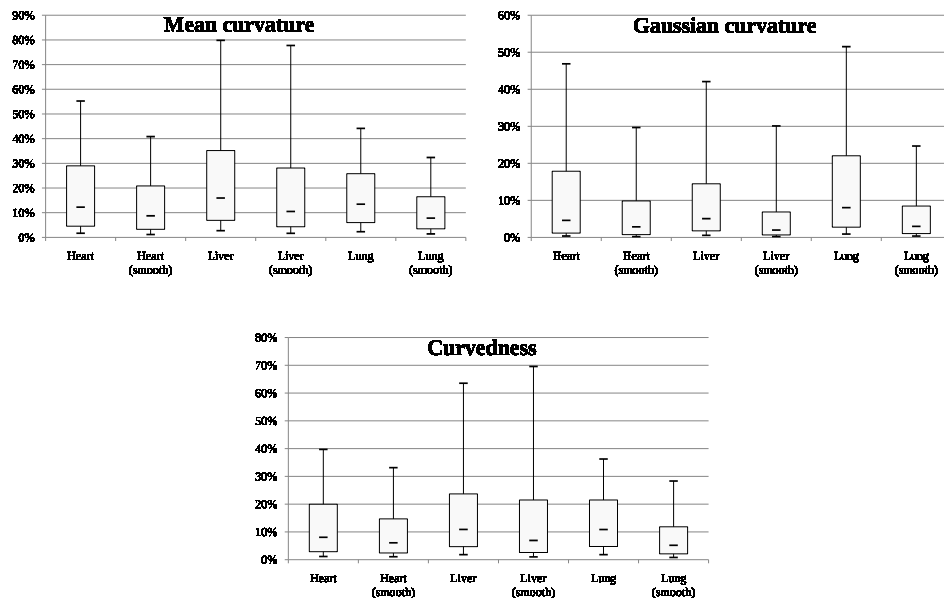


Figure 35: Descriptors distortions between ToF and CT surfaces shown in relative percentual variation, for the original (noisy) and the smoothed ToF surfaces.

but also assessment of local variations. By the single use of global measures, the specific characterization of the surface variations cannot be precisely determined. This likely due to the global value not indicating where variations occur. On the other hand, by comparing values or visualizations of descriptors with descriptor differences, the localization of problematic areas is straightforward.

When dealing with noisy surfaces, our framework did not show significant improvements compared to common distance measures, as noise affects the correct computation of descriptors. This issue can be observed in the “lung” data set, for example. In this case, noise-robust surface descriptors must be added to the framework for a correct assessment of differences between such surfaces. Furthermore, the quality of the initial alignment of the surfaces to be compared is crucial. Incorrect alignments lead to incorrect correspondences between the surfaces.

Due to the errors inherent to the ToF technology, many discrepancies are observed between ToF and CT. The relatively low resolution of ToF range images compared to CT images explains many of these variations. As the sampling is comparatively low, ToF range images are not able to capture sharp features and regions with high curvature variation. These are, however, clearly visible in high resolution CT data. In our experiments, acquiring ToF and CT surfaces sequentially in order to minimize deformations, moderate to high variations on curvature properties were observed. Surface matching methods specifically designed for the registration of multi-modal data, such as ToF to CT, cannot entirely rely on descriptor similarities for finding corresponding points.

SURFACE MATCHING

*It takes these very simple-minded instructions
- 'Go fetch a number, add it to this number, put the result there, perceive if
it's greater than this other number' -
but executes them at a rate of, let's say, 1,000,000 per second.
At 1,000,000 per second, the results appear to be magic.*

— Steve Jobs

As explained in the previous chapters, surface matching in intra-operative situations is very challenging. Because of noise, deformations, distortions, and the availability of only partially overlapping, nearly flat surfaces, most existing methods for surface matching are inadequate for the use in intra-operative registration. The main consequences of the above issues are different descriptors for points lying at the same anatomical locations and the absence of features that can be consistently selected on both source and target surfaces. Furthermore, methods that directly solve *quadratic assignment problems* (QAP) employing geodesic distances or heat diffusion metrics as regularization terms are both computationally inefficient and error prone, as most points have similar geodesic or diffusion fields.

In order to overcome the aforementioned issues, we present here two different approaches for surface matching. In the first approach (Sec. 6.1), the surfaces are segmented in regions of similar curvature properties. Employing segmentation eliminates the requirement for feature selection. These regions are represented by adjacency graphs, and correspondences are found by means of graph matching. To solve the graph matching problem, which is also usually formulated as a QAP, two different techniques were employed: The first technique does not solve the QAP directly, but iteratively computes similarity scores for graph nodes based on the similarity of their neighborhoods. These scores are used to compute a set of correspondences as a *linear assignment problem* (LAP). The second technique for solving the graph matching problem is based on *relaxation*.

The second approach for surface matching (Sec. 6.2), does not rely on the consistent selection of features. Features are instead selected only on the source surface. For each feature, a set of correspondences candidates on the target surface are computed, based on their descriptor similarities. As the surfaces are nearly planar, a new error

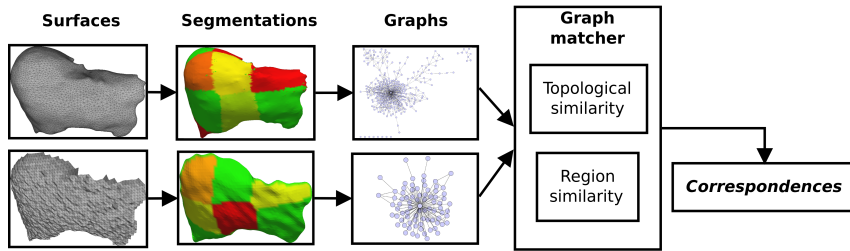


Figure 36: The region-based approach for surface matching. In order to match two surfaces, they are segmented and graph representations are created. These graphs are then matched in order to establish correspondences between them.

metric for measuring the reliability of a particular spatial landmark configuration is presented. To more reliable landmark configurations, a smaller *configuration error* is assigned, which is used to weight a *fitting error*. The fitting error provides a means of measuring how well two surfaces fit to each other given a set of correspondences between them. We further employ a constrained greedy approach for optimization.

Both approaches for surface matching were evaluated in a variety of experiments, designed to reflect the diverse settings and issues that may occur in intra-operative registration scenarios. For all experiments, two modalities for surface acquisition were employed: Surfaces acquired by a time-of-flight (ToF) camera, representing the intra-operative surfaces, and surfaces generated from computed tomographies (CT), representing the pre-operative surfaces. Furthermore, simulated ToF surfaces were used in some experiments in order to provide ground truth data for the evaluation of our approaches. All experiments were designed to be as close to reality as possible, but still allowing an accurate evaluation of the methods in question.

6.1 REGION-BASED APPROACH

According to Audette et al. [22], region-based surface matching consists of three steps (Fig. 36): First, the input meshes are segmented into regions based on the homogeneity of the local shape characteristics or some specific boundary circumscription. Second, the surface is represented as an adjacency graph, as the notion of neighborhood is natural between regions. Finally, a matching between the graphs is computed in order to establish correspondences between the surfaces.

One of the first methods to employ a graph matching-based approach for correspondences search in the context of surface matching

was introduced by Kehtarnavaz and Mohan [148]. This method uses region features obtained from segmenting the input surfaces based on *classes* of curvature [33]. The graphs are matched by searching a common *clique*: A subgraph of a particular graph in which every two nodes are connected. The common clique search problem is known to be NP-complete, and thus methods attempting to solve it are very time-consuming and cannot guarantee an optimal global solution in polynomial time. Furthermore, the search for common cliques bears two problems. On the one hand, there may be multiple common cliques on the target surface for a clique on the source surface. This is particularly the case when the surfaces are nearly flat, making the matching ambiguous. On the other hand, a common clique might not exist in case two non-identical surface instances shall be matched (*e.g.* due to noise or distortions).

More recently, it was shown that graph matching is a powerful tool for the establishment of correspondences between features [259, 280, 90]. These graph matching techniques utilize both the binary similarity between features and higher order similarity scores in order to minimize an error metric. They also consider the feature's neighborhoods. Still, the graph matching problem remains difficult, as due to noise and distortions, isomorphisms usually do not exist, posing an *inexact graph matching problem*. This problem is also known to be NP-complete [1]. Furthermore, as we deal with partially overlapping surfaces, we must search for common subgraphs instead of a direct matching between the given graphs.

As references to 3D surfaces in the graph matching literature are scarce [281], we applied here methods that iteratively compute similarity scores between the graph nodes, in order to evaluate the applicability of graph matching in intra-operative registration. These methods provide a similarity matrix, which can be used to compute correspondences by means of solving a linear assignment problem, which can be computed in polynomial time [81]. However, the literature describing graph matching is very vast, even for inexact graph matching, and we kindly refer the reader to [30, 69] for more information on this topic.

In the context of graph matching, we employ two different approaches for optimization: In the first approach, we iteratively score the similarity of neighborhoods and propagate it throughout the graph, resulting in similar neighborhoods having higher scores. This allows us to cast the problem as a *linear assignment problem* (see Secs. 3.3 and 3.4) and to solve it in polynomial time. In the second approach, the graph matching problem is formulated as a *quadratic as-*

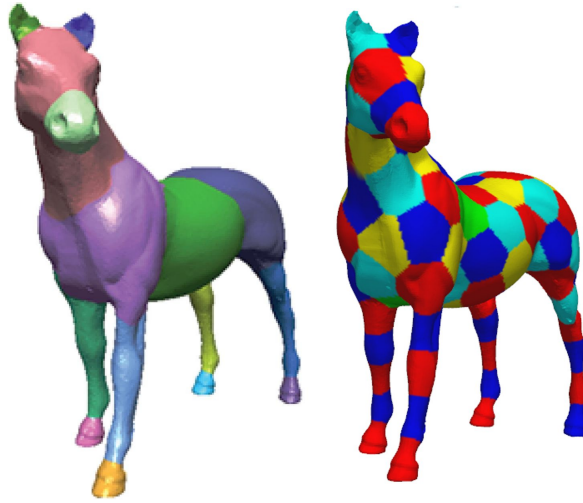


Figure 37: Examples of a part-type (left; taken from [168]) and surface-type mesh segmentation (right). The surface-type segmentation was computed by creating same-sized pentagons above the parametrized surface.

segment problem (see Secs. 3.3 and 3.4) and solved by means of relaxation.

In Sec. 6.1.2 we show how to construct graphs given a mesh segmentation. Sec. 6.1.1 reviews mesh segmentation techniques. The use of graph matching for the computation of correspondences is presented in Sec. 6.1.3.

6.1.1 Mesh segmentation

Mesh segmentation can be distinguished in two types: part-type and surface-type (Fig. 37) [231]. In part-type segmentation, the object is subdivided in meaningful parts, while in surface-type segmentation, the object is subdivided in patches according to some specific criteria. For more information on the topic, we kindly refer the reader to [18, 231, 59].

As most surfaces of interest for intra-operative registration do not present meaningful parts, the segmentation method of choice should be part-type. In order to fulfill the requirements of surface matching for intra-operative registration, a segmentation method must provide consistent results across surfaces, even in the presence of noise and distortions, or when only partially overlapping surfaces are available. Furthermore, the segmentation method must be able to partition the surfaces into patches of similar size, which should only contain elements with similar characteristics, in order to ensure relevant graph

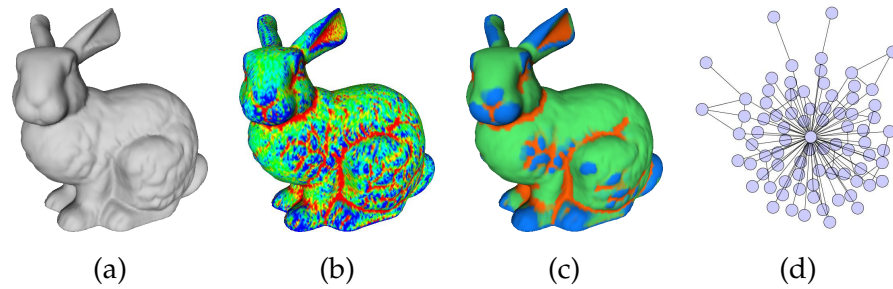


Figure 38: Segmentation results of the total-variation based method [78], on the mean curvature. (a) The input mesh; (b) The mean curvature; (c) Results of the segmentation using three different labels; (d) The respective adjacency graph representing the segmentation. Figs. (a-c) were taken from [78].

topologies and to allow the computation of descriptors that are coherent with the entire region. Designing a mesh segmentation algorithm that fulfills all these criteria is not trivial. A consistent mesh segmentation is critical for achieving accurate registration results.

Most mesh segmentation methods had initially been designed for texture mapping or remeshing (*e.g.* [67, 289, 284]), thus being consistent across similar meshes not of concern. In the case of methods that employ mesh segmentation for morphing (*e.g.* [158, 226]), consistency across similar meshes is crucial. In this case, however, correspondences between the surfaces are known beforehand, and this information is incorporated on the segmentation method for the achievement of consistent segmentations.

Delaunoy et al. [78] presented a method for mesh segmentation based on the minimization of the total-variation of a function defined on a surface (Fig. 38). This method bears the advantage of being very robust to noise. However, it requires a set of pre-defined labels, making it hard to achieve consistent results for the segmentation of multi-modal surfaces, as the labels would need to be adapted for each specific case. Furthermore, if the labels are not well distributed among the function's image, the segmentation results may generate graph topologies that are very ambiguous for matching. An example of such a scenario is shown in Fig. 38c, where the big region in green is a neighbor of all other regions, thus generating a star-shaped graph, with a single node in the middle (Fig. 38d). In addition, the computation of a descriptor that characterizes this entire green region is not trivial, as the region has an unconventional shape and contains very distinct components.

In this work, a simple region-growing method based on the variation of curvature properties is employed. Without any given noise

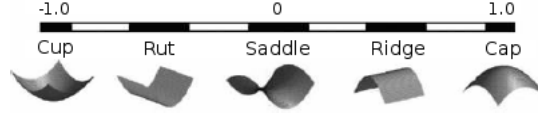


Figure 39: The shape-index and the respective shapes (based on the illustration found in [155])

or distortions (*i.e.* perfect conditions), this method generates exactly the same partitioning for meshes and sub-meshes. As measures for curvature properties, we employ *shape-index* and the *curvedness* [155]. The *shape index* of a particular surface vertex v gives a quantitative measure of the shape in the range $[-1, 1]$ (Fig. 39) and is computed as follows:

$$s(v) = \frac{2}{\pi} \arctan \frac{k_2(v) + k_1(v)}{k_2(v) - k_1(v)} \quad (k_1(v) \geq k_2(v)) \quad (6.1)$$

where $k_1(v)$ and $k_2(v)$ are the two principal curvatures of the surface at v , which measure how the surface bends at this position [189].

Curvedness is a positive measure that indicates the intensity of the curvature at a particular vertex v and is computed as follows:

$$c(v) = \sqrt{\frac{k_1^2(v) + k_2^2(v)}{2}} \quad (6.2)$$

It vanishes ($= 0$) only at planar vertices (where the shape index is undefined because at planar vertices $k_1(v) = k_2(v) = 0$) and is inversely proportional to the size of the object, and therefore not scale invariant [155].

Given two adjacent vertices v_1 and v_2 of a particular mesh, the variations in shape index (Δs) and curvedness (Δc) are computed according to the following equations:

$$\Delta s(v_1, v_2) = |s(v_1) - s(v_2)| \quad (6.3)$$

$$\Delta c(v_1, v_2) = \begin{cases} \frac{\max(c(v_1), c(v_2))}{\min(c(v_1), c(v_2))} - 1 & \min(c(v_1), c(v_2)) > 0 \\ 0 & c(v_1) = 0 \wedge c(v_2) = 0 \\ \infty & \text{otherwise} \end{cases} \quad (6.4)$$

Because of the fact that the curvedness scales inversely with size [155] and is thus dependent on the actual size of the object, computing its variation as a difference would deliver different values for the same vertices in two different object scales. In contrast, applying a quotient leads to a scale invariant measure of change in curvedness. In the

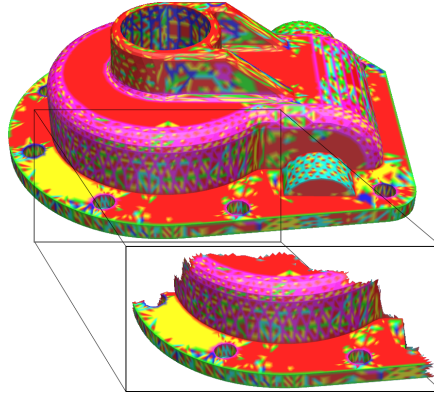


Figure 40: The segmentation of a mesh and a sub-mesh, which was extracted from the marked region. The regions are colored according to a sequential graph coloring method. Note that the submesh segmentation is equal to the segmentation of its corresponding part in the reference mesh (except for the boundaries of the submesh).

particular case where one of the vertices is associated with a planar surface region, there is either no variation, when both vertices are planar, or it is set to ∞ (maximal variation).

Starting at any vertex v , we use local variations of shape index and curvedness between it and its neighbors to segment the mesh. Vertices with variations smaller than given threshold values ($\delta_{\text{shape-index}}$ for shape index and $\delta_{\text{curvedness}}$ for curvedness) are assigned to the same region. This procedure is summarized in the following steps:

1. Create a new region R and add v to it;
2. For every vertex v_i adjacent to v that is not contained in any region, compute the shape index and curvedness variations $\Delta s(v, v_i)$ and $\Delta c(v, v_i)$. If $\Delta s(v, v_i) < \delta_{\text{shape-index}}$ and $\Delta c(v, v_i) < \delta_{\text{curvedness}}$, add v_i to R ;
3. For every new vertex v_n in R , make $v \leftarrow v_n$ and restart step 2, until there are no new vertices in R ;
4. Find a vertex v that has not yet been assigned to any region and restart step 1, until all vertices are contained in some region.

Fig. 40 shows an example of the segmentation of a surface and of an extracted subsurface. Note that as only local curvature changes are considered for the segmentation, the set of regions on the sub-mesh is a subset of the set of regions on the original mesh (except at the boundaries, where curvature, and thus shape index and curvedness, are not defined).

6.1.2 Graph construction

Once a surface has been partitioned into regions (Sec. 6.1.1), an attributed graph $G = (V, E, D)$ can be constructed, where V is a set of nodes, E is a set of arcs¹, and D a set of node attributes. Every nodes $n \in V$ represents a surface region and is attributed with a descriptor that can be used to characterize this region. Every arc $e \in E$ represents a neighborhood relation between two regions on the surface. In the case where oriented graphs are required, the edges are oriented from regions with lower descriptor magnitude to regions with higher descriptor magnitude.

For characterizing the regions, descriptors can be computed by using the mean of curvature properties, for example. More complex descriptors can also be used, as the computation of normal histograms for the region in question, as is presented in Sec. 6.2.2. In this work, we have adopted the mean of two curvature-based measures that describe the local shape of a particular mesh as region descriptors: the *shape index* and *curvedness*, which are both position and orientation independent [155] (see Sec. 6.1.1).

6.1.3 Correspondence search

We assume two distinct, weighted graphs $G_S = (V_S, E_S, D_S)$ and $G_T = (V_T, E_T, D_T)$ (see Sec. 6.1.2), which represent source and target surface, respectively. Our correspondence search approach is based on the computation of a similarity matrix between their nodes. This matrix is then used to compute an assignment between the nodes, so that the global similarity (the sum of the similarities of all assigned node pairs) is maximal. Our procedure is comprised of four stages: *descriptor similarity scoring* (sec. 6.1.3.1), *topological similarity scoring* (sec. 6.1.3.2), *correspondence computation* (sec. 6.1.3.3) and *post-processing* (sec. 6.1.3.4).

Let us define the similarity matrix $M_{\text{sim}} = [p_{n_t n_s}]_{|V_T| \times |V_S|}$, which holds the node similarity scores, based on which a set of correspondences between the graphs is computed. The similarity matrix M_{sim} is composed of two parts:

$$M_{\text{sim}} = M_{\text{desc}} + M_{\text{top}} \quad (6.5)$$

where $M_{\text{desc}} = [q_{n_t n_s}]_{|V_T| \times |V_S|}$ represents the node descriptor similarity scores, which holds the similarity between the descriptors of

¹ In this work we refer to the vertices and edges of the graphs as *nodes* and *arcs*, respectively. This is done in order to distinguish them from the vertices and edges of the surface meshes.

each pair of nodes in $(V_T \times V_S)$. $M_{\text{top}} = [r_{n_t n_s}]_{|V_T| \times |V_S|}$ represents the topological similarity scores, which hold the similarity between the topology of each pair of node in $(V_T \times V_S)$. It is computed according to a scoring method that minimizes an error functional. Once, the similarity scores have been computed, M_{sim} is used to determine an optimal global assignment between the graphs (correspondences). Finally, a post-processing based on the graph adjacency structure is employed to eliminate probably false correspondences and to identify potential new ones.

6.1.3.1 Descriptor similarity scoring

In order to assess the similarity between to nodes, we compute the similarity of their descriptors. The similarity scores between them are assessed by a Gaussian kernel over a function that computes the distance between two descriptors, thus delivering a similarity value in the interval $(0, 1]$. This kernel assigns higher scores to smaller distances, and *vice versa*. If the distance between the descriptors of two nodes is higher than a threshold value, the similarity between them is set to 0 (zero), denoting that a match between these nodes is impossible. Let $w_s \in D_T$ and $w_t \in D_T$ denote the descriptors of the nodes n_s and n_t , respectively. The descriptor similarity matrix $M_{\text{desc}} = [q_{n_t n_s}]_{|V_T| \times |V_S|}$ is then computed as follows:

$$q_{n_t n_s} = \begin{cases} e^{-\frac{d_{\text{desc}}(w_s, w_t)^2}{2\sigma^2}} & d_{\text{desc}}(w_s, w_t)^2 \leq \delta_{\text{desc}} \\ 0 & \text{otherwise} \end{cases} \quad (6.6)$$

where $d_{\text{desc}}(\cdot)$ denotes a distance metric between descriptors, σ the width of the Gaussian kernel, and δ_{desc} a descriptor distance threshold. In the case of the descriptor being represented by combinations of surface properties that lie in different spaces, multiple Gaussian kernels can be employed. For example, one can assume that the descriptors are composed of k different metrics, denoted by w^i , for $i = 1 \dots k$. In this case, the similarity score can be computed as follows:

$$q_{n_t n_s} = \frac{1}{k} \sum_{i=1}^k e^{-\frac{d_{\text{desc}}^i(w_s^i, w_t^i)^2}{2\sigma_i^2}} \quad (6.7)$$

where $d_{\text{desc}}^i(\cdot)$ denotes the distance metric for component i of the descriptor, and σ_i the width for the Gaussian kernel of this component.

6.1.3.2 Topological similarity scoring

To score the topological similarity between two weighted graphs, we consider two distinct methods: A direct scoring method [278] and a

method based on probabilistic relaxation, known as *SoftAssign* [118]. These methods consider both the topology of the graphs and the weights of the nodes to compute the scores. This is very significant, as the sole consideration of the similarity of topologies by itself would be insufficient to find corresponding node pairs, because many nodes are topologically identical.

DIRECT SCORING METHOD The direct scoring method iteratively computes local scores for each pair of nodes in $(V_T \times V_S)$ by a coupled node-arc scoring method [278]. According to this method, two nodes are considered similar if their topological neighborhoods are similar (*i.e.* their arcs are similar). By the same principle, two arcs are similar if their respective source and target nodes are considered similar. Starting with initial node scores, arc scores are computed, which are in turn used to compute the node scores iteratively. This procedure is repeated until a convergence limit is reached. In this way, scores between nodes are propagated to neighboring nodes at each iteration step. Note that this method requires the graphs to be oriented. As arc scores are computed in order to compute node scores, an update of the latter is provided in every second iteration only. In addition, two matrices are required to store the arc and node scores.

We modified the original method in order to use a single matrix and to deliver an update of the node scores in every iteration. Let us assume two nodes $n_s \in V_S$ and $n_t \in V_T$, and the functions $\text{src}(e)$, which returns the source node of a particular arc e , and $\text{trg}(e)$, which returns its target node. The scores for each element of the matrix M_{top} are computed iteratively as follows:

$$r_{n_t n_s}^k = \sum_{\text{src}(e_j)=n_t, \text{src}(e_i)=n_s} (r_{\text{src}(e_j)\text{src}(e_i)}^{k-1} + r_{\text{trg}(e_j)\text{trg}(e_i)}^{k-1}) + \sum_{\text{src}(e_j)=n_t, \text{trg}(e_i)=n_s} (r_{\text{src}(e_j)\text{src}(e_i)}^{k-1} + r_{\text{trg}(e_j)\text{trg}(e_i)}^{k-1}) \quad (6.8)$$

until $|r_{n_t n_s}^k - r_{n_t n_s}^{k-1}| \leq \varepsilon$ for every n_s and n_t , where ε denotes the convergence limit, and k the iteration. In order to consider the node weights, we initialize the topological similarity matrix with the descriptor similarities (Sec. 6.1.3.1), as $r_{n_t n_s}^0 = q_{n_t n_s}$.

SOFTASSIGN The *SoftAssign* algorithm [118] enforces a dual assignment constraint and computes a matrix (topological similarity matrix; M_{top}) with each element $r_{n_t n_s}$ denoting the probability of the correspondence between nodes n_t and n_s . The problem is solved by relaxing the boolean constraints of the assignment problem to allow

fuzzy assignments, thus making the error metric treatable (not NP-hard; see Sec. 3.4). After infinite iterations, M_{top} becomes a permutation matrix. The algorithm finds the topological similarity matrix that approximately minimizes the quadratic energy functional, *i.e.*, a *quadratic assignment problem* (QAP; see Secs. 3.3 and 3.4) given by:

$$E_{\text{SoftAssign}}(M_{\text{top}}) = -\frac{1}{2} \sum_{n_t \in V_T} \sum_{n_s \in V_S} \sum_{n_i \in V_T} \sum_{n_j \in V_S} r_{n_t n_s} r_{n_i n_j} \psi(n_t, n_s, n_i, n_j) \quad (6.9)$$

where $\psi : V_T \times V_S \times V_T \times V_S \rightarrow \mathbb{R}$ denotes the dual similarity function between an arc in G_S and an arc in G_T , given by:

$$\psi(n_t, n_s, n_i, n_j) = \begin{cases} \frac{1}{2}(q_{n_t n_s} + q_{n_i n_j}) & ((n_t, n_i) \in E_T) \wedge ((n_s, n_j) \in E_S) \\ 0 & \text{otherwise} \end{cases} \quad (6.10)$$

where $M_{\text{desc}} = [q_{n_t n_s}]_{|V_T| \times |V_S|}$ denotes the descriptor similarity matrix (Sec. 6.1.3.1).

6.1.3.3 Correspondences computation

In this stage, correspondences between the nodes of both graphs are computed, so that the global similarity becomes maximal. Such a problem is known as *linear assignment problem* (LAP; see Secs. 3.3 and 3.4) and there are several methods to solve it [81]. Those methods take a cost matrix and create an assignment between each row element to a column element, so that the sum of the costs of the assigned elements is maximal (or minimal, depending on the application).

We use the similarity matrix M_{sim} , which incorporates both topological and descriptor similarity scores for each pair of nodes $(n_t, n_s) \in V_T \times V_S$, as a cost matrix for the assignment problem. However, as M_{sim} is not square in cases with an unequal number of nodes in the graphs, the chosen method needs to deal with rectangular matrices. We therefore chose the popular *Munkres'* algorithm [41].

6.1.3.4 Post-processing

The goal of graph matching post-processing is to optimize the correspondences configuration between the input graphs by exploiting the information given by the graphs adjacency. We describe here two ways of performing this post-processing step. First, only probably false assignments are eliminated, using a simple heuristic. The second method applies a greedy optimization method to maximize an assignment energy.

ELIMINATION OF FALSE CORRESPONDENCES In order to reduce false matches among the assigned nodes, we adopt the premise that, if a particular node $n_s \in V_S$ was assigned to a node $n_t \in V_T$, there must be some other nodes in the neighborhood of n_s that were assigned to nodes in the neighborhood of n_t . If this premise is not confirmed, the assignment between these nodes is probably incorrect and thus removed.

Let $N(n, \delta)$ denote the topological neighborhood of a particular node n with radius δ , and $C_{\text{corresp}}(N(n_s, \delta_s), N(n_t, \delta_t))$ be a function that returns the number of nodes in the set $N(n_s, \delta_s)$ that are assigned to nodes in $N(n_t, \delta_t)$. If n_s was assigned to n_t in the correspondences computation stage, this assignment is eliminated if the following equation does not hold:

$$C_{\text{corresp}}(N(n_s, \delta_s), N(n_t, \delta_t)) \geq \beta \quad (6.11)$$

where β denotes the minimal number of node assignments that must exist between $N(n_s, \delta_s)$ and $N(n_t, \delta_t)$ in order to maintain the assignment between n_s and n_t .

GREEDY OPTIMIZATION The goal of this post-processing method is to reconfigure the correspondences between the input graphs in order to maximize an assignment energy. The assignment energy is based on the similarities of assigned nodes within a neighborhood. The higher the number of similar nodes assigned to each other within a particular neighborhood, the higher the probability that these assignments are correct.

Let us introduce the set $C \subseteq V_S \times V_T$, whose elements denote node correspondences between G_S and G_T , computed in Sec. 6.1.3.3. The set C is defined by an injective function $f : V_S \rightarrow V_T$, meaning that every $n_s \in V_S$ is mapped to at most one element of V_T , and $C \in \mathcal{U}$, with \mathcal{U} being the set of all possible correspondences configurations. We also define the topological neighborhood of a particular node n with radius δ , $N(n, \delta)$. The assignment energy $E : \mathcal{U} \rightarrow \mathbb{R}$ of a particular correspondence configuration between G_S and G_T is:

$$E(C) = \sum_{n_s \in V_S, n_t \in V_T, (n_s, n_t) \in C} l(n_s, n_t, C) \quad (6.12)$$

$$\begin{aligned}
l(n_s, n_t, C) = & q_{n_t n_s} + \sum_{n_j \in N(n_s, \delta_s), n_i \in N(n_t, \delta_t), (n_i, n_j) \in C} q_{n_i n_j} \\
& - \sum_{n_j \in N(n_s, \delta_s), n_i \notin N(n_t, \delta_t), (n_i, n_j) \in C} q_{n_i n_j} \\
& - \sum_{n_j \notin N(n_s, \delta_s), n_i \in N(n_t, \delta_t), (n_i, n_j) \in C} q_{n_i n_j} \quad (6.13)
\end{aligned}$$

where $l : V_S \times V_T \times \mathcal{U} \rightarrow \mathbb{R}$ denotes the local assignment energy between two nodes. The local assignment energy between two nodes n_s and n_t increases with the increasing number of nodes in their neighborhoods that are assigned to each other.

Inserting a new assignment between nodes n_s and n_t , causes the assignment energy $E(C)$ to vary according to:

$$\begin{aligned}
\delta(n_s, n_t, C) = & q_{n_t n_s} + \sum_{n_j \in N(n_s, \delta_s), n_i \in N(n_t, \delta_t), (n_i, n_j) \in C} (q_{n_t n_s} + q_{n_i n_j}) \\
& - \sum_{n_j \in N(n_s, \delta_s), n_i \notin N(n_t, \delta_t), (n_i, n_j) \in C} (q_{n_t n_s} + q_{n_i n_j}) \\
& - \sum_{n_j \notin N(n_s, \delta_s), n_i \in N(n_t, \delta_t), (n_i, n_j) \in C} (q_{n_t n_s} + q_{n_i n_j}) \quad (6.14)
\end{aligned}$$

while the variation caused by an assignment removal is given by $-\delta(n_s, n_t, C)$.

An iterative greedy optimization procedure is employed for the optimization of an initial correspondences configuration C_0 . The insertion or removal of a node correspondence is performed in every iteration. The operation (node insertion or removal) chosen is the one that increases $E(C)$ the most in the current correspondence configuration. Let us define the node correspondence operations $O = \{-1, 0, 1\}$, with -1 denoting a correspondence removal, 1 an insertion, and 0 a void operation. We also define the functions $c_S : V_S \times \mathcal{U} \rightarrow \{V_T, \emptyset\}$ and $c_T : V_T \times \mathcal{U} \rightarrow \{V_S, \emptyset\}$, which provide the correspondent of a particular node in the current correspondence configuration, as follows:

$$\begin{aligned}
c_S(n_s, C) &= \begin{cases} n_t & \exists n_t \in V_T ((n_s, n_t) \in C) \\ \emptyset & \text{otherwise} \end{cases} \\
c_T(n_t, C) &= \begin{cases} n_s & \exists n_s \in V_S ((n_s, n_t) \in C) \\ \emptyset & \text{otherwise} \end{cases} \quad (6.15)
\end{aligned}$$

We can now define the function $\Delta : O \times V_S \times V_T \times \mathcal{U} \rightarrow \mathbb{R}$, which provides the signed energy variation of the insertion or removal of a

correspondence between a node in V_S and a node in V_T , if the operation is possible in the current correspondences configuration, and $-\infty$ otherwise. The function $\Delta(o, n_s, n_t, C)$ also considers the possibility of an assignment insertion between nodes that are already assigned, meaning that already existing assignments must be removed prior to the insertion of the new one. The definition of $\Delta(o, n_s, n_t, C)$ is as follows:

$$\Delta(o, a, b, C) = \begin{cases} -\delta(n_s, n_t, C) & (o = -1) \\ & \wedge((n_s, n_t) \in C) \\ \delta(n_s, n_t, C) & (o = 1) \\ & \wedge(c_S(n_s, C) = \emptyset) \\ & \wedge(c_T(n_t, C) = \emptyset) \\ \delta(n_s, n_t, C) & (o = 1) \\ -\delta(n_s, c_S(n_s, C), C) & \wedge(c_S(n_s, C) \neq \emptyset) \\ & \wedge(c_T(n_t, C) = \emptyset) \\ \delta(n_s, n_t, C) & (o = 1) \\ -\delta(c_T(n_t, C), n_t, C) & \wedge(c_S(n_s, C) = \emptyset) \\ & \wedge(c_T(n_t, C) \neq \emptyset) \\ \delta(n_s, n_t, C) & (o = 1) \\ -\delta(n_s, c_S(n_s, C), C) & \wedge(c_S(n_s, C) \neq \emptyset) \\ -\delta(c_T(n_t, C), n_t, C) & \wedge(c_T(n_t, C) \neq \emptyset) \\ -\infty & \text{otherwise} \end{cases} \quad (6.16)$$

An operation in a particular correspondences configuration that increases $E(C)$ the most (it may not be unique) is found by the function $n : \mathcal{U} \rightarrow \mathcal{O} \times V_S \times V_T$:

$$n(C) = \begin{cases} \arg \max_{o \in \mathcal{O}, n_s \in V_S, n_t \in V_T} & \max_{o \in \mathcal{O}, n_s \in V_S, n_t \in V_T} \\ \Delta(o, n_s, n_t, C) & \Delta(o, n_s, n_t, C) > 0 \\ (0, \emptyset, \emptyset) & \text{otherwise} \end{cases} \quad (6.17)$$

We can observe that the sets \mathcal{O} , V_S and V_T are finite, and that $\Delta(o, n_s, n_t, C)$ is a primitive recursive function, since it is composed of sums, membership tests in finite sets and case decisions, and therefore $\Delta(o, n_s, n_t, C)$ is total. As a result of these observations,

we notice that $\arg \max_{o \in O, n_s \in V_S, n_t \in V_T} \Delta(o, n_s, n_t, C)$ can be computed in finite time, as can $n(C)$. The time complexity of $n(C)$ is upper bounded by $|V_S| \cdot |V_T|$ computations of $l(n_s, n_t, C)$, which is cubic at most (considering that the neighborhood sets are already known). This results in a maximal complexity of $O(x^5)$, where $x = \max(|V_S|, |V_T|)$. However, as shown further on in the text, the local assignment energies $l(n_s, n_t, C)$ can be computed iteratively, allowing the computation of $n(C)$ in $O(x^2)$ (with a pre-processing stage upper bounded by $O(x^5)$).

After having chosen the operation to execute, the set C must be modified accordingly. For this purpose, the operation execution function $e : O \times V_S \times V_T \times \mathcal{U} \rightarrow \mathcal{U}$ is defined. It performs the required assignments considering that the nodes n_s and n_t may already have correspondences. In this case, the existing assignments are previously removed and the new one is inserted after:

$$e(o, n_s, n_t, C) = \begin{cases} C - \{(n_s, n_t)\} & (o = -1) \\ & \wedge ((n_s, n_t) \in C) \\ C \cup \{(n_s, n_t)\} & (o = 1) \\ & \wedge (c_S(n_s, C) = \emptyset) \\ & \wedge (c_T(n_t, C) = \emptyset) \\ (C - \{(n_s, c_S(n_s, C))\}) \cup \{(n_s, n_t)\} & (o = 1) \\ & \wedge (c_S(n_s, C) \neq \emptyset) \\ & \wedge (c_T(n_t, C) = \emptyset) \\ (C - \{(c_T(n_t, C), n_t)\}) \cup \{(n_s, n_t)\} & (o = 1) \\ & \wedge (c_S(n_s, C) = \emptyset) \\ & \wedge (c_T(n_t, C) \neq \emptyset) \\ ((C - \{(n_s, c_S(n_s, C))\}) - \{(c_T(n_t, C), n_t)\}) \cup \{(n_s, n_t)\} & (o = 1) \\ & \wedge (c_S(n_s, C) \neq \emptyset) \\ & \wedge (c_T(n_t, C) \neq \emptyset) \\ C & \text{otherwise} \end{cases} \quad (6.18)$$

The greedy optimization executes until no more operations exist that increase $E(C)$ by more than a given threshold value $t \in \mathbb{R}^+$. The method for the post-processing of an initial correspondence set between two graphs is summarized in Alg. 6.1.

Algorithm 6.1 Post-processing of an initial correspondence set C_0 between two input graphs G_A and G_B . The method is performed until the assignment energy variation is smaller than a given threshold value t .

```

1 Input:  $G_S, G_T, C_0, t$ 
    $(o, n_s, n_t) \leftarrow n(C_0)$ 
3  $C \leftarrow C_0$ 
  while  $\Delta(o, n_s, n_t, C) > t$ 
5    $C \leftarrow e(o, n_s, n_t, C)$ 
    $(o, n_s, n_t) \leftarrow n(C)$ 

```

We now show that the post-processing method will always converge to at least a local maximum of $E(C)$ for every combination of input graphs G_S and G_T , and correspondence sets C . As V_S and V_T are finite, so are all $C_i \in \mathcal{U}$ and also \mathcal{U} , because there is only a finite amount of combinations between V_S and V_T . It is also possible to sort all C_i in increasing order of $E(C_i)$, and to connect each C_i that can be reached with a single operation. In this way we have at one side of every particular C_i the correspondences configurations that can be reached with negative assignment energy variation, and on the other side the ones that can be reached with positive energy variation (Fig. 41). As the optimization algorithm only moves towards positive energy variation, it will reach a configuration C that can not be improved by a single operation at some point. This is due to the fact that the amount of possible C_i is finite. However, the final correspondence configuration reached may not be the global maximum of $E(C)$, as in this case some moves towards negative energy variation might be necessary.

Implementation As the computation of the local assignment energies $l(n_s, n_t, C)$ is the most expensive part in our method, we avoid executing it repeatedly by computing it once for all nodes with respect to the initial correspondence configuration. After that we just update it iteratively as new node correspondences are inserted or removed. We start our implementation with a pre-processing stage, where all neighborhood sets $N(n, \delta)$ are computed, and the sets $C^S = \{n_s \in V_S | c_S(n_s, C_0) \neq \emptyset\}$ and $C^T = \{n_t \in V_T | c_T(n_t, C_0) \neq \emptyset\}$ are initialized. A matrix $L = [l_{n_s n_t}]_{|V_S| \times |V_T|}$, with $l_{n_s n_t} = \delta(n_s, n_t, C_0)$, is also computed for every node pair $(n_s, n_t) \in V_S \times V_T$. The pre-processing stage has maximal time complexity of $O(\chi^5)$, and must be computed only once.

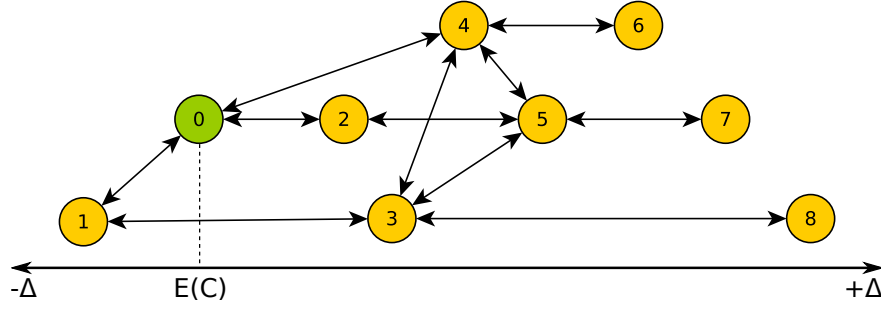


Figure 41: Graphical representation of the correspondences configuration between two graphs. Every node represents a correspondences configuration C_i , while the arrows represent configurations that are related by a single node correspondence operation. The nodes are sorted in increasing order of assignment energy ($E(C)$). As our optimization algorithm only moves towards maximal positive energy variation, convergence is guaranteed. Starting at configuration C_0 , configuration C_6 will be the final one. However, the global maximum (C_8), may only be reached with moves towards negative energy variation.

The operation with highest $\Delta(o, n_s, n_t, C)$ can be found by iterating through L , with a time complexity of $O(x^2)$. If the chosen operation involves a first removal of assignments before inserting a new one, each step of this operation is considered as a separate step. In every iteration of the optimization algorithm, after inserting or removing an assignment between (n_s, n_t) , the sets C^S and C^T must be updated accordingly. Also, the local assignment energies of the nodes in the neighborhoods of n_s and n_t must be updated:

- In the case of an insertion operation: $\forall n_i \in N(n_s, \delta_s), n_j \in N(n_t, \delta_t) (l_{n_i n_j} = l_{n_i n_j} + q_{n_t n_s} + q_{n_j n_i}), \forall n_i \in N(n_s, \delta_s), n_j \notin N(n_t, \delta_t), n_j \neq n_t (l_{n_i n_j} = l_{n_i n_j} - q_{n_t n_s} - q_{n_j n_i}), \forall n_i \notin N(n_s, \delta_s), n_j \in N(n_t, \delta_t), n_i \neq n_s (l_{n_i n_j} = l_{n_i n_j} - q_{n_t n_s} - q_{n_j n_i})$;
- In the case of a removal operation: $\forall n_i \in N(n_s, \delta_s), n_j \in N(n_t, \delta_t) (l_{n_i n_j} = l_{n_i n_j} - q_{n_t n_s} - q_{n_j n_i}), \forall n_i \in N(n_s, \delta_s), n_j \notin N(n_t, \delta_t), n_j \neq n_t (l_{n_i n_j} = l_{n_i n_j} + q_{n_t n_s} + q_{n_j n_i}), \forall n_i \notin N(n_s, \delta_s), n_j \in N(n_t, \delta_t), n_i \neq n_s (l_{n_i n_j} = l_{n_i n_j} + q_{n_t n_s} + q_{n_j n_i})$.

At the end, a clean-up procedure can be performed, eliminating all assignments with $l(n_s, n_t, C)$ smaller than a given threshold value.

6.1.4 Evaluation and results

For the evaluation of the region-based approach, we have performed experiments to assess the following:

- The applicability of the graph matching algorithm for the registration of real data (Sec. 6.1.4.1). As we have already established that mesh segmentation is crucial to obtain correct registration results (Sec. 6.1), the applicability of the graph matching algorithm to the registration of real data is performed using structures whose representation by graphs is already implicit: These are the anatomical trees, such as the bronchial or vessel systems. This allows us to evaluate the robustness of graph matching without being influenced by the quality of the segmentation.
- The specificity of the graph matching algorithm for the registration of meshes and sub-meshes of different sizes and noise levels (Sec. 6.1.4.2).
- The improvement of results that are provided by graph matching post-processing (Sec. 6.1.4.3).

6.1.4.1 Matching of anatomical trees

Finding correspondences between anatomical trees may be challenging due to noisy data, motion, artifacts, or problems associated with the extraction method, such as missing or false branches. Several tree matching approaches have been investigated in the literature, which has been reviewed by Metzen et al. [188]. Graham and Higgins [119, 120] presented an extensive theoretic framework for finding correspondences in anatomical trees and explicitly addressed the above mentioned distortions. They applied a dynamic programming algorithm to map both trees onto a common one using similarity scores between the nodes and two kinds of deformation models to capture distortions. Unfortunately, a comprehensive evaluation of the method does not exist. Furthermore, the authors use parallelism between edges as a similarity measure, which makes the method dependent on the position of the trees, *i.e.*, the pose of the patient during image acquisitions must be identical (rotations are not allowed).

Here, tree matching is performed in a similar manner as for surface matching (Sec. 6.1), with two main differences: (1) No segmentation is required to construct the graphs, as trees can be naturally represented by them (each fork represents a node, while branches represent edges); (2) Node attributes are computed based on tree properties, such as branch diameter, for example. A schematic representa-

tion of the proposed method is shown in Fig. 42. Topological similarity scoring was performed using the direct scoring method. Only the elimination of false correspondences is performed was carried out as a post-processing step.

TREE PROPERTIES Six properties were chosen, which do not require knowledge about the absolute position of a particular node, making the matching independent of position: Diameter (in mm; Fig. 43a), internal distance to the root (in mm; Fig. 43b), node level in the tree (Fig. 43c), maximum and minimum angle between the edges that connect the children (Fig. 43d), and the angle between the (virtual) line connecting the root node with the node under consideration and the first branch (Fig. 43e).

ASSESSMENT OF THE ACCURACY OF THE GRAPH MATCHING ALGORITHM BY ANATOMICAL TREE MATCHING For the evaluation of our method, we used 15 CT data sets of porcine lungs. Each data set was composed of an expiration and an inspiration tree. The volumes were segmented and the trees were pruned. Correspondences between both trees of all data sets were set manually and used as ground truth.

Five data sets were used for the estimation of the Gaussian kernel widths and descriptor distance thresholds. These values were used to evaluate the method based on the other 10 data sets. The results obtained are shown in table 2. A total of 60% of all nodes that could be matched, according to the ground truth, were matched correctly (*assignable, correct*) while 3% were matched incorrectly (*assignable, wrong*), 37% were not matched (*assignable, not matched*). Within the set of nodes that could not be matched according to ground truth data, 16% were incorrectly matched (*unassignable, wrong*) while 84% were correctly left unmatched (*unassignable, correct*). The time required for the creation of both trees, including the computation of properties, was smaller than 10^{-3} milliseconds for all data sets on a 2.4 GHz Intel machine. The matching stage, including topological and properties similarity scoring, assignment and reduction of false matches, took less than one second on average.

6.1.4.2 Graph matching for surface registration

In order to evaluate the proposed correspondence search method we performed three evaluation studies on meshes with different surface characteristics (Fig. 44). In the first study, we evaluated the performance of our algorithm with respect to the size of the submesh. In

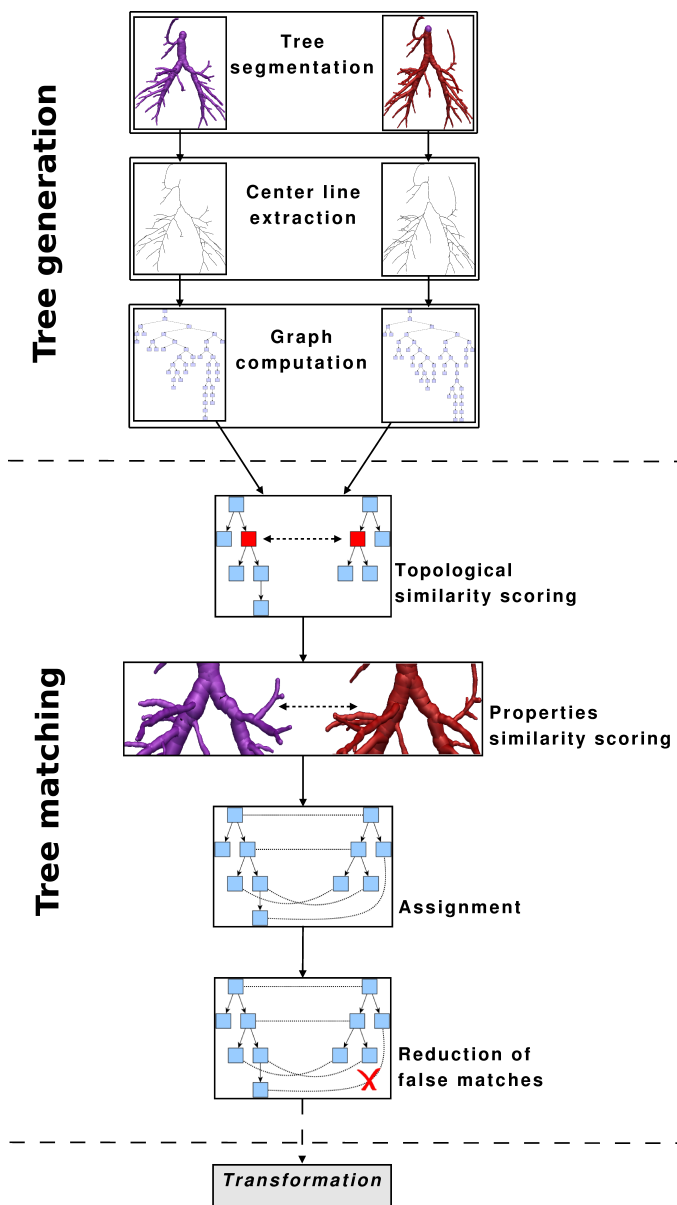


Figure 42: Matching of anatomical trees.

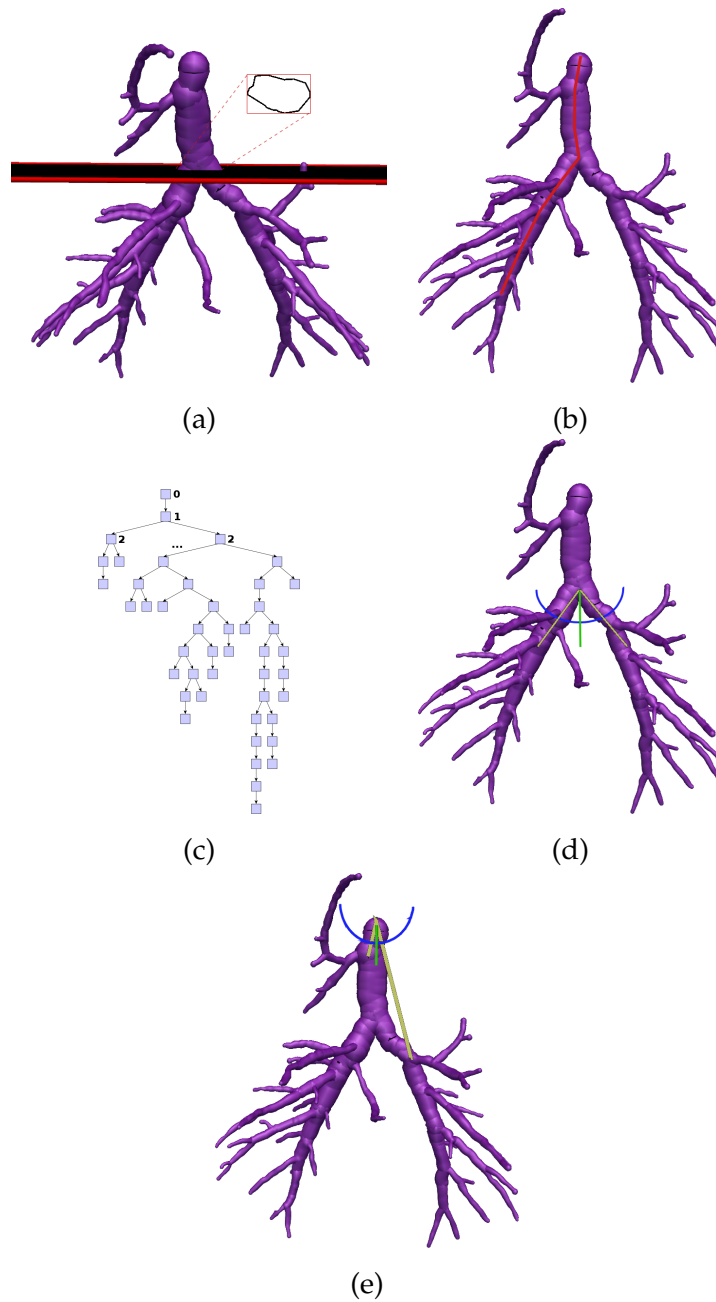


Figure 43: Tree properties: (a) diameter; (b) internal distance to the root; (c) node level in the tree; (d) maximum and minimum angle between the edges that connect the children; (e) angle between the (virtual) line connecting the root node with the node under consideration and the first branch.

Test data set	1	2	3	4	5	6	7	8	9	10	Avg. (%)
Nodes	28&28	23&20	25&16	44&29	52&45	54&35	25&24	50&44	22&17	56&43	
Assignable	24	19	15	27	42	35	19	36	15	41	91±7
CORRECT	20	11	9	13	30	22	9	19	9	22	60±11
WRONG	0	2	0	2	1	0	0	2	0	2	3±4
NOT MATCHED	4	6	6	12	11	13	10	15	6	17	37±10
Unassignable	4	1	1	2	3	0	5	8	2	2	9±7
CORRECT	4	1	0	2	2	-	5	7	2	2	84±33
WRONG	0	0	1	0	1	-	0	1	0	0	16±33
Time (ms)	20	> 10 ⁻³	100	330	170	1810	20	80	20	210	

Table 2: Results for the anatomical tree matching evaluation. *Nodes* show the number of nodes in the two corresponding trees starting with bigger ones. *Assignable* refers to the number of nodes in the smaller tree that have a corresponding node in the bigger tree (according to the ground truth assignment). *Unassignable* refers to the number of nodes in the smaller trees that do not have a corresponding one. In this case, *correct* shows the number of nodes that were correctly left unassigned, while *wrong* shows the total of nodes that were assigned to any other one. The times shown are the durations of topological and properties similarity scoring, assignment and reduction of false matches. The averages are given in percentages of the total in their categories (*assignable* or *unassignable*), except for the totals in the categories themselves, which are given in percentages of the smaller tree.

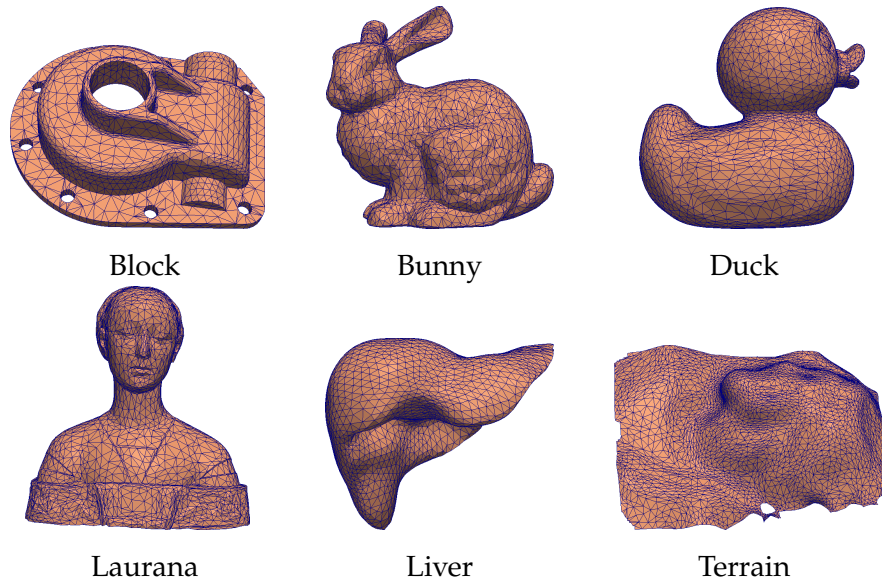


Figure 44: The six reference surfaces used for the evaluation of the proposed method. All meshes (with exception of the liver), are available at the AIM@SHAPE Shape Repository (<http://shapes.aim-at-shape.net>) and were decimated to approximately 10000 faces.

a second study, we assessed the influence of the shape descriptor on our method, by repeating the first experiment with an alternative mesh segmentation approach. In this other approach, surfaces are partitioned based on curvature classes as described in [33], instead of using continuous curvature measures such as the shape index and curvedness. Finally, we tested the method on noisy data in order to evaluate the noise tolerance of the novel algorithm. In all studies, topological similarity scoring was performed with the direct scoring method, and only the elimination of false correspondences was performed in the post-processing stage. An alignment between the shapes was obtained by the computation of a transformation that matched the centroids of the assigned regions in a least squares sense according to the algorithm of Horn [134].

STUDY 1 To evaluate the performance of the proposed correspondence search algorithm, the following experiment was performed for each reference surface shown in Fig. 44: For each integer i in $[1, 50]$, 500 random submeshes (samples) with an area making up $i\%$ of the area of the reference mesh were generated using a region growing method. Each of the samples was translated and rotated with a random rigid transformation. The proposed correspondence search algo-

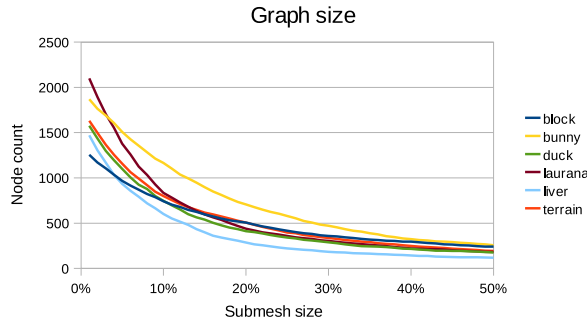


Figure 45: Number of nodes in the graphs of the reference meshes as a function of the submesh size.

rithm was applied to initially realign the submesh with the reference mesh. Subsequently, ICP was performed to adjust the positioning. To assess the accuracy of the alignment, the percentage of correctly classified regions and the percentage of incorrectly classified regions was determined. Furthermore, the quality of the final match (after ICP) was evaluated by calculating all distances between the transformed submesh vertices and their corresponding vertices in the reference mesh. The match was considered correct, if all distances were smaller than 10^{-6} (note that all meshes were scaled to fit into the unit box).

The parameters of the proposed method were chosen empirically on a set of liver meshes as follows: Shape index and curvedness kernel widths were set to 0.1, and shape index and curvedness kernel thresholds to 0.5. For the elimination of false positives, the neighborhood radii were set to 2 for both graphs, while the minimal number of node assignments was set to 8. The curvedness threshold was set according to the submesh size to be matched: The smaller the submesh size, the smaller the threshold. This way, a reduction of graph size and thus a reduction of processing time could be achieved for relatively big submeshes. By default, the curvedness threshold was linearly increased from 0.1 (corresponding to a submesh size of 1%) to 2 (corresponding to 50%). In the case of Laurana, the interval was set to $[0.1, 3]$ because this surface showed more curvature variations than the others. The resulting sizes of the graphs are shown in Fig. 45.

STUDY 2 To evaluate the influence of the shape descriptor on the presented correspondence search method, we repeated the first experiment with an alternative mesh segmentation method. As initially proposed by Besl and Jain [33], the surfaces were partitioned based on curvature *classes*, identified by the sign of the Gaussian and mean curvatures. As this segmentation method does not depend on any

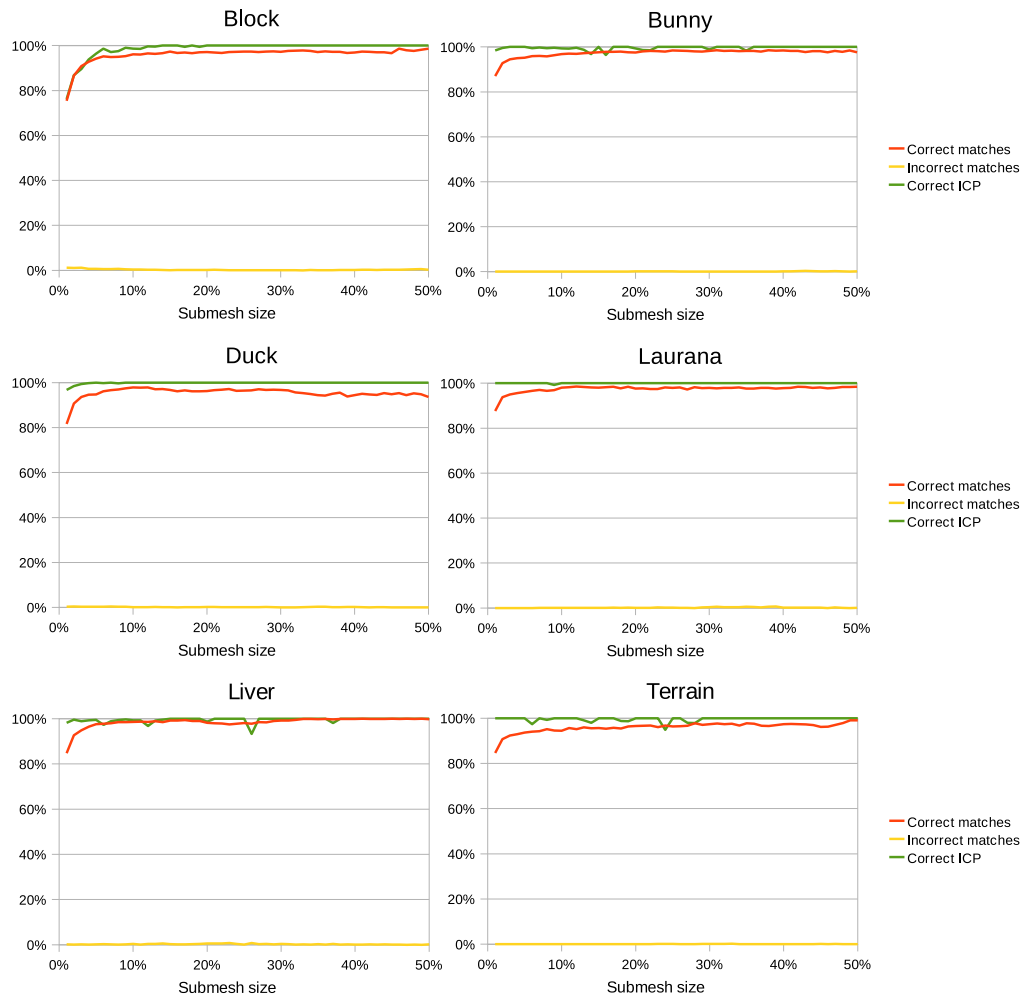


Figure 46: Results of the surface matching experiment described in study 1. For each reference mesh, the mean percentage of correct matches and incorrect matches after the initial alignment averaged over 500 samples is shown as a function of the submesh size. *Correct ICP* represents the quality of the final match (after ICP).

parameters, the reference mesh graph size was the same for all experiments (230, 217, 132, 275, 131 and 142 nodes for the shapes Block, Bunny, Duck, Laurana, Liver and Terrain, respectively).

STUDY 3 In the case of noisy data, the parameters of the segmentation algorithm must be chosen carefully and in an application-specific manner. To illustrate this procedure, we acquired medical image data from seven different patients and extracted the corresponding liver meshes. Two livers were used to empirically optimize the mesh segmentation parameters for each submesh size $i \in \{10\%, 20\%, 30\%, 40\%, 50\%\}$. The smaller the submesh, the smaller the tolerance in the mesh segmentation, thus yielding bigger graphs. Bigger graphs increase the number of nodes that are common to both meshes and helps identifying correct correspondences.

The Gaussian kernel parameters were set to 0.05 for the width and 0.15 for the threshold, both for shape index and curvedness. The shape index thresholds for both, submesh and reference mesh, were set to 0.15, 0.15, 0.12, 0.12 and 0.1 for submesh sizes of 50%, 40%, 30%, 20% and 10%, respectively. The curvedness thresholds were set in the same way to 0.15, 0.13, 0.12, 0.1, and 0.03 respectively. For the elimination of false positives, the neighborhood radii were set to 2 for both graphs, while the minimal number of node assignments was set to 4.

For each submesh size, a set of 50 random submeshes was extracted. A random vector with direction and magnitude drawn randomly from the interval $[0, (30\% \bar{d})]$, where \bar{d} is the mean edge length of the reference mesh, was added to each vertex to simulate noise in the data. The submeshes were then smoothed through the method proposed in [143].

Unlike the previous experiments, the segmentation of the submesh was potentially different from the segmentation of the corresponding area in the reference mesh due to the noise in the data. Hence, there was generally no isomorphism in the graphs, making an evaluation of the correctly assigned nodes impossible. However, vertex correspondences were known, and we thus assessed the quality of the alignment by computing the percentage of submesh vertices that were assigned to their corresponding vertices in the reference mesh after the final iteration of the ICP.

SUMMARY OF THE RESULTS The results of our evaluation are shown in Figs. 46, 47 and 48. Whenever a common *clique* between the graphs existed (fig. 46), *i.e.*, there was no noise added to the data, the percentage of correct and incorrect node assignments averaged

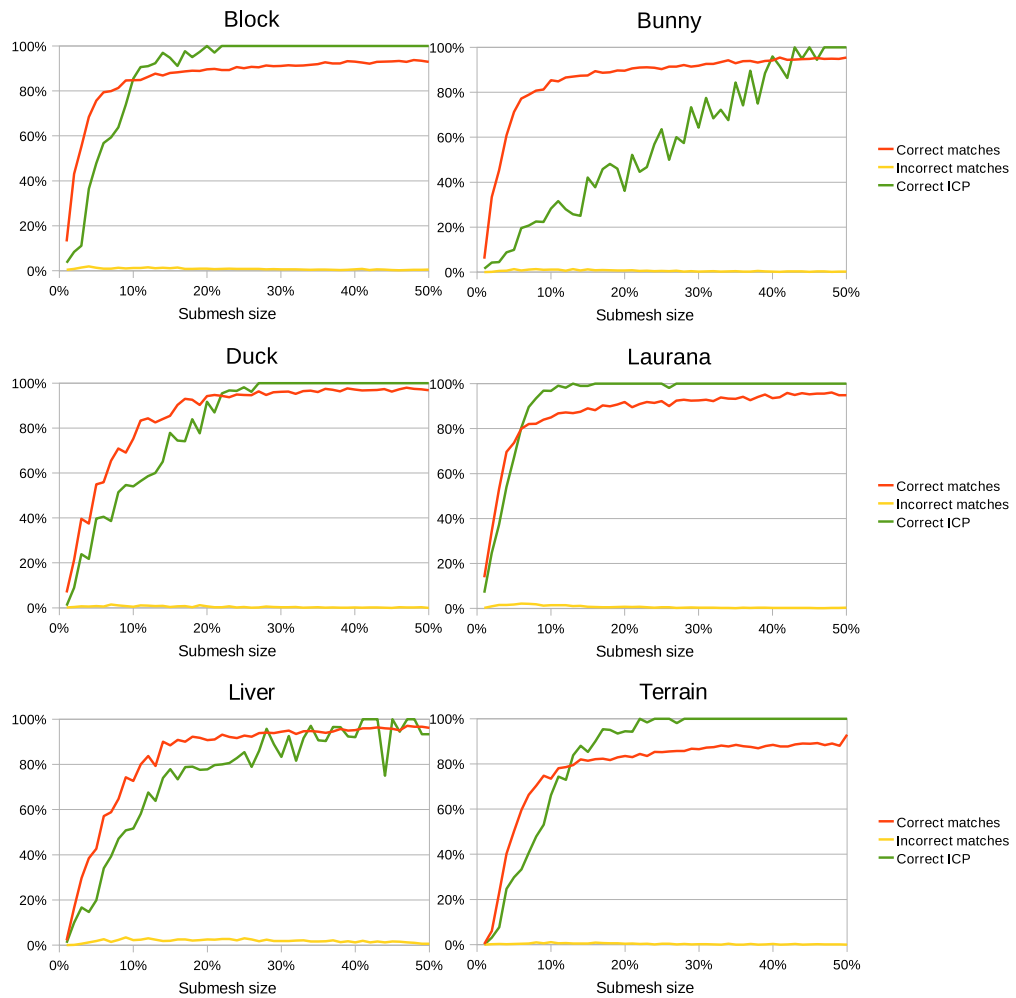


Figure 47: Results of the surface matching experiment described in study 2. The mesh segmentation was performed using curvature classes as opposed to shape index and curvedness. For each reference mesh, the mean percentage of correct matches and incorrect matches after the initial alignment averaged over 500 samples is shown as a function of the submesh size. *Correct ICP* represents the quality of the final match (after ICP).

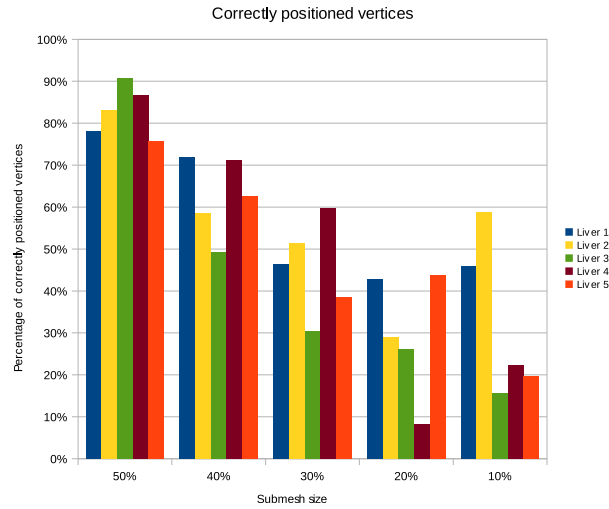


Figure 48: Results of the experiment described in study 3, where the proposed correspondence search method was evaluated with noisy data. The mean percentage (averaged over 50 samples) of submesh vertices that were assigned to their corresponding vertices in the reference mesh after the final iteration of the ICP is shown for all liver meshes as a function the submesh size.

over all six objects ranged from $83.5\% \pm 6.1\%$ and $0.3\% \pm 0.3\%$ respectively (submesh size: 1%) to $97.9\% \pm 3.1\%$ and $0.1\% \pm 0.1\%$ respectively (submesh size: 50%). The correct transformation was found in almost all cases ($94.9\% \pm 11.9\%$ for submesh size: 1%, $100.0\% \pm 0.0\%$ for submesh size: 50%). Depending on the object and the submesh size, processing times for the correspondence search including mesh segmentation and graph generation ranged from 30 ms to 12 s. All processing times were measured using a non-threaded 2.4 GHz Intel machine.

According to Figs. 46 and 47, the segmentation based on curvature classes yields significantly worse results than the segmentation based on shape index and curvedness. In the earlier case, we obtained a percentage of correct and incorrect node assignments ranging from $7.0\% \pm 6.7\%$ and $0.1\% \pm 0.2\%$, respectively (submesh size: 1%), to $94.9\% \pm 1.9\%$ and $0.3\% \pm 0.3\%$, respectively (submesh size: 50%). For small submeshes ($\leq 10\%$) the ICP did not converge into the global optimum. The correspondences search processing time ranged from 1 millisecond to 2.6 seconds.

Regarding noisy data (fig. 48), the mean percentage averaged over 5 livers of submesh vertices that were assigned to their corresponding vertices in the reference mesh, after the final iteration of the ICP, ranged from $32.5\% \pm 21.7\%$ (submesh size: 1%) to $82.9\% \pm 7.5\%$ (sub-

mesh size: 50%). Processing times for the correspondences search ranged from 29 to 100 seconds.

6.1.4.3 Post-processing using greedy optimization

In order to assess the accuracy of the proposed method for post-processing using greedy optimization (Sec. 6.1.3.4), we performed a quantitative evaluation using randomly generated graphs in addition to a qualitative evaluation with real scanned surfaces. To score topological similarities, the *SoftAssign* algorithm (Sec. 6.1.3.2) was employed.

QUANTITATIVE EVALUATION In the quantitative evaluation, 100 attributed small-world graphs with 200 nodes were randomly generated. Attributes were drawn randomly from the interval $[0, 1]$. From each of them, 50 subgraphs for each size in $\{100\%, 90\%, \dots, 10\%\}$ of the original graph were arbitrarily extracted. The performed experiment consisted of: (1) Applying structural and attribute errors to the subgraphs; (2) Randomly generating a new attributed small-world graph with the modified subgraph in it, so that both original and new graphs had approximately the same size; (3) Matching the newly generated graph with the original one using *SoftAssign* followed by the post-processing algorithm.

Attribute errors are randomly drawn numbers which were added to the original attributed graph. In this experiment, three classes of attribute errors were used, with maximal errors in $\{0, 0.2, 0.4\}$. Structural errors are the probability of each node and edge in the subgraph to be removed. Three classes of structural errors were employed in this experiment, with probabilities in $\{0\%, 20\%, 40\%\}$.

Once matching was complete, the results obtained with the *SoftAssign* algorithm and the ones obtained after application of the post-processing algorithm were compared with the ground truth. Nodes contained in the extracted subgraph must be correctly assigned to their corresponding nodes in the original graph, while the other nodes must be left unassigned.

The results of the evaluation are shown in Fig. 49. As both input graphs have approximately the same size, we denote the number of nodes that are common to both graphs as *intersection size*, i.e., the size of the extracted subgraph in percent of the original graph. By using post-processing, the percentage of correct assignments increased from 0% (Intersection size: 100%; Attribute error: 0%; Structural error: 0%) to 77.3% (Intersection size: 80%; Attribute error: 40%; Structural error: 0%) of the graph size. Processing times ranged from 2 to 3.9 s for the

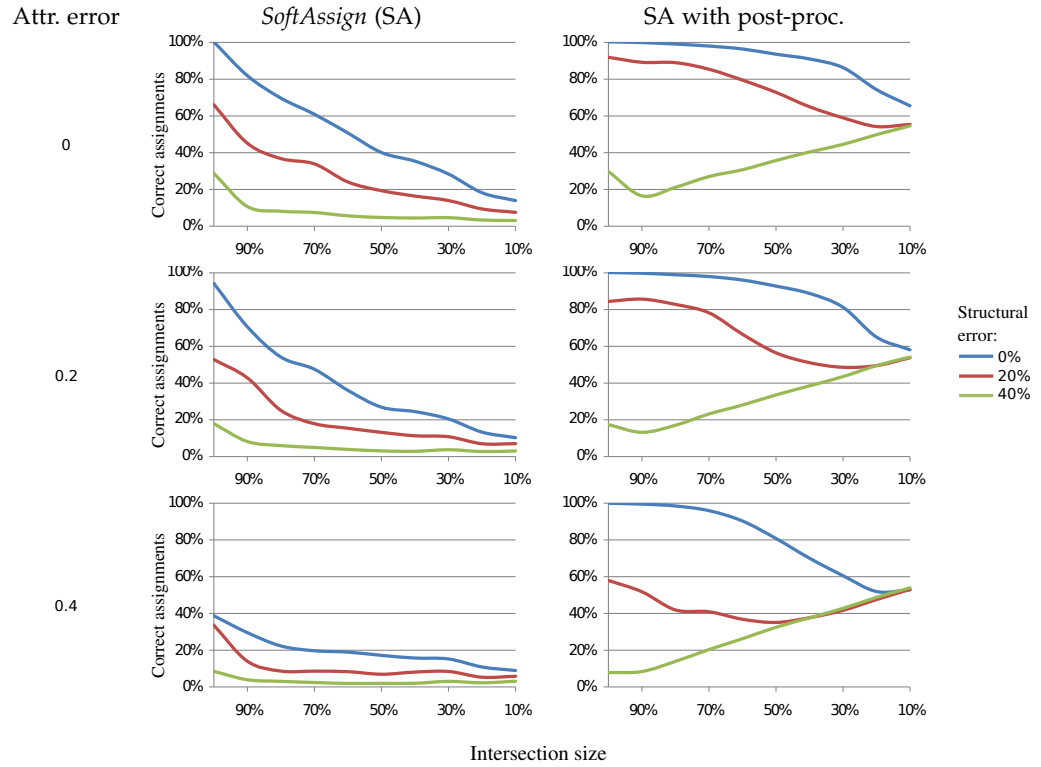


Figure 49: Results for the evaluation of the post-processing algorithm using greedy optimization with simulated data. In this evaluation, 100 attributed small-world graphs with 200 nodes were randomly generated. From these graphs, 50 submeshes were extracted. These submeshes were included into other randomly generated graph, so that both graphs have approximately the same size. Additionally, structural and attribute errors were applied. Both graphs were matched and the results before and after post-processing were compared with the ground truth. *Intersection size* denote the size of the intersection between the graphs, *i.e.*, the size of the extracted subgraph. The results were averaged for each intersection size. *Correct assignments* are given by the percentage of nodes in the second graph that were correctly assigned to its corresponding one in the original graph, or were correctly left unassigned if there were no corresponding node.

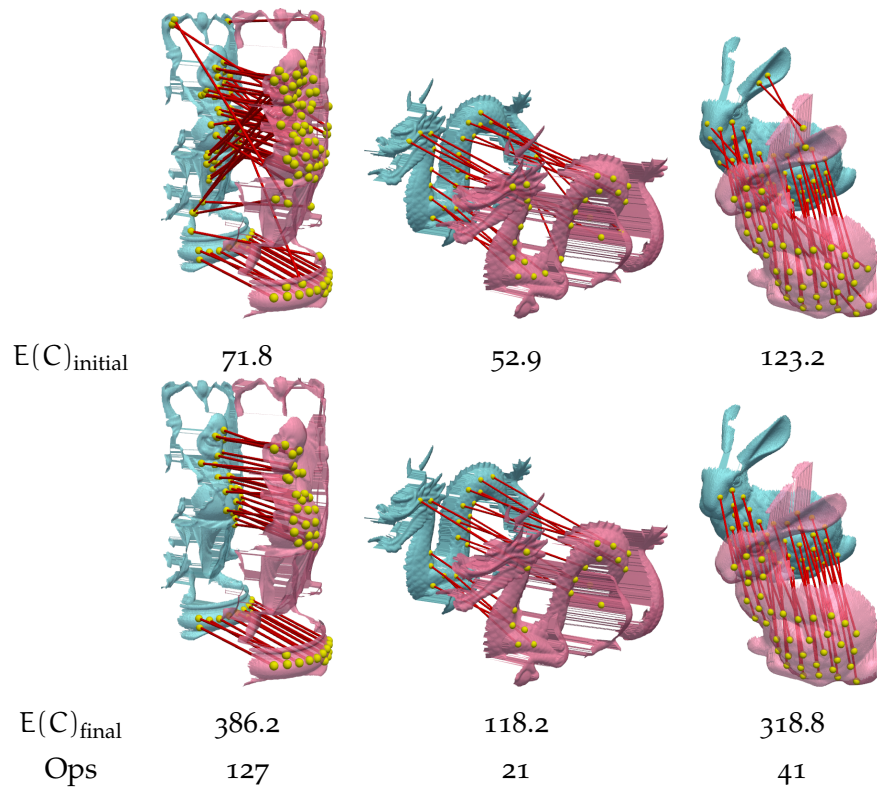


Figure 50: Results of the evaluation of post-processing using greedy optimization with real scanned data. Shown in the images are the correspondences found with *SoftAssign* (top row) and after post-processing (bottom row). Also the assignment energy ($E(C)$) before and after post-processing is shown. *Ops* denotes the amount of assignment operations (insertion or removal) that were performed during post-processing.

SoftAssign, and from 24 to 125 ms for the post-processing, in a non-threaded 2.4 GHz Intel machine.

QUALITATIVE EVALUATION A qualitative evaluation with real scanned data was performed using freely available models from the *Stanford 3D Scanning Repository* (<http://graphics.stanford.edu/data/3Dscanrep/>). The meshes were segmented using mean curvature as surface descriptor. Shape index and curvedness [155] were employed as surface descriptors for the matching phase. Post-processing was performed with the vertices of the 2-ring neighborhoods and threshold value set to 0.1. The results for this evaluation are shown in Fig. 50. In average, the assignment energy increased about 240% with the application of the post-processing algorithm, taking about 63 assignment operations (insertions or removals).

6.1.5 Discussion

We show that the region-based approach proved to be a viable alternative for surface matching. The graph matching algorithm showed to be highly accurate for matching incomplete and noisy data. In the case of surface matching however, with the graph representation not being implicit to the data, thus requiring a previous segmentation of the meshes in order to make the computation of an adjacency graph possible, the results are tied to the consistency of this segmentation. As expected, the more noise and distortions between the surfaces, the more inconsistent the segmentations. As high noise and distortions occur in intra-operative environments, the successful application of the region-based approach crucially depends on the robustness of the segmentation algorithm. The design of a segmentation algorithm that fulfills all criteria of intra-operative registration is not trivial. This might prevent the successful application of the proposed region-based approach in the intra-operative environment. Even though post-processing showed to increase the robustness of the matching algorithm, it does not perform well given very degenerated graph structures (structural errors).

Note, however, that, in the case of the *direct scoring method* for graph topology scoring (Sec. 6.1.3.2), as neighborhoods similarities are scored and propagated throughout the graph, *i.e.*, each node similarity score includes not only direct similarity, but also the neighborhood similarity, we are able to compute a set of correspondences by means of minimization of the *linear assignment problem* (LAP) error (see Sec. 3.3). When using the *SoftAssign* method, the graph matching problem is directly cast as a *quadratic assignment problem* (QAP), and is solved by means of relaxation of the assignment boolean constraints (See Sec. 3.4). In this case, the correspondence computation stage (Sec. 6.1.3.3) only has the effect of projecting the fuzzy (relaxed) assignment back to the boolean domain.

6.1.5.1 Matching of anatomical trees

According to our results (Sec. 6.1.4.1), the graph matching approach showed to be highly accurate and fast when registering noise and irregular data. Despite its simplicity, the proposed method for elimination of false matches has proven to be very efficient, keeping the rate of incorrect matches very low. However, after eliminating false matches, many nodes were left unassigned and thus did not contribute to the computation of a transformation. We put the results obtained with the presented method with the ones presented in other

state-of-the-art works. The matching rate and run-time of the proposed method are comparable to the results presented by Graham and Higgins [119, 120]. Unfortunately, the authors only presented the number of assigned nodes for four data sets, without discriminating correct and wrong ones. The run-time was only given for a single data set (5 seconds to match two trees with 341 and 131 branches respectively, using a 3.4 GHz Pentium). Compared to the work of Metzen et al. [188], our method yields a higher rate of correct matches (53% versus 19%²) while the rate of incorrect matches is comparable (4% vs. 2%). No time measurements were provided by the authors.

6.1.5.2 *Surface registration*

According to the evaluation presented (Sec. 6.1.4.2), the graph matching based method for surface registration is highly accurate, yielding a correct match in almost all cases for submeshes without noise that made up at least 5% of the size of the reference meshes used in the study. This held independently of the shape of the object and the initial position of the submesh. Even in the presence of noise, good matches were obtained for submeshes larger than 30% of the mesh size.

We are aware of the fact that we used a relatively simple method for mesh segmentation by combining a classical region growing approach with a surface descriptor that is rather unstable in the presence of noise. However, the good performance of our algorithm despite these conditions demonstrates the huge potential of the graph-based registration approach. On the other hand, the results obtained with curvature classes indicate that the performance significantly depends on the descriptor and segmentation technique chosen.

When mesh segmentation was based on curvature classes, the ICP did not converge to the correct solution, in general. This can be attributed to the fact that segmentation using curvature classes tends to create bigger regions. Hence, the regions at the boundaries of the submeshes can be considerably smaller than the corresponding regions in the reference mesh. As a consequence, the centroid of a subregion may have a relatively large geometrical distance to the centroid of the corresponding region in the reference mesh. Furthermore, the initial alignment may not be close enough to the reference solution in order to guarantee ICP convergence. To overcome this problem, a weighted least square approach could be applied to rigidly align the centroids

² Both correct and incorrect rates were calculated based on the results found in the paper. The rates represent the mean of the percentages of the smaller tree.

of corresponding regions in a way that the regions at the boundaries of the submesh are given less weight.

In the presence of noise, the parameters for the mesh segmentation play an important role. We obtained reliable ICP convergence for submeshes with sizes of 50% of the reference mesh. As expected, the accuracy decreased, with decreasing submesh sizes. However, when matching submeshes of size of 10%, the accuracy increased compared to the previous sizes. In this case, the curvedness threshold was set very low, thus decreasing the tolerance of the mesh segmentation and generating bigger graphs. Having bigger graphs, also increases the number of common nodes in both graphs, and reduces the amount of geometrical information in the regions. This allowed the region representatives to get closer to the represented data. Increasing the number of common nodes and bringing the representatives closer to the represented data increases the possibility of unique identification of regions. Unfortunately, if the same parameters had been used for the other submesh sizes, the processing times would have been significantly higher. Aggregating a lot of geometric information into single regions, as when segmenting the meshes in region classes or using higher thresholds, degraded the alignment for smaller submeshes.

Several issues remain to be addressed: Firstly, processing times varied considerably with the shape of the object and the submesh size. This can be attributed to the fact that the duration of the assignment computation depends crucially on the percentage of entries in the similarity matrix with values equal to $-\infty$, *i.e.*, on the count of possible assignments. In general, computation times could potentially be significantly decreased by a concurrent implementation (*e.g.* on the GPU).

The accuracy of the proposed correspondence search method in the presence of noise is very encouraging. Yet, several measures need to be taken to make the matching more robust. In general, matched regions will not be of the same size if the subsurface does not represent an exact instance of a part of the reference mesh. As our method only creates one-to-one assignments, several regions might be left unassigned, if two or more regions in one mesh correspond to the same one in the other. For this reason, the method should be further extended for multiple-to-one assignments.

6.1.5.3 *Post-processing using greedy optimization*

The post-processing method was evaluated with both simulated and real data (Sec. 6.1.4.3). With simulated data a significant increase in the percentage of correctly assigned nodes was observed. This in-

crease was particularly higher with higher attribute errors and lower structural errors. In the case of higher structural errors, we can observe that the percentage of correct assignments increases with decreasing intersection size. This is a rather non-intuitive behavior. An explanation for this behavior lies on the fact that the number of nodes with no correspondence is higher with smaller intersection sizes. As shown in a previous evaluation (Secs. 6.1.4.2 and 6.1.4.1), the elimination of incorrect assignments can be performed very effectively. As it is very unlikely that nodes with no correspondents have similar neighborhoods in the opposite graph, they get mostly negative local assignment energies, and their assignments are removed during the optimization.

In the evaluation using real scanned data, the increase of the assignment energy before and after the employment of the post-processing algorithm can be confirmed by a visual inspection of the correspondences. Using simulated data, we showed that our algorithm performs better with lower structural error. Better results can thus be achieved if the graphs representing both surfaces are structurally similar. The structure of the graphs directly depends on the employed mesh segmentation algorithm. Obtaining similar segmentations can be very challenging, especially when dealing with scanned data acquired from different angles. The choice for a particular segmentation algorithm must therefore be made very carefully and in an application specific manner.

As our post-processing method only requires graphs, the initial correspondence set and a node similarity matrix between both input graphs, the initial correspondence search stage may easily be adapted for different mesh segmentation and matching algorithms. Furthermore, our method is generic enough to be applied for the optimization of the assignment energy in other areas than surface registration (*e.g.* image registration), as long as the input data can be represented by graphs.

The initial correspondence search stage is also very important for the correct convergence of the post-processing algorithm. The probability that the global maximum is achieved increases with increasing quality of the initial correspondences. Without initial correspondences, the chances for our method to hit a local maximum, which may be very far from the global one, are very high. For our method to perform correctly, a reliable initial correspondence configuration must be supplied, so that the incorrect assignments can be properly identified and manipulated, and new assignments can be accurately inserted. Without a initial correspondences, the first operations would be the insertion of assignments between the nodes with higher sim-

ilarity. If these assignments are wrong, the future operations will be based on a wrong initial correspondence configuration, and a improper local maximum will be reached.

Although a simple optimization method was employed, significant increases in the assignment energy were achieved. However, as no decreases in the assignment energy are allowed during optimization, it may occur that the global maximum is not reached. More elaborate optimization schemes, allowing some degree of energy reduction, can be employed in order to increase the chances of reaching the global maximum. An example for these optimization methods are genetic algorithms. Furthermore, in order to increase the accuracy of the method, the local energy computation can be supplemented by including not only the direct (unary) similarity between nodes, but also neighborhood similarities and higher order similarities.

6.2 POINT-BASED APPROACH

In this approach, we directly use points of the meshes for finding correspondences, instead of trying to group them in common regions. An overview of the entire processing pipeline of our surface matching method is shown in Fig. 51. For the purposes of this work, we consider that the required surfaces are already available: Pre-operative volumes have already been segmented, and intra-operative range images have already been preprocessed to compensate for lens distortions and other systematic errors (refer to [125] for more information on the topic). We also consider that the surfaces are represented by triangle meshes. Throughout this section, we refer to the intra-operative surface as source and to the pre-operative one as target. We aim to establish correspondences between the input surfaces, which can be used to compute a mapping between them.

The first step in our method is to select a set of feature points. As we cannot assume that feature points are consistently selected on both source and target surfaces, *i.e.*, that selected feature points are located in the same positions on both surfaces, we perform feature selection only on the source surface. Assuming that the similarity between descriptors of corresponding points lies within a certain tolerance, we select a set of candidate correspondences based on descriptor similarity for each source feature. Correspondences are searched by a constrained greedy optimization, implemented as a combinatorial tree search: For each combination of correspondences that satisfies a given set of constraints, and passes a validation stage, an error value is computed according to an error metric. The combination of

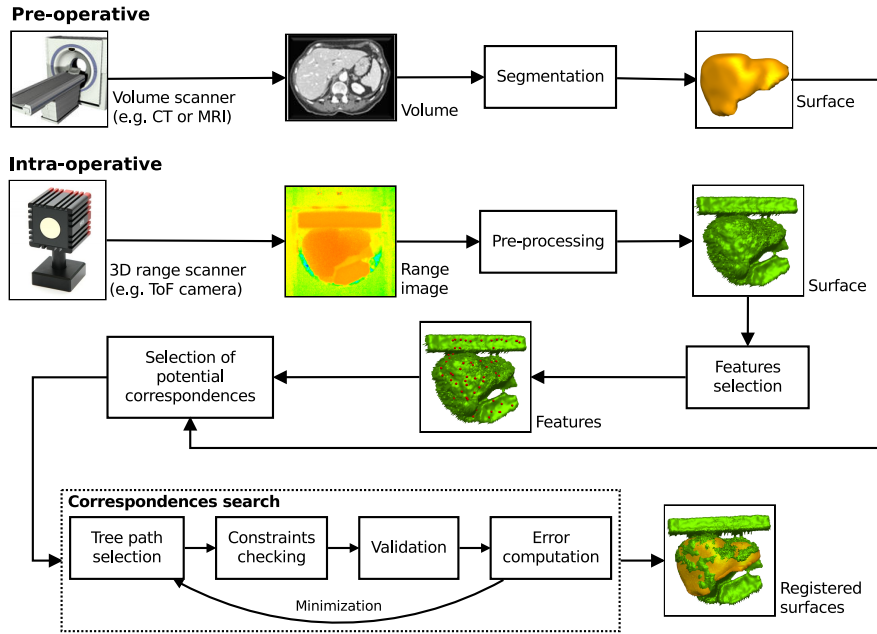


Figure 51: Processing pipeline for our surface-based intra-operative registration method.

correspondences with smallest error is taken as the final correspondence set.

As the error metrics and the consistent selection of features play critical roles in surface matching, we elaborate on their importance in Sec. 6.2.1. Sec. 6.2.3 shows how we reduce the search-space by selecting features and candidate correspondences. In Sec. 6.2.4 we present the new error metric for feature-less surfaces employed in this work. Finally, in Sec. 6.2.5 we present the method used for searching for correspondences and error optimization.

6.2.1 Error metrics and feature selection in surface matching

The error metric is an important part of surface matching. As surface matching is an optimization procedure, which searches for the correspondence set that minimizes an error value given by a specific error metric, the metric must be robust enough to differentiate a possibly correct match, assigning a lower error to it, from a probably incorrect one, which should receive a higher error. Let us assume two discrete surfaces $S = \{s_i\}$ and $T = \{t_j\}$, the source and the target surfaces, respectively. Let us also assume that each point $s_i \in S$ and $t_j \in T$ has a descriptor, which characterizes its neighborhood, and that the function $q : S \times T \rightarrow \mathbb{R}$ measures the incompatibility between a point on

S and one on T based on their descriptor dissimilarity, assuming that a distance metric between two descriptor entities has been defined along with the descriptor itself. Finally, let us assume the set of correspondences $C \subset S \times T$ between source and target surfaces, which is defined by its characteristic function $\sigma_C : S \rightarrow T$. This function is partial and injective, *i.e.*, not all points have correspondences, but the ones that have a correspondence, have a single one. Wang et al. [263] and Raviv et al. [213] stated their error metrics as a *quadratic assignment problem* (QAP), as follows:

$$E_{\text{QAP}}(C) = w \sum_{(s_i, t_j) \in C} \sum_{(s_k, t_l) \in C} (d_{\text{geo}}(s_i, s_k) - d_{\text{geo}}(t_j, t_l))^2 + \sum_{(s_i, t_j) \in C} q(s_i, t_j) \quad (6.19)$$

where $d_{\text{geo}}(\cdot, \cdot)$ denotes the geodesic distance between two points on the same surface, and w a weighting scalar to balance the influence of both equation terms. They minimize descriptor dissimilarity, maintaining geometric consistency at the same time, as distances between source points must be equivalent to the distances between corresponding target points, *i.e.*, ensuring that points belonging to a particular neighborhood on the source surface are assigned to points in a similar neighborhood on the target surface.

Although the minimization of $E_{\text{QAP}}(\cdot)$ works sufficiently well surfaces with complex shapes, such as a human shape, for example, where neighborhoods are very well characterized by different geodesic fields, in the case of intra-operative registration, where the surfaces are nearly flat, and points have similar geodesic fields, this minimization is error-prone. Minimizing $E_{\text{QAP}}(\cdot)$ implicitly means that if a spatial configuration of source landmarks fits to the spatial configuration of target landmarks, these probably correspond to each other. For surfaces representing a human shape, for example, having a landmark on each hand and one on the head, there is probably only a single matching configuration on the target surface. However, for nearly planar surfaces, there are several configuration alternatives that would match.

For example, let us consider the cases depicted in Fig. 52. In these examples we want to match a set of source landmarks to a set of target landmarks, which were previously selected in order to reduce the search-space in the correspondence search stage. In configuration 1, landmarks are located close to each other, where points have similar inter-point distances. In configuration 2, landmarks are located in more unique positions, where inter-point distances differ from each other, forming a more widespread and non-planar spatial landmark

configuration. Because of the poor landmark configuration chosen in configuration 1, registration becomes ambiguous: Several registration alternatives deliver small errors. However, because of the more elaborate landmark selection in configuration 2, the correct registration becomes easily distinguishable.

While consistent feature selection could solve the problem of having poor landmark configurations, for the surfaces of interest for intra-operative registration, no consistent selection is possible, *i.e.*, features vary greatly in number and position between the surfaces. This fact is due to the inter-modal aspect of the surfaces. As pre-operative surfaces are usually generated by the segmentation of medical images, while intra-operative surfaces are generated by 3D range scanners, the following issues occur: Different noise levels, distortions due to different acquisition principles, deformations due to breathing and surgical manipulation, lack of prominent landmarks.

Zhang et al. [285] presented a very powerful method for the registration of surfaces undergoing very large non-isometric deformations. Their error metric is not based on distance differences and descriptor similarities, but on the surface deformation required to register one surface to the other. They search for the set of correspondences that requires the minimum amount of deformation to match the surfaces. While this error metric can solve some of the previous method's problems, consistent feature selection still plays a major role in their method: Because of the fact that the computation of deformations is very time-consuming, they require consistent features in order to provide a solution in a reasonable space of time.

Without the advantages provided by consistent feature selection, preventing us from directly employing a simpler error metric, as it becomes error-prone, and a more complex one, due to the fact that it becomes computationally untreatable, we focus our attention in the incorporation of a measure of reliability of a particular spatial configuration of landmarks in the error metric. This would allow us to employ an easier and faster method to compute the error metric, losing the dependency on consistent feature selection.

Given a set of correspondences C , we can generalize the error metrics for surface registration as follows:

$$E_{\text{reg}}(C) = wE_{\text{fit}}(C) + E_{\text{corresp}}(C) \quad (6.20)$$

where $E_{\text{fit}}(\cdot)$ is an error measure of how well two surfaces fit geometrically to each other, and $E_{\text{corresp}}(C)$ is an error metric of how alike the source landmarks are to their corresponding ones on the target surface. Supposing we have a metric for spatial configuration reliabil-

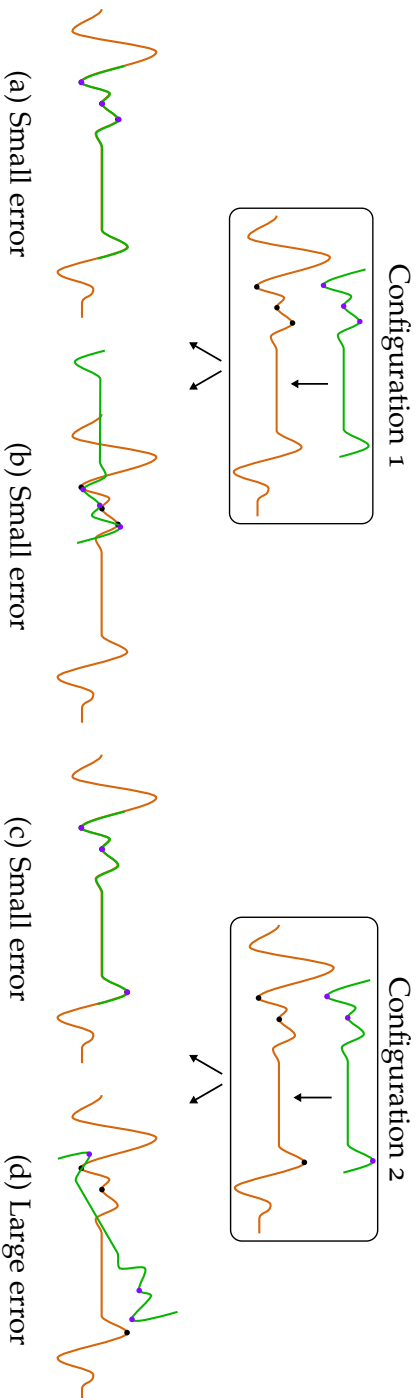


Figure 52: Comparison of the registration of a set of source and target landmarks, selected close to each other (configuration 1) and from more unique locations (configuration 2), thus forming a more widespread and non-planar spatial landmark configuration. In cases (a) and (b), both right and wrong registrations deliver small registration errors, while in cases (c) and (d), only the correct one delivers a small error.

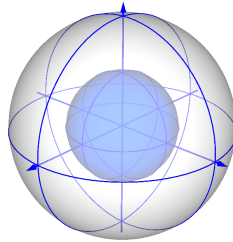


Figure 53: Spherical support volume structure used for the descriptor presented in [257].

ity of landmarks, $E_{\text{conf}}(\cdot)$, we incorporate it as a weighting term of the error metric, as follows:

$$E_{\text{reg}}(C) = (wE_{\text{fit}}(C) + E_{\text{corresp}}(C))E_{\text{conf}}(C) \quad (6.21)$$

In Sec. 6.2.4, we present a new error metric for matching featureless surfaces, with the incorporation of a measure for configuration error.

6.2.2 Descriptor

Our descriptor is based on the descriptor structure presented in [257], where a so-called *spherical support volume* around a vertex is subdivided into smaller volumes, which are indexed in local spherical coordinates (Fig. 53). For each smaller volume subdivision, histograms of angles between vertices' normals and the local coordinate system are computed. The reasoning behind is that the use of a descriptor structure, where the location inside the support volume plays a role, helps increasing the descriptor's discriminative power, while the histograms reduce the influence of the different triangulations and of the noise [257]. Key, however, for a volume subdivision that is effective in increasing the discriminative power, is the computation of a local coordinate system that is repeatable across surfaces, independently of the different triangulations or noise levels of the meshes.

In [257] the three eigenvectors of a distance weighted approximation of the covariance matrix of the points within the support volume is used as an orthonormal basis for the local coordinate system. The direction of the vectors (positive or negative) is disambiguated by majority voting, *i.e.*, the more points in the positive (or negative) direction of a vector, the more votes for this particular direction. One problem with this scheme, however, is that, if the meshes are not evenly sampled, or not sampled in the same way, both the matrix computation as well as the disambiguation of the vector directions will be

affected. This problem reduces the descriptor's repeatability. Noise will affect them as well, by introducing statistical outliers, which can become significant for smaller support volumes or not densely sampled meshes.

To reduce the influence of uneven sampling and of noise, instead of using the points within the support volume for the computation of the local coordinate system, we employ their normals, which, as shown by Petrelli and Di Stefano [207], are more robust and repeatable across surfaces. The normals are weighted by the influence area (Voronoi area [189]) of their respective points, which helps accounting for uneven distributions of the mesh vertices, as vertices in low density regions have bigger Voronoi areas. We compute the approximation of the covariance matrix C_i for the spherical support volume S_i around vertex v_i as follows:

$$w_i = \frac{1}{\sum_{v_j \in S_i} A(v_j)} \quad (6.22)$$

$$C_i = w_i \sum_{v_j \in S_i} A(v_j) \vec{n}_j \vec{n}_j^T \quad (6.23)$$

where $A(\cdot)$ denotes the Voronoi area of a vertex, \vec{n}_j the unit normal of vertex v_j , \vec{n}_j^T the transposed normal vector, and r is the radius of S_i . As C_i is symmetric, its eigenvectors form an orthonormal basis for the region around v_i . It is important to notice that C_i is not a covariance matrix, but it features the properties that we seek for the computation of the basis for the local coordinate system: It is invariant to translation and scale, since the normals are invariant to these operations. We can also show that, if we rotate the data set, the eigenvectors of the matrix C_i rotate the same amount as the data (see appendix A).

Although the eigenvectors of C_i form an orthonormal basis, their directions must still be disambiguated. In order to account for uneven vertex distributions, instead of orienting the vectors in the direction of higher point concentration, we orient them in the direction of higher influence area. Let \vec{x}_i^+ , \vec{y}_i^+ and \vec{z}_i^+ denote the eigenvectors of C_i in increasing order of their eigenvalues, while \vec{x}_i^- , \vec{y}_i^- and \vec{z}_i^- denote their

opposite directions. The disambiguated basis vector \vec{x}_i is computed as follows:

$$w_{x_i}^+ = \sum_{(v_j \in S_i) \wedge (\vec{n}_j \cdot \vec{x}_i^+ \geq 0)} A(v_j) \quad (6.24)$$

$$w_{x_i}^- = \sum_{(v_j \in S_i) \wedge (\vec{n}_j \cdot \vec{x}_i^- > 0)} A(v_j) \quad (6.25)$$

$$\vec{x}_i = \begin{cases} \vec{x}_i^+ & w_{x_i}^+ \geq w_{x_i}^- \\ \vec{x}_i^- & \text{otherwise} \end{cases} \quad (6.26)$$

The \vec{z}_i axis is disambiguated by the same procedure, while the \vec{y}_i axis is obtained as $\vec{y}_i = \vec{z}_i \times \vec{x}_i$. Our local orthonormal coordinated system for vertex v_i is given by $(\vec{x}_i, \vec{y}_i, \vec{z}_i)$.

Having a repeatable and disambiguated local coordinate system, we are now able to subdivide the support volume into smaller sub-volumes by creating boundaries in the elevation, azimuth and radial dimensions. Let $R = \{r_0, \dots, r_{(N_{\text{radius}}-1)}\}$, $\Theta = \{\theta_0, \dots, \theta_{(N_{\text{elevation}}-1)}\}$, and $\Phi = \{\phi_0, \dots, \phi_{(N_{\text{azimuth}}-1)}\}$ denote the radial, elevation and azimuth subdivisions, respectively. As subdivisions closer to the center of the sphere are smaller than the others and may not contain significantly enough points for computing a repeatable histogram, we define a minimum radius r_{min} in order to avoid this problem. The subdivision $S_i(r_k, \theta_l, \phi_m)$, in which lies a point $p = (x_p, y_p, z_p)$ within the support volume of a particular vertex $v_i = (x_i, y_i, z_i)$, can be identified by first converting p to v_i 's local coordinate system, as follows:

$$p_{\text{local}} = \begin{bmatrix} \vec{x}_i \\ \vec{y}_i \\ \vec{z}_i \end{bmatrix} (p - v_i) \quad (6.27)$$

where $p_{\text{local}} = (x_{\text{local}}, y_{\text{local}}, z_{\text{local}})$ denotes the point p in local coordinates. After computing p_{local} , we are able to compute it in local spherical coordinate systems, as follows:

$$r = \sqrt{x_{\text{local}}^2 + y_{\text{local}}^2 + z_{\text{local}}^2} \quad (6.28)$$

$$\theta = \cos^{-1} \left(\frac{z_{\text{local}}}{r} \right) \quad (6.29)$$

$$\phi = \tan^{-1} \left(\frac{y_{\text{local}}}{x_{\text{local}}} \right) \quad (6.30)$$

where $p_{\text{spherical}} = (r, \theta, \phi)$. Note that the inverse of tangent in equation 6.30 must take the correct quadrant of the plane defined by x

and y into account. We assign $p_{\text{spherical}}$ to its respective subdivision $S_i(r_k, \theta_l, \phi_m)$ in the support volume as follows:

$$k = \begin{cases} \left\lfloor \frac{r-r_{\min}}{r_{\max}-r_{\min}} N_{\text{radius}} \right\rfloor & r < r_{\max} \\ N_{\text{radius}} - 1 & \text{otherwise} \end{cases} \quad (6.31)$$

$$l = \begin{cases} \left\lfloor \frac{\theta}{2\pi} N_{\text{elevation}} \right\rfloor & \theta < 2\pi \\ N_{\text{elevation}} - 1 & \text{otherwise} \end{cases} \quad (6.32)$$

$$m = \begin{cases} \left\lfloor \frac{\phi}{\pi} N_{\text{azimuth}} \right\rfloor & \phi < \pi \\ N_{\text{azimuth}} - 1 & \text{otherwise} \end{cases} \quad (6.33)$$

where r_{\max} denotes the radius of the support volume S_i .

Within each subdivision $S_i(r_k, \theta_l, \phi_m)$ of the support volume of a vertex v_i , vertex counts are accumulated in bins of a histogram according to the cosine of the angle between \vec{n}_j and \vec{z}_i ($\vec{n}_j \cdot \vec{z}_i$), for every $v_j \in S_i(r_k, \theta_l, \phi_m)$. Let $H_{(r_k, \theta_l, \phi_m)} = \{b_0, \dots, b_{(N_{\text{bins}}-1)}\}$ denote the bins of the histogram of the vertices within subdivision $S_i(r_k, \theta_l, \phi_m)$. We identify the bin b_q in which a vertex v_j will be added to the count as follows:

$$q = \begin{cases} \frac{\vec{n}_j \cdot \vec{z}_i + 1}{2} N_{\text{bins}} & (\vec{n}_j \cdot \vec{z}_i) < 1 \\ N_{\text{bins}} - 1 & \text{otherwise} \end{cases} \quad (6.34)$$

In order to compensate for uneven vertex distributions, instead of adding 1 to the vertex count of a particular bin, we add the Voronoi area of vertex v_j , $A(v_j)$. Also, in order to compensate for small shifts of the local coordinate system, we accumulate $A(v_j)$ to the neighboring subdivisions and bins, weighted by the inverse of the distance of v_j to the center of each subdivision or bin. Finally, each histogram is normalized in order to account for different point densities within the subdivisions.

The descriptor for a point v_i is given by the vector D_i composed of the different histograms of each support volume's subdivision. In order to account for variations in point densities, vertex distributions and shifts of the local coordinate system, D_i is normalized.

6.2.3 Selection of features and candidate correspondences

Feature selection is usually employed to detect points that are most unique on the surfaces. As mentioned above, the consistent detection of these points, *i.e.*, the selection of points that are in the same positions on both surfaces, has two main advantages for surface matching: Firstly, as feature points are most unique, their corresponding

ones are usually easier to identify, *i.e.*, their descriptors are very different from the other descriptors on the surface. Secondly, the search-space for possible correspondences is reduced, as only correspondences among feature points need to be searched. However, as we have shown before, surfaces of interest for intra-operative registration are mostly feature-less and feature selection across these surfaces is inconsistent. We therefore adopt the strategy employed by Gelfand et al. [115], selecting features only on the source surface and computing sets of candidate correspondences for each one of them. Candidate correspondences are selected under the assumption that the similarity between descriptors of corresponding points lies within a certain tolerance. In contrast to [115], we cannot assume rigidity between surfaces and also need to consider the previously mentioned noise and distortions, allowing higher descriptor similarity tolerances and having to work with bigger candidate correspondence sets. This issue is dealt with in Sec. 6.2.5 and 6.2.4.

For the selection of feature points on the source surface, we employ the multi-scale approach presented by Ho and Gibbins [131]. According to this approach, a point is considered a feature if it corresponds to the maximum or minimum curvedness (a measure of curvature intensity [155]) in the local neighborhoods of growing scales. One advantage of this approach is that it delivers a measure of *feature confidence*. We use this measure to separate the features into two different sets: The *basis set* (B), containing the N_B highest confidence features, which are at least a certain geodesic distance apart from each other, for a better distribution above the surface (we use the clustering radius r_{cluster} as the minimum distance; see below); and the *validation set* (V), containing all of the others. Features in V that are close to each other, within a distance δ_{val} , are merged into the highest confidence feature. The purpose of these sets is explained in Sec. 6.2.5.

For each landmark $s_i \in B \cup V$ on the source surface, a set of candidate correspondences on the target surface is selected, based on descriptor similarity. We employ here the descriptor presented in Sec. 6.2.2. It is important to note, however, that the method for surface matching presented here is independent of the descriptor chosen, requiring only the definition of a distance metric between two instances of the same descriptor.

Together with descriptor similarity, we also use the *shape-index* [155] to better identify potential correspondences. The shape-index is a scale-, rotation-, and translation-invariant measure, which continuously describes a local surface shape in the interval $[-1, 1]$ (Fig. 39 on page 86). Points with very divergent local shapes are discarded as potential candidate correspondences. Having a function $q : S \times T \rightarrow \mathbb{R}$

that measures the incompatibility between a point on S and one on T based on their descriptor dissimilarity, the candidate correspondence set T_i of landmark s_i on the source surface is defined as follows:

$$T_i = \{t_j \in T : (q(s_i, t_j)^2 \leq \delta_{\text{desc}}) \wedge (|si(s_i) - si(t_j)| \leq \delta_{si})\} \quad (6.35)$$

where T denotes the target surface, $t_j \in T$ a point on the target surface, δ_{desc} the dissimilarity tolerance, $si(\cdot)$ the shape-index of a particular point, and δ_{si} the shape deviation tolerance. Here, $q(\cdot, \cdot)$ is taken as the Euclidean distance between the descriptors of the input points. Because of the fact that the descriptors used here are normalized vectors, the maximum distance between two descriptors is 2. As for the normals, the shape-index is also computed according to [54]. Both source and target landmarks lying close to the boundaries of the mesh, within a distance smaller than the spherical volume of the descriptor, are discarded, as the descriptors in these cases are incomplete, and therefore unreliable.

After selecting the candidate correspondences, the points inside each set are clustered in order to reduce their sizes. The clustering is performed around the candidate points that have most similar descriptors to s_i , within a geodesic radius given by r_{cluster} . For the clustering, the candidate correspondences of s_i are sorted according to their descriptor similarity. Starting from the most similar candidate, all other points within the defined radius are removed from the candidate correspondence set. Note that by clustering the candidate correspondences for a particular source landmark, we are introducing a localization error into the correct correspondence position equal to r_{cluster} at most.

6.2.4 A new error metric for the registration of feature-less surfaces

As shown in Sec. 6.2.1, the spatial configuration of landmarks plays a crucial role in surface matching. As the consistent identification of unique landmarks by means of feature selection is not possible for surfaces of interest of intra-operative registration, a measure of spatial configuration reliability is desirable in the error metric. Following the formulation of the surface registration error metric presented in Eq. 6.21, we show here how the composing terms are estimated. In Secs. 6.2.4.1 and 6.2.4.2, we present the error measures for geometric fitting ($E_{\text{fit}}(\cdot)$) and correspondence likelihood ($E_{\text{corresp}}(\cdot)$), respectively. In Sec. 6.2.4.3, we revisit the work of Fitzpatrick et al. [102] on error estimation for point-based registration, and derive an expression for

the reliability of the spatial configuration of landmarks, which we denote *configuration error* ($E_{\text{conf}}(\cdot)$).

6.2.4.1 Fitting error

For the computation of the fitting error, we employ the pseudo-Hausdorff distance, which, as shown by Eckstein et al. [93], is able to deal with non-rigid deformations of similar shapes. The pseudo-Hausdorff distance is a global metric that allows one to measure how far two subsets of a metric space are from each other. The fitting error for a is computed as follows:

$$E_{\text{fit}}(C) = \frac{1}{|C|} \sum_{(s_i, t_j) \in C} A(s_i) \left(\sum_{(s_k, t_l) \in C} \frac{A(t_l)}{d_{\text{Eucl}}(\varphi(s_i), t_l) + \varepsilon} \right)^{-1} + \frac{1}{|C|} \sum_{(s_i, t_j) \in C} A(t_j) \left(\sum_{(s_k, t_l) \in C} \frac{A(s_k)}{d_{\text{Eucl}}(t_j, \varphi(s_k)) + \varepsilon} \right)^{-1} \quad (6.36)$$

where $A(\cdot)$ denotes the influence area of a particular point [189], $d_{\text{Eucl}}(\cdot, \cdot)$ the Euclidean distance, ε a small number for ensuring that the denominators are not zero, and $\varphi : S \rightarrow T$ a mapping from the source to the target surface spanned by the correspondence set C .

6.2.4.2 Correspondence error

Like Wang et al. [263] and Raviv et al. [213], we compute the correspondence error as follows:

$$E_{\text{corresp}}(C) = \frac{1}{|C|} \sum_{(s_i, t_j) \in C} q(s_i, t_j)^2 \quad (6.37)$$

where $q : S \times T \rightarrow \mathbb{R}$ measures the incompatibility between a point on S and one on T based on their descriptor dissimilarity, as introduced in Sec. 6.2.3.

6.2.4.3 Configuration error

In their work on rigid point-based registration, Fitzpatrick et al. [102] came to the conclusion that the use of a fitting error by itself is an unreliable indicator of registration accuracy. The same conclusion can be intuitively extended to other point-based registration errors, as even a complex deformation error would deliver similar values for cases (a) and (b) (Fig. 52), making the distinction between a right and wrong registration nearly impossible. For this reason, Fitzpatrick

et al. [102] derived an approximate expression for measuring the registration error at a spatial position x , which can be any point in the space, based on a spatial configuration of landmarks and their localization errors. This error is known as the *target registration error* (TRE), and it is widely used to guide the placement of markers around a particular target (e.g. a tumor) in marker-based surgery guidance systems. Although variations of the TRE expression were proposed for anisotropic, inhomogeneous noise [73], they do not have closed-form solutions for their computation. The expected value for the TRE at a spatial position x , under the assumption of isotropic, homogeneous, independent, zero-mean Gaussian noise, is given by:

$$\langle \text{TRE}^2(x, L) \rangle = \frac{\langle \text{FLE}^2(L) \rangle}{|L|} \left(1 + \frac{1}{3} \sum_{k=1}^3 \frac{d_k^2}{f_k^2} \right) \quad (6.38)$$

where L denotes the landmarks, d_k the distance from the target to the principal axis \vec{v}_k of L , and f_k the root-mean-square distance of the landmarks to their principal axis \vec{v}_k . The *fiducial localization error* (FLE) denotes the imprecision in locating a landmark. In the context of marker-based guidance, FLE denotes the error of the tracking system in locating the markers. The TRE expression states that the more widespread and non-planar the landmark set, the more reliable it is. In the context of surface matching, a set of widespread correspondences is also beneficial because it reduces the likelihood of ambiguities: The larger the area spanned by the fiducials (in all directions) the fewer potential matches with a good fitting value can be found.

In our case, a correspondence set ($C \subset C_S \times C_T$) is composed of a set of landmarks on the source surface (C_S) and a set of corresponding points on the target surface (C_T), one for each landmark. Let us also assume that we have a target point defined beneath the target surface. In this context, we can define the purpose of intra-operative registration to be the search for a transformation that establishes a relation between the current patient situation, represented by the intra-operative surface, to the pre-operatively planned target (Fig. 54a). Having a pre-defined target, one could propose the use of the TRE expression in order to find a set of correspondences that minimizes the error to the target point. Unfortunately, the TRE expression assumes that the correct correspondences are previously known (*i.e.*, the spatial relation between target point and intra-operative data) and measures the error to the target point based on the localization error of these known correspondences. Minimizing the TRE expression for surface matching based on a real target point would misguide the registration towards the location of the target point, as the distances to the principal axes (d_k) are smaller there (Fig. 54b).

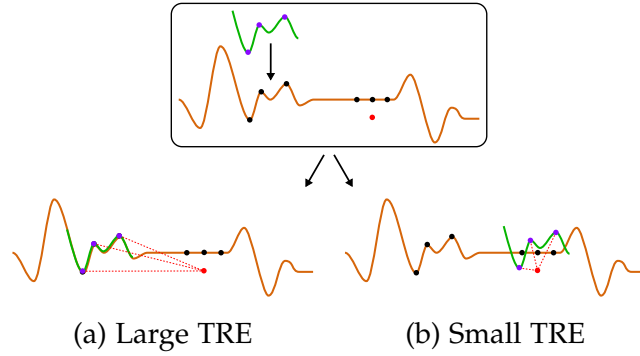


Figure 54: The computation of *target registration error* (TRE) [102] for surface matching may be misleading, as the TRE assumes that the correct correspondences are previously known. In (a), the correct registration returns a large TRE, as the distance to the target point (red dot) to the source landmarks is large. Minimizing the target registration error misguides the registration towards the target point, as shown in (b), where the distances between target point and source landmarks are smaller.

In order to constrain the TRE as a local measure for the spatial configuration error of a set of landmarks, we compute a sphere around the landmarks, centered on their centroid, and average the TRE for all possible targets inside this sphere. The advantage of this approach is that the error values of different spatial configurations are directly comparable, as the target points are defined locally for each particular landmark configuration. Furthermore, as all target points are defined locally for the landmarks, inside their bounding sphere, the result is only dependent on their spatial configuration, which is what we want to quantify, and not on the distance to the targets. The expression for the estimation of the average TRE is as follows:

$$\begin{aligned}
 \overline{\langle \text{TRE}^2(\mathbf{L}) \rangle} &= \frac{1}{|\mathbf{U}|} \int_{\mathbf{u}} \langle \text{TRE}^2(\mathbf{u}, \mathbf{L}) \rangle \, d\mathbf{u} \\
 &= \frac{\langle \text{FLE}^2(\mathbf{L}) \rangle}{|\mathbf{L}|} \left(1 + \frac{\pi}{8} \sum_{k=1}^3 \frac{1}{f_k^2} \right) \quad (6.39)
 \end{aligned}$$

where \mathbf{U} denotes the space within the bounding sphere, and $|\mathbf{U}|$ the volume of the bounding sphere. The FLE is set to the correspondences localization error introduced by the clustering of the potential correspondences (Sec. 6.2.3), *i.e.*, $\langle \text{FLE}^2(\mathbf{L}) \rangle = r_{\text{cluster}}^2$. For a particular correspondence set \mathbf{C} , we compute the configuration error as follows:

$$E_{\text{conf}}(\mathbf{C}) = \max(\overline{\langle \text{TRE}^2(\mathbf{C}_S) \rangle}, \overline{\langle \text{TRE}^2(\mathbf{C}_T) \rangle}) \quad (6.40)$$

where C_S and C_T denote the source and target landmarks in C , respectively.

6.2.5 Correspondences search

The purpose here is to find the set of at least N correspondences with minimal registration error $E_{\text{reg}}(\cdot)$ (Eq. 6.21). A correspondence between the source surface S and the target surface T is given by a tuple of points $c = (s_i, t_j^i)$, where $s_i \in S$ and $t_j^i \in T_i \subseteq T$, where T_i denotes the set of candidate correspondences for point s_i (see Sec. 6.2.3). A set of correspondences $C = \{c_0, \dots, c_n\}$ is a finite enumeration of correspondences, such that each point s_i and t_j appears only once in the enumeration, *i.e.*, there are no multiple correspondence assignments to a single point (see Sec. 6.2.1).

A greedy optimization is employed to find the minimum of the error metric. The optimization enumerates all possible initial correspondence sets of N_{init} correspondences. In order to speed-up the search, we utilize, in this initial stage, only the landmarks in the basis set B , which are the most confident landmarks on the surfaces, according to a confidence measure (see Sec. 6.2.3). During this enumeration, every time a correspondence is added to a potential initial correspondence set, a set of constraints is applied, declaring an initial correspondence set invalid if any of the constraints is not fulfilled (Sec. 6.2.5.1). These constraints enforce the reduction of the search-space even more. If an initial correspondence set passes all of the constraints, these correspondences are validated by a voting strategy (Sec. 6.2.5.2). Any other correspondence that “votes” in favor of a particular initial correspondence set is also added to it. If a correspondence set receives at least N_{votes} votes, such that $N = N_{\text{init}} + N_{\text{votes}}$, its error is computed according to the error metric presented in Sec. 6.2.4. The set with the smallest error is taken as the final correspondence set (C_{final}), and is used for mapping the source surface onto the target surface. Note that if no correspondence set manages to pass the constraints check and voting stage without being declared invalid, no correspondences are returned by this method. This optimization is implemented as the traversal of a search-tree (Sec. 6.2.5.3), similar to [115, 285].

6.2.5.1 Constraints

In order to reduce the search-space, a set of constraints is applied to the correspondence set every time a new correspondence is added to it. If the set, with the new correspondence, does not fulfill any of these constraints, the set is said to be invalid, and is discarded. For

a new correspondence (s_i, t_j^i) inserted in the correspondence set, the following constraints apply:

- *Uniqueness constraint*: Each point on the source surface is only allowed to have a single corresponding point on the target surface. Inversely, every point on the target surface can only correspond to a single point on the source surface. Following these premises, neither s_i nor the point represented by t_j^i may be found in any other correspondence in the set.
- *Geodesic distortion constraint*: For every pair of correspondences in the set, the geodesic distance between the points on the source surface must be similar to the distance between their corresponding points, within some distortion and deformation tolerance δ_{geo} , plus the uncertainty introduced by the clustering radius r_{cluster} . This means, for every correspondence (s_k, t_l^k) in the set, the assertion $|d_{\text{geo}}(s_i, s_k) - d_{\text{geo}}(t_j^i, t_l^k)| \leq (\delta_{\text{geo}} + 2r_{\text{cluster}})$ must hold, where $d_{\text{geo}}(\cdot, \cdot)$ gives the geodesic distance between two points (we compute geodesic distances on triangular meshes according to [252]).
- *Gauss map orientation constraint*: The Gauss map is a mapping of a surface to the unit sphere. In order to ensure that both the source points and the corresponding target points have the same orientation, the orientation of their Gauss maps is considered for each triplet of correspondences in the set. Two Gauss maps have the same orientation if, and only if, the determinant of their normals have the same sign [281]. This means, for every correspondence triplet (s_i, t_j^i) , (s_k, t_l^k) and (s_m, t_n^m) , in the set, the assertion $\det(\vec{n}_i, \vec{n}_k, \vec{n}_m) \det(\vec{n}_j^i, \vec{n}_l^k, \vec{n}_n^m) \geq 0$ must hold, where \vec{n}_x denotes the unit normal of point s_x .

6.2.5.2 Voting

Every initial correspondence set, of size N_{init} , that fulfills all of the above constraints, is submitted to a voting strategy, which aims to evaluate the adequacy of the mapping spanned by these correspondences to other possible correspondences. For this reason, a mapping $\Phi : S \rightarrow T$ is computed. For every source point in the set $B \cup V$ that does not yet have a correspondence, its closest correspondence candidate is searched for as follows:

$$t_j^i = \arg \min_{t_k^i \in T_i} d_{\text{Eucl}}(\Phi(s_i), t_k^i) \quad (6.41)$$

where $d_{\text{Eucl}}(\cdot)$ denotes the Euclidean distance. A vote is cast in favor of the correspondence set if the addition of the correspondence (s_i, t_j^i)

to it still allows all of the constraints in Sec. 6.2.5.1 to be fulfilled. If a correspondence set achieves at least N_{votes} votes, it is said to be validated, and all correspondences that cast votes in its favor are added to it.

6.2.5.3 Implementation

The optimization strategy is implemented as a search-tree, whose root is the empty set, each node represents a correspondence, and a path from a tree-leaf to the root represents a correspondence set. During the construction of the tree, the constraints are applied with the insertion of every new branch. If any of the constraints is not fulfilled, the branch is pruned. Every path that reaches a length of N_{init} nodes is submitted to the voting procedure. The path with the smallest error is declared as the final set of correspondences.

6.2.6 Evaluation and results

For the evaluation of the point-based approach, we performed here experiments for assessing:

- The robustness of the presented descriptor in the characterization of surfaces with high noise levels (Sec. 6.2.6.1);
- The robustness and accuracy of the point-based surface matching method for the registration of surfaces acquired from objects under intra-operative conditions. For these experiments, a respiratory motion simulator was employed for applying deformation to the models. Acquisitions were conducted using a time-of-flight (ToF) camera and a computed tomography (CT) (Sec. 6.2.6.2).

6.2.6.1 Descriptor

In order to evaluate the robustness of the presented descriptor (Sec. 6.2.2) in the presence of noise, we compared it to the descriptor presented by Tombari et al. [257], using time-of-flight (ToF) data simulated from computed tomography (CT) data, according to [181]. The data set was composed of a head, a knee and a liver (Fig. 55). One characteristic of this data set is the low occurrence of prominent points. A set of virtual markers were generated throughout the entire organs' volumes, which were used as ground-truth reference.

The experiment consisted of computing a registration between the surfaces and evaluating the average of the distance between corre-

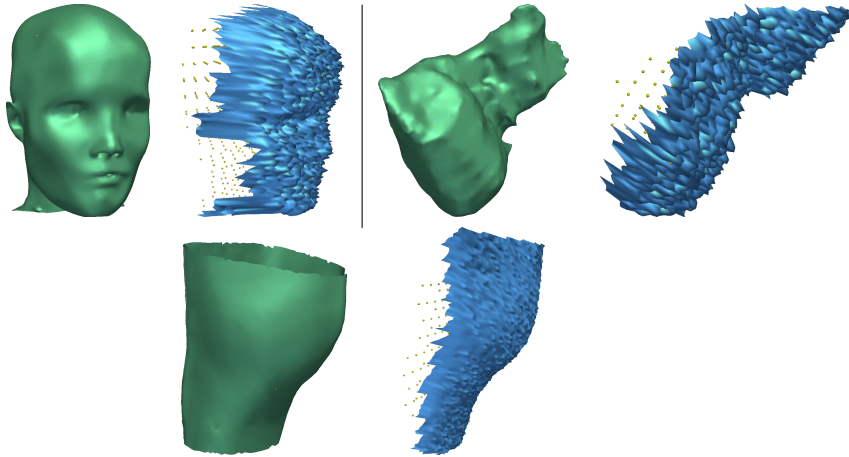


Figure 55: Simulated ToF data (blue) and its corresponding CT surface (green) used in the evaluation. Top-left: Human head; Top-right: Human liver; Bottom: Human knee.

sponding markers after registration. For surface registration, we employed in this experiment the method presented by Gelfand et al. [115], which can be used for finding a rigid registration between the surfaces. Though this method was not designed for matching surfaces from different modalities or in the presence of high noise levels, it is enough to illustrate the robustness of the presented descriptor. A minimum of 20 correspondences were searched, using a clustering radius of 20 mm (see [115] for more details on the parameters). The radius of the descriptor (r_{\max}) was set to 30 mm. For each data set, the descriptor similarity threshold varied incrementally in the interval $[0.4, 1]$ (ϵ ; see [115]), with 0.1 step size. The results are shown in Fig. 56. In all cases, the correspondences search method was able to find an parameter interval where the mean distance between ground-truth markers lie close to 5 mm, using the descriptor presented here. However, the descriptor presented by Tombari et al. [257] is not descriptive enough to ensure correct correspondences.

6.2.6.2 Surface registration

The objective of our experiments was to evaluate the accuracy, computation time, and robustness to parameter variation of the surface matching method presented here for the registration of data scanned by a ToF camera to surfaces extracted from CT images. We also evaluated the influence of the newly introduced configuration error, and

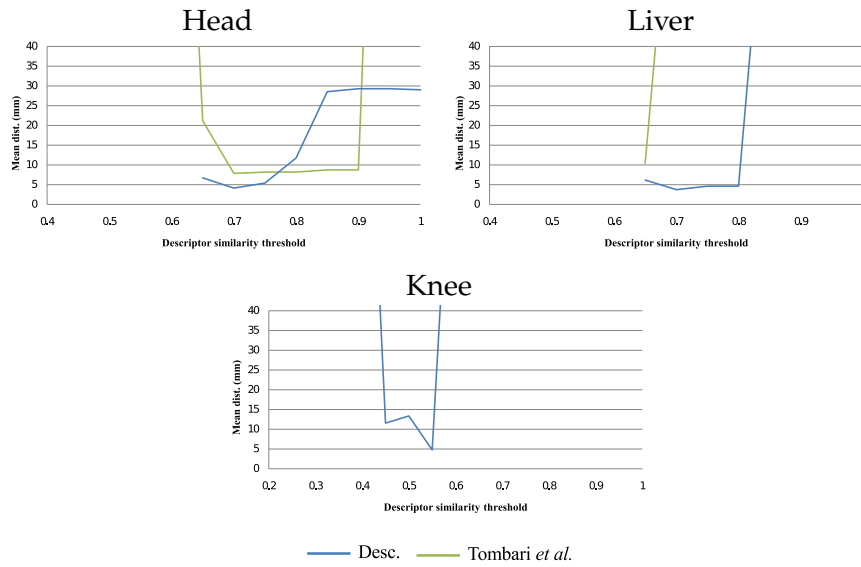


Figure 56: Comparison between the descriptor presented here (blue) and the one presented by Tombari et al. [257] (green). Shown are the mean distances between corresponding ground-truth markers for different descriptor similarity thresholds.

the robustness of our method for the registration of surfaces undergoing deformation. The validation data sets consisted of real scanned data, including a physical phantom of the human liver and a real porcine liver (*ex vivo*).

The physical phantom data set was composed of a previously segmented CT scan of a human liver phantom, and of real ToF acquisitions of this phantom. For the ToF acquisitions, the phantom was placed in a respiratory motion simulator [178] (Fig. 57a). Two acquisitions were made, one for the full expiration state (Fig. 57c), another for the inspiration state (2.4 cm; Fig. 57d). Before the CT and ToF acquisitions took place, a set of small nearly flat (≤ 1 mm) markers was spread over the surface, to serve as a ground-truth reference. The physical phantom allows us to evaluate our method on an object that actually resembles the shape of a liver, as a real liver loses its original shape when extracted from the body, due to the lack of blood flow and to the loss of fluids.

The porcine liver data set was acquired in the same way as the physical phantom data set. Though the shape of the explanted organ does not resemble the original shape of the liver anymore, due to loss of fluids, this data set allows us to assess the performance of our method for the registration of surfaces acquired from real tissue, with its particular coefficients of light reflection and absorption.

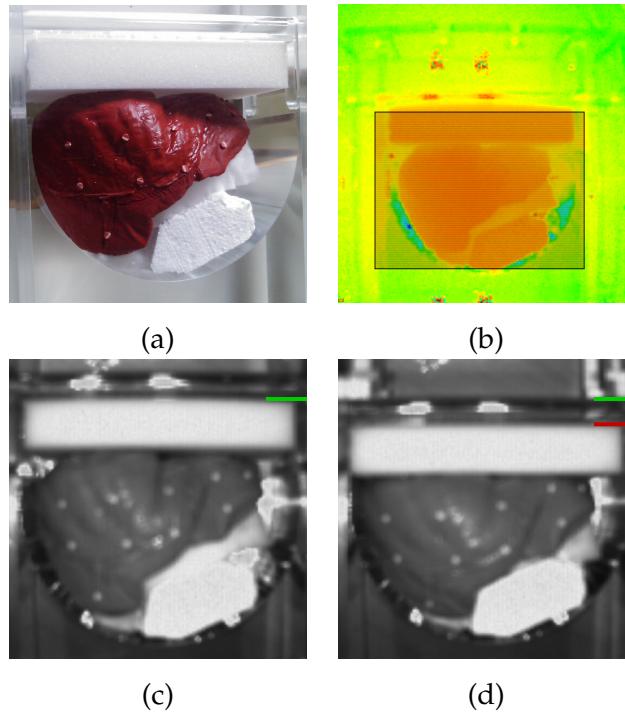


Figure 57: Evaluation setting using a physical phantom of the human liver. In (a), the phantom is positioned in a respiratory motion simulator [178]. In (b), a rectangular region of interest is selected on the ToF scan. The respiratory simulation is shown in (c) (expiration) and (d) (inspiration). The green lines next to the top right corners indicate the expiration position of the device, while the red one indicates the inspiration position. The distance between the green and red lines is 2.4 cm. Note that the original motion simulator was modified with polystyrene in order to fill empty spaces and increase the robustness of the setting.

In all acquisitions of real ToF data, the integration time parameter of the ToF camera was adjusted so as to minimize the influence of the markers on the final surface (phantom: 500 μs ; porcine: 200 μs ; human: 700 μs). ToF range images were calibrated, in order to compensate for lens distortions, and were roughly smoothed using a bilateral filter (kernel widths: $\sigma_{\text{domain}} = 1.7$; $\sigma_{\text{range}} = 21$), in order to reduce noise. Also, for each ToF acquisition, a rectangular region of interest was manually defined (Fig. 57b), as well as a distance range of interest. All CT meshes were decimated to about 10k points.

The experiments conducted for this evaluation consisted of applying the presented method for surface matching to register ToF data to corresponding CT data, and assessing the matching performance. All ToF to CT registrations were evaluated with three clustering radii

(10, 20 and 30 mm), in order to assess their influence on the registration. For every radius, the descriptor distance tolerance δ_{desc} , which directly influences the sizes of the potential correspondence sets, was chosen in the interval $[0.4, 1]$, with a step size of 0.1. This interval was found in previous experiments with ToF and CT to be tolerant enough, to allow points on the source surface to have a reasonable number of correspondence candidates, but not too tolerant, so that every point on the target surface is a candidate. The correspondence search was performed twice, for a minimum of 10 correspondences ($N_{init} = 5$, $N_{votes} = 5$, $N_B = 10$) and for a minimum of 15 correspondences ($N_{init} = 5$, $N_{votes} = 10$, $N_B = 15$), in order to evaluate the robustness of the method with more or less correspondences. The shape-index tolerance, δ_{si} (Sec. 6.2.3), was fixed at 0.25, which corresponds approximately to the eight curvature classes presented by Besl and Jain [33]. The descriptor was computed with a spherical volume with a radius of 30 mm, with two subdivisions on the radial dimension, 5 on the elevation, and 10 on the azimuth (see Sec. 6.2.3). The histograms were computed with 20 bins. This descriptor was the most adequate in a set of tests, considering the decimation levels of the meshes. As we do not have any *a priori* information about the noise on the surfaces and on the precision of the surface triangulation, we set the fiducial localization error (FLE; see Sec. 6.2.4.3) to 1. Both surface mappings required in our method (φ and Φ ; Secs. 6.2.4.1 and 6.2.5.2, respectively), as well as the registration between the surfaces using the final correspondence set (C_{final}), were computed rigidly according to [134]. All of the experiments were performed twice: In the first run the configuration error was incorporated in the error metric, while in the second run, the error was computed without it. The computations were performed with a single core 2.9 GHz Intel processor.

To assess the performance of our method, we measured: (1) The computation time; and (2) The accuracy of the registration. The accuracy was assessed by the mean distance between corresponding surface markers.

The results of our experiments are shown in Tab. 3. In this table, the average error for the entire descriptor distance interval is shown for each experiment.

With incorporation of the configuration error in the error metric, the error remained under 10 mm for the experiments using clustering radii of 10 and 20 mm, with exception for the phantom model in the inspiration state, with a clustering radius of 20 mm and searching for a minimum of 10 correspondences (this case is depicted in Fig. 58). Only very small differences could be observed between the searches for a minimum of 10 or 15 correspondences. With a cluster-

ing radius of 30 mm, errors below 10 mm could only be found for the registration in the expiration stage. Note that it was not possible to find correspondences over the whole descriptor distance tolerance interval, as shown in Fig. 58, for the experiment with porcine liver when searching for a minimum of 15 correspondences.

Without the incorporation of the configuration error, low errors could be only found in a very few cases, mostly when searching for at least 15 correspondences. For the expiration state, better results were obtained with a clustering radius of 30 mm, while for the inspiration state, the results are more optimal with a clustering radius of 20 mm.

Computation times ranged from less than one minute, for a descriptor distance tolerance of 0.4 searching for a minimum of 10 correspondences, up to 5.5 hours, for a descriptor distance tolerance of 1.0 searching for a minimum of 15 correspondences (Fig. 58). Searching for 15 correspondences results in significantly longer computation times.

Fig. 59 shows the correspondences found between both phantom and porcine liver surfaces, in the inspiration state, when searching for a minimum of 10 correspondences, with a clustering radius of 10 mm and a descriptor distance tolerance of 0.4. Note that the surfaces are nearly flat and noisy, and that there are also other objects in the environment that do not belong to the objects of interest. In the case of the porcine liver (Fig. 59b), a reflection spot can be observed directly in the middle of the surface (elongated peak), generated by the light reflection properties of real tissue. Despite these issues, the presented method was able to identify a set of correctly corresponding points.

A few qualitative results for the registration of surfaces with relatively small partial surfaces and high noise levels are shown in Figs. 60 and 61, respectively. Also, the method was employed for obtaining correspondences between the faces of different persons, as shown in Fig. 62. Observe that, even though the models have one face side almost symmetric to the other, the method was able to correctly match them.

6.2.7 Discussion

In the following sections, the results obtained for the descriptor evaluation (Sec. 6.2.7.1) and for the evaluation of the presented surface registration method (Sec. 6.2.7.2) are discussed.

	Data	Cluster. rad.		20		30		
		Corresp.	10	15	10	15	10	15
<i>With conf. error</i>	Exp.	Phantom	3.1 ± 0.5	3.4 ± 0.4	5.4 ± 1.5	6.0 ± 2.6	8.9 ± 3.8	10.4 ± 7.6
		Porcine	5.3 ± 2.0	5.2 ± 2.2	2.3 ± 0.7	2.3 ± 1.0	3.6 ± 1.1	3.1 ± 2.3
	INSR.	Phantom	4.1 ± 2.0	3.8 ± 1.6	13.8 ± 21.1	5.9 ± 2.9	93.3 ± 51.8	65.2 ± 34.2
		Porcine	5.3 ± 1.9	5.4 ± 2.2	5.5 ± 4.3	3.0 ± 1.7	37.1 ± 38.3	19.6 ± 9.2
<i>Without conf. error</i>	Exp.	Phantom	103.5 ± 16.9	89.6 ± 58.5	22.3 ± 42.5	9.8 ± 9.7	20.6 ± 24.6	12.0 ± 6.4
		Porcine	27.5 ± 25.7	7.2 ± 6.2	25.6 ± 35.0	6.7 ± 11.0	5.2 ± 2.6	3.5 ± 2.4
	INSR.	Phantom	105.6 ± 25.6	88.6 ± 59.2	59.4 ± 53.6	36.2 ± 54.1	99.9 ± 56.1	86.4 ± 36.7
		Porcine	64.6 ± 37.0	30.7 ± 36.3	25.5 ± 37.9	8.3 ± 5.0	41.6 ± 45.1	20.3 ± 8.9

Table 3: Results of the experiments. The table shows the errors between ground-truth markers for each experiment, averaged for the entire descriptor tolerance interval. All values are given in mm. We show the errors for both the computation of the registration with and without the incorporation of the configuration error (Sec. 6.2.4.3) in the error metric. Exp. indicates that the surfaces were in the expiration state (not deformed), while INSR. denotes the inspiration state.

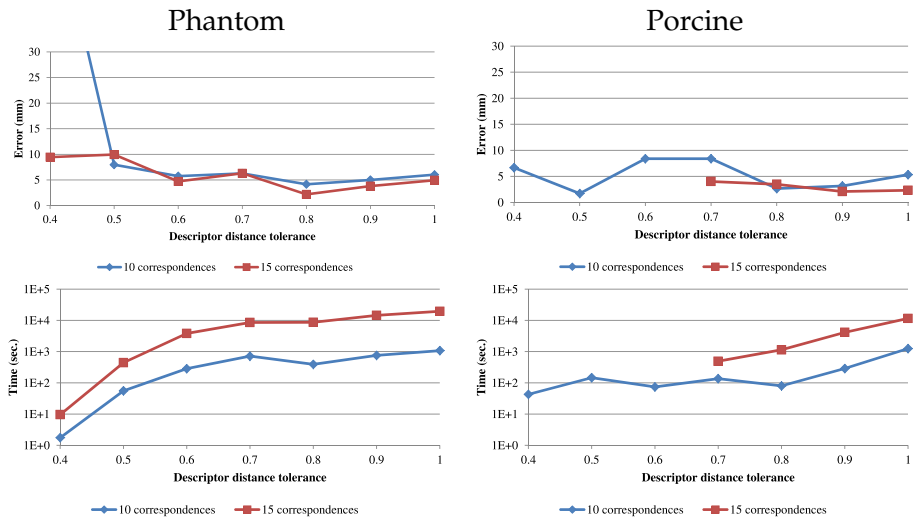


Figure 58: Error and computation times for the registration experiment with phantom and porcine liver surfaces in the inspiration state (deformed), using a clustering radius of 20 mm. Note that the computation time axes have a logarithmic scale.

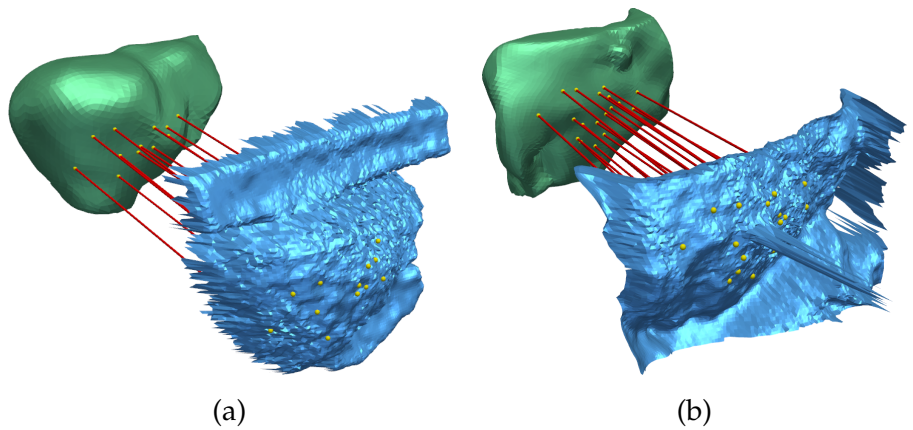


Figure 59: Correspondences found between the surfaces acquired by a time-of-flight (ToF) camera and a computed tomography (CT) of a liver phantom (a) and a porcine liver (b). The ToF surfaces are in the inspiration state (deformed).

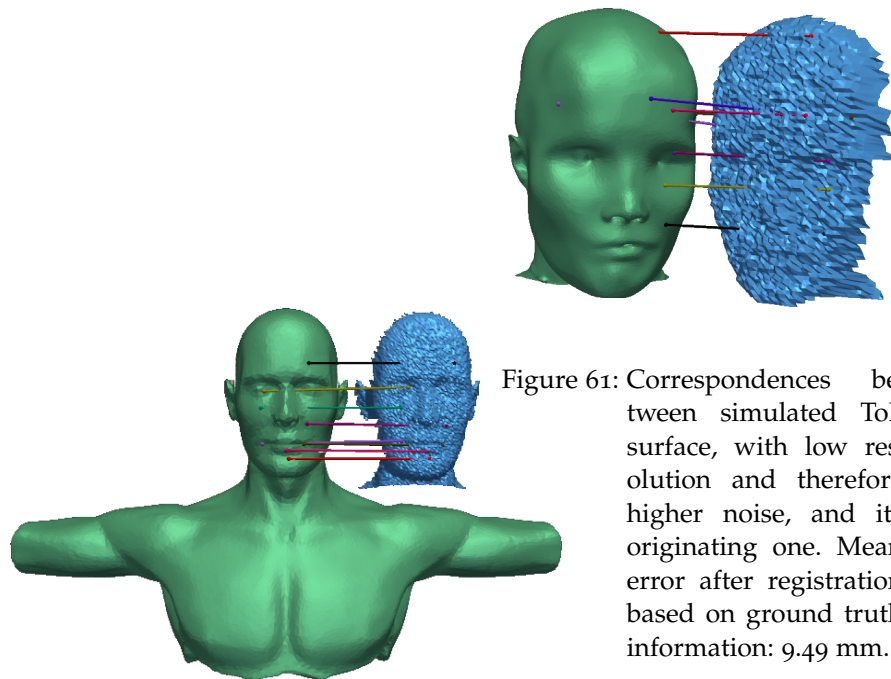


Figure 61: Correspondences between simulated ToF surface, with low resolution and therefore higher noise, and its originating one. Mean error after registration based on ground truth information: 9.49 mm.

Figure 60: Correspondences between simulated ToF surface and its originating one. Mean error after registration based on ground truth information: 4.29 mm.

6.2.7.1 Descriptor

The descriptor presented here (Sec. 6.2.2) makes use of the robustness and repeatability of the vertices normals [207] for enhancing a previously existing descriptor, from Tombari et al. [257]. Furthermore, our descriptor also compensates for uneven and unbalanced vertex distributions, by introducing the usage of vertex influence areas, represented by their Voronoi areas [189].

In a study on descriptor repeatability [207], the descriptor presented by Tombari et al. [257] showed to be relatively stable, but performed poorly in the presence of noise and different point densities. Similar results could be observed in our experiments, where the data sets present high level of noise, different densities and the meshes are feature-less. With exception of the “head” data set, which is visually the one that presents most prominent surface features for matching, no consistent registration was possible for the other data sets. Important to notice, however, that the registration method employed for the experiments was designed for rigid registration of intra-modal

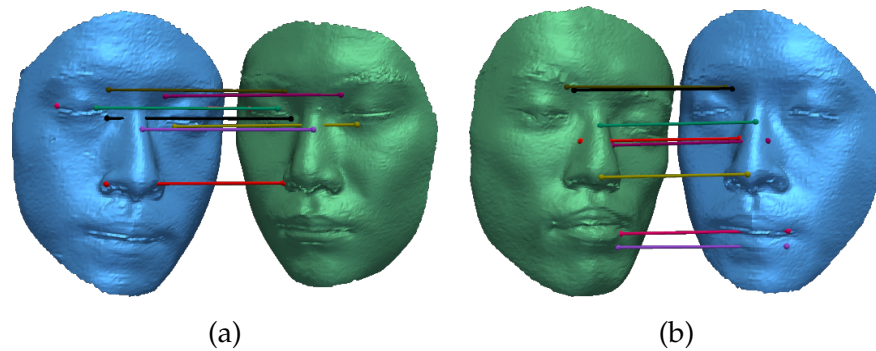


Figure 62: Correspondences between the surfaces of the faces of three different persons.

surfaces, thus considering that descriptors in approximate same locations on the two surfaces to be matched are almost similar. Also important is the fact that the results presented are the average of the distance of all corresponding virtual markers, which were evenly spaced throughout the entire CT volume, which means that a small error on the surface alignments accumulates to bigger errors on the markers that are farther away from the surface.

On the other hand, our descriptor showed to be more stable and have more discriminative power, even for surfaces with no prominent features, such as the “knee” data set, for example. In all cases, parameter intervals could be found, where the registration was very close. Although the experiments could be improved, the results obtained confirm the observation from Petrelli and Di Stefano [207], related to the higher repeatability and robustness of normals, even in noisy situations.

6.2.7.2 Surface registration

A method for automatic surface matching that directly addresses the registration challenges imposed by intra-operative environments was presented. The most critical consequences of these challenges are: (1) No consistent selection of feature points is possible; and (2) Descriptors of surface points at the same anatomical locations are not similar. Having to deal with those two issues, methods for automatic surface matching presented so far are, to the best of our knowledge, inadequate for intra-operative registration, as the consistent selection of very few feature points and reliable descriptor similarities for points at the same locations are basic assumptions that most methods rely on. In order to overcome these issues, we presented a method that finds surface correspondences by the selection of two spatial configu-

ration of landmarks on the source and on the target surfaces that can be better fitted to each other, according to an error metric. This error metric not only incorporates a fitting error, but also a new measure for spatial configuration reliability.

The main advantage of the presented method, in comparison to related work on surface matching, is its independence from consistent feature selection, *i.e.*, the selection of points on both source and target surfaces that correspond to the same locations. This characteristic allows for robust registration of feature-less surfaces and for cases where the consistent selection of features is not normally possible, such as for intra-operative registration. The implicit assumption in surface matching that, if a set of source landmarks fits to a set of target landmarks, they probably correspond to each other, is relaxed in this work. Instead, we balance the registration error with a measure of spatial configuration reliability, which we call the *configuration error*. The configuration error is based on a metric for the estimation of registration error in any given spatial position based on the localization error of a set of landmarks, called *target registration error*. Although widely applied in the context of point-based registration, this metric is not directly applicable to our surface matching task for two reasons: Firstly, neither the landmarks nor the correspondences are known a-priori. In contrast, the metric is used to actually find appropriate features and correspondences. Secondly, the position of the target relative to the ToF surface is not known. On the other hand, the formula has important properties relevant to our task: It prefers features, that have a large mean squared distance to their principal axes and a small fiducial localization error. In the case of feature-based surface matching, widespread landmarks are extremely advantageous because they reduce the possibility of ambiguous assignments.

Overall, the proposed method proved to be very accurate. As most methods for intra-operative registration presented so far focus only on the fine alignment of the surfaces, assuming that an initial registration is already provided, the robustness of the initial alignment is an issue for them. Because of the importance of the initial alignment, these methods took several measures in order to reduce the sensitivity to it. The results presented here are definitely in the acceptable range, as after employing our method for registration, the surfaces are already very closely aligned (see [52, 66, 91] for more details on fine alignment of surfaces for intra-operative registration).

The evaluation experiments were performed in as realistic intra-operative conditions as possible, and were carefully planned for the evaluation of our method. These experiments required a complex setting, involving motion simulators, ToF and CT acquisitions, and re-

liable identification of ground-truth landmarks. Although the number of data sets was relatively small, they successfully reflect several issues that may occur when dealing with surfaces acquired intra-operatively. Furthermore, the acquired data sets allowed for very elaborate evaluation of the registration algorithm and the parameters that influence its behavior. The proposed method was also successfully applied to the registration of a large set of simulated data.

Searching for a minimum of 10 or 15 correspondences did not have a great impact on the accuracy, as one might guess, since the search for 15 correspondences requires more validation points, reducing the probability of finding an incorrect match. The impact on the computation time, however, was significant, as searching for 15 correspondences introduces more correspondence combinations that need to be analyzed.

Much credit for the stability of the method must be given to the incorporation of the configuration error in the error metric. Picking feature configurations that are somehow unique (lower configuration error) increases the probability of having a correct match, if the registration error is also small. In other words, if a set of correspondences that fits an unique set of landmarks can be found, there is a good chance that these correspondences are correct. As the location of feature points is not reliable, disregarding the configuration error in the error metric causes instability in the method. Even though searching for a minimum of 15 correspondences showed to be better in this case, failing to consider the configuration error in the registration makes the method unreliable. The incorporation of the configuration error allows us to search for fewer correspondences in less computation time, while still maintaining accuracy. In order to improve the efficacy of the configuration error even more, measures of curvature intensity (*e.g.* curvedness [155]) could be incorporated in the equation as a measure of localization confidence, as points in locations with higher curvature are probably more accurate than points lying in flat regions.

Throughout the experiments, the choice of parameters did not have any major influence on the accuracy, but did have a greater impact on the computation times. To optimize the computation time, a lower descriptor distance tolerance would be recommended. However, by setting the tolerance to a very low value, many correspondence candidates are eliminated, and the possibility that none or false correspondences are found is higher. As we have already shown that the descriptor representing the same locations on ToF or CT may differ, a moderate tolerance must be allowed in order to make sure that

the correct corresponding location is also listed among the candidate correspondences.

For the registration experiments in the inspiration stage (deformation), a loss of accuracy was observed when using a larger clustering radius (30 mm). Because of the large clustering and the lack of features, registration in this case becomes ambiguous. The size of the clusters has a direct impact on the level-of-details of the representation of the target surface (number of candidate correspondences). The smaller the cluster, the more details (higher number of candidate correspondences) are considered. For nearly-flat surfaces, a large clustering radius has the same effect as aggressive smoothing, eliminating the small differences between surface parts that could be decisive for successful matching.

Both the required surface mappings, in the correspondences validation stage and in the computation of the registration error, were computed rigidly. Although the use of a rigid transform was sufficient for our purposes, it is evidently not enough when dealing with higher degrees of deformation. The choice of a more adequate class of deformation for the computation of the mappings might improve the robustness of the method and requires further investigation. However, as mentioned above, there are already methods for surface-based intra-operative registration that, given an initial alignment, compute the local scale deformation of the surfaces.

Future developments of our surface matching method include the use of non-rigid mappings for the error metric, making the method more robust to higher degrees of deformation. Also, to speed it up and for the generation of a denser set of correspondences, we are considering the development a hierarchical approach, such as presented in [281, 213]. Methods such as the one presented in [233] allow for the computation of a dense set of correspondences, given a set of sparse ones. Having a dense set of correspondences allows for more robust and accurate registration. Further measures to optimize the computation time could also be considered, such as a concurrent version of the algorithm. As we rely here on a constrained greedy combinatorial analysis to search for the minimum of the error metric, the evaluation of each possible correspondence set can be performed independently, and can be distributed across several processors or even across the cores of modern graphic cards (*e.g.* GPU, CUDA).

In conclusion, the presented method for automatic surface matching overcomes the challenges imposed by intra-operative environments and is ready for integration in clinical trials. We have shown that the incorporation of a configuration error in the error metric is very significant for the robustness and accuracy of surface matching

methods, and makes an important contribution to the field of partial surface matching. Overall, our method showed to be accurate enough to initialize fine registration procedures that account for local scale displacements.

CONCLUSIONS

Stay hungry, stay foolish.

— Steve Jobs

Commencement speech, Stanford Univ., 2005

Surface matching for intra-operative registration is not trivial. This is mainly due to the nature of the surfaces involved in the process: They are multi-modal, partially overlapping, noisy, nearly-flat surfaces without any prominent landmarks. However, performing intra-operative registration based on surface acquisitions bears several advantages over current approaches. The main advantage is the independence of artificial fiducial markers, which are attached to the patient.

Current approaches for surface-based intra-operative registration only deal with deformations and distortions at local scales, while the rough alignment between the surfaces must be performed manually. Current approaches for automatic surface matching focus on the registration of shapes with high degrees of isotropic deformation. However, these techniques mostly rely on the existence of prominent landmarks or spatial landmark configurations that are uniquely identifiable.

In this work, two approaches for automatic registration of surfaces in intra-operative conditions were presented. In the first approach, the surfaces were segmented into regions of similar surface characteristics, such as curvature. This segmentation was used to build an adjacency graph representing these regions and their neighborhood relations. Graph matching was then employed to solve the registration problem. Having graph representations of the surfaces at hand also allowed for post-processing of the correspondences after optimization, with wrong correspondences being identified based on the consistency of correspondences among neighborhoods. Although the graph matching optimization procedures proved to be reliable even for the matching of very small sub-surfaces to the complete surfaces, the quality of the registration deteriorated very fast with the inclusion of structural errors between the graphs. Structural errors between graphs occurred due to differences between surface segmentations. The ideal segmentation method for surface-based intra-operative registration should thus be able to maximize intra-region similarities and

minimize inter-region similarities, while creating enough neighborhood relations in order to avoid ambiguous graph topologies¹. Furthermore, the segmentation method must be able to generate similar segmentations even when only partial surfaces are given. It must also be robust across surfaces, even in the presence of noise, distortions or deformations. The design of such a segmentation method is not trivial. Although graph-based surface registration showed to be very robust for the registration of feature-less, partial surfaces, the problem of encountering reliable features shifts to encountering reliable regions, which was shown to be hard under the given intra-operative conditions. Nevertheless, this approach seems promising, and should be further investigated. Its adaptation to the registration of isometrically deformed surfaces is trivial, and good results are expected. In these cases most regions remain unchanged and the neighborhoods remain constant. Graph matching thus shows very high potential.

Regarding the second surface matching approach, the influence of the error metric to be optimized and the consistent selection of features across surfaces² was investigated for point-based approaches. As most methods cast the surface matching problem as a *quadratic assignment problem* (QAP), having a second order term that ensures that distances between pairs of landmarks on the source surface are similar to the distances between their corresponding points on the target surface, consistent feature selection across surfaces becomes essential for two reasons: First, to avoid correspondence ambiguities, as there are several spatial configurations of landmarks that would fit to each other in nearly flat regions, but very few configurations would fit to each other when dealing only with prominent points. Second, to reduce the search-space and computation times, as the solution of a QAP is NP-hard. The consistent selection of features is difficult for surfaces of interest in intra-operative registration, due to noise, distortions, and the fact that they are nearly flat. As a consequence, the search for a set of correspondences between surface becomes error-prone. In order to overcome these issues, we incorporated a measure of landmark spatial configuration reliability in the error metric, along with several measures for a faster optimization, thus making the consistent selection of features across surfaces irrelevant. The incorporation of the configuration error in the error metric showed a signifi-

¹ For example, a very large region incorporates a lot of geometric information within itself, and it has many neighbors composed by smaller regions within itself, thus forming a star topology. In this case, as neighborhood information is scarce, only the node representing the center of the star can be uniquely identified, while other correspondences become very ambiguous.

² Consistent feature selection across surfaces refers to the ability of selecting prominent and unique points on the same locations for two different surfaces.

cant increase in the accuracy of the registration and the robustness to variations of the method's parameters. Although registrations were obtained within one minute, computation times can potentially be further improved.

In summary, this thesis presents two different approaches for *automatic* surface matching, *i.e.*, without making any assumptions about the initial positioning of the surfaces. Throughout several experiments, which aimed to simulate intra-operative conditions, the presented approaches proved to be *robust, accurate, and fulfilled the computation time requirements* of intra-operative registration. Finally, as no assumptions were made about the surfaces or the object they represent, the presented approaches for surface matching are *generic* and may thus be employed in diverse situations requiring the registration of nearly flat, feature-less surfaces. The applicability to various types of objects was demonstrated throughout the experimental evaluations, by using several models from different sources. Regarding the mentioned achievements, all objectives initially stated in the introduction of this work have been met.

Nevertheless, there is still a long way between the methods presented here and a fully automatic, surface-based surgery guidance system. As shown in Chap. 2, several components are required to cover the entire pipeline. The methods presented here have been designed to deal with several central issues that occur in intra-operative environments. However, these environments are generally very dynamic and a prediction of all potential problems is impossible.

Regarding the field of partial surface matching, the methods described in these thesis represent an important effort towards the registration of partially overlapping, nearly flat, feature-less surfaces. This problem has hardly been investigated in the literature, as most methods always assume that a set of unique landmarks can always be identified. Furthermore, several fields, such as object reconstruction and shape retrieval from databases, can profit from the achievements presented here.

7.1 SUMMARY OF CONTRIBUTIONS

The specific contributions of this work are:

1. Two different approaches for *automatic* surface matching in intra-operative conditions, *i.e.*, for the registration of multi-modal, partially overlapping, nearly flat, noisy, feature-less surfaces.

- a) A region-based approach, where registration is cast as a graph matching problem. Graph matching is solved by an iterative scoring of neighborhood similarities and the computation of correspondences as a *linear assignment problem* (LAP), thus allowing the registration to be solved in polynomial time. Taking advantage of the graph representations, post-processing methods were employed to identify false assignments and to increase the number of correspondences.
 - b) A point-based approach that does not rely on the consistent selection of features across surfaces. It incorporates a metric for the computation of a landmark spatial configuration reliability (configuration error), in order to compensate for the lack of prominent and unique locations.
2. A surface descriptor for the representation of noisy point neighborhoods, which is robust to distortions and deformations.
3. A data structure for the representation of surfaces as polygonal meshes ensuring topological consistency. The data structure provides an efficient representation of regular surfaces (2-manifolds) and non-regular surfaces (2-manifolds with boundaries and 2-pseudomanifolds), allowing the retrieval of adjacency information between mesh components in linear time, identification of boundaries and anomalies in constant time, and iteration through mesh components in linear time. Furthermore, several manipulation operators are presented, which guarantee topological consistency after modification.
4. A framework for the assessment of differences between multimodal surfaces, not only focusing on geometrical differences, but also on differences between surface properties, such as normals and curvatures. In contrast to other frameworks, this framework does not consider the Euclidean closest point as an indication of surface correspondences. Instead it assumes that the correct correspondences might lie further away due to distortions and deformations. An anisotropic metric of distance computation is employed in order to overcome this issue.
5. An assessment of differences between surfaces representing real organs generated by a time-of-flight (ToF) range scanner and by a computed tomography (CT), which represent the main surface generation modalities employed in the experiments described in this thesis. High degrees of distortions were observed, even in the statical setting employed for this assessment.

6. A comprehensive survey of methods for automatic surface matching, and a classification according to the descriptor, error metric, optimization approach and the scale of registration (rough or fine) used. The applicability of the surface registration under intra-operative conditions was also considered.

7.2 FUTURE WORK

This thesis goes to great lengths regarding surface registration. The work presented here further provides ideas for future work, arising from the observations made throughout the development of this research:

- *Speed-up*: In order to decrease computation times, several authors have proposed hierarchical approaches to compute correspondences. Although hierarchy is already exploited to some extent in this work, new concepts of multi-scale/multi-resolution surface analysis [32] might assist the achievement of an hierarchical algorithm not only based on feature confidence measures, but also on the representation of surface information in multiple levels-of-details. Furthermore, as correspondence search is performed in a greedy manner, parallelization is applicable to this minimization problem. Technologies such as *graphic processing units* (GPUs) can be employed for significantly faster optimization.
- *Non-rigidity*: Throughout this work, even in experiments that include surface deformations, rigid mappings were employed in order to compute fitting errors and final registrations. The introduction of non-rigid mappings in the presented work is straightforward. Non-rigid mappings might also increase the robustness and accuracy of fitting errors if the correct class of mapping is chosen. However, for the computation of a non-rigid transformation as final registration, a sparse set of correspondences, as provided by the methods presented here, is not sufficient, as interpolations and extrapolation would be required for several parts of the surfaces. In this case, non-rigidity can be solved by fine alignment procedures or by the computation of dense correspondence sets.
- *Dense correspondences*: Several works focus on the computation of dense correspondence sets given a set of sparse correspondences. Apart from the computation of an initial alignment, dense correspondences further allow the computation of non-

rigid mappings that model local scale deformations and distortions. As for many methods dealing with the computation of dense correspondences, only the set of initial correspondences is required as input, the incorporation of such a method in the registration approaches presented here is straightforward. See Sec. 3.4.2 for more details on these methods.

- *Robustness*: Most surface matching algorithms only consider descriptor similarity and difference between geodesic distances among pairs of corresponding points in their error metric in order to identify fitting landmark spatial configurations. Here we show that the incorporation of a spatial configuration reliability measure can significantly improve the robustness and accuracy of the registration. Although the configuration error did provide significant improvements, it considers spatial configurations only. Two aspects can be further regarded:
 - Instead of computing the configuration error in the geometrical space only, further types of configuration errors may be incorporated in the error metric. For example, the application of the presented expression for the computation of configuration error in geometric space may be directly employed for the computation of such an error in the descriptor space. In this case, the selected correspondence configurations are not only geometrically widespread and non-planar, but so are their descriptors as well, ensuring higher variability in the descriptor space and better overall reliability.
 - The shape information between two landmarks may be incorporated, which is neglected in error metrics. In order to illustrate this idea, let us assume two landmarks on the source surface and their corresponding points on the target surface. In order for this correspondences to be correct, not only the geodesic distance between source landmarks must be similar to the distance between their corresponding points, but also the shape between the source landmarks must be similar to the shape between their corresponding points. One alternative for the incorporation of this shape information is to draw a direct geodesic line above the surface connecting the two landmarks. This line can be viewed as a 2D curve. The 2D curves spanned by two source landmarks and their corresponding points must be similar. Computation of curve similarities can be

performed, for example, by the maximization of Pearson correlation [86], among other approaches [264, 70].

ROTATION INVARIANCE OF THE LOCAL COORDINATE SYSTEM

We show here that, if a rotation is applied to the vertices' unit normals within the spherical support volume of a particular vertex, the eigenvectors of the approximation of the covariance matrix (as presented in Sec. 6.2.2) rotate by the same amount.

Proof. The approximation of the covariance matrix of the unit normals set $\{\vec{n}_0, \dots, \vec{n}_N\}$, which we denote C , is *similar* to the approximation of the covariance matrix of this normals set rotated by a rotation matrix R , which we denote C_R , thus having the same eigenvalues.

Let $\{\vec{m}_0, \dots, \vec{m}_N\}$ represent the set $\{\vec{n}_0, \dots, \vec{n}_N\}$ rotated by a rotation matrix R . Two matrices C and C_R are similar if $C = S^{-1}C_R S$, for an invertible matrix S . Matrices C and C_R are obtained as follows:

$$C = \sum_{i=0}^N A(v_i) \vec{n}_i \vec{n}_i^T \quad (\text{A.1})$$

$$C_R = \sum_{i=0}^N A(v_i) \vec{m}_i \vec{m}_i^T \quad (\text{A.2})$$

where $A(v_i)$ denotes the Voronoi area of the vertex to which the normal \vec{n}_i (and also \vec{m}_i) belongs to. For the purpose of showing the rotation property of the approximation of the covariance matrix, we can consider all Voronoi areas equal to 1. As the $\{\vec{n}_0, \dots, \vec{n}_N\}$ can be obtained by the inverse rotation R^{-1} of the set $\{\vec{m}_0, \dots, \vec{m}_N\}$, we know that:

$$\sum_{i=0}^N \vec{n}_i \vec{n}_i^T = \sum_{i=0}^N R^{-1} \vec{m}_i \vec{m}_i^T (R^{-1})^T \quad (\text{A.3})$$

Because of the fact that the rotation matrix is orthogonal, and thus $R^{-1} = R^T$, we obtain $(R^{-1})^T = (R^T)^T = R$. This allows us to rewrite equation A.3 as:

$$\sum_{i=0}^N \vec{n}_i \vec{n}_i^T = \sum_{i=0}^N R^{-1} \vec{m}_i \vec{m}_i^T R \quad (\text{A.4})$$

$$= R^{-1} \left(\sum_{i=0}^N \vec{m}_i \vec{m}_i^T \right) R \quad (\text{A.5})$$

and therefore:

$$C = R^{-1}C_R R \quad (\text{A.6})$$

thus showing that C and C_R are similar and have the same eigenvalues. □

Proof. The eigenvectors of C_R are equal to the eigenvectors of C rotated by R , *i.e.*, if \vec{v} is an eigenvector from C with eigenvalue λ , then $R\vec{v}$ is an eigenvector from C_R with same eigenvalue:

$$Cv = \lambda v \rightarrow C_R Rv = \lambda Rv \quad (\text{A.7})$$

We have shown before that C and C_R are similar and therefore have the same eigenvalues. In the following equations, we show how to obtain the implication shown in equation [A.7](#):

$$Cv = \lambda v \quad (\text{A.8})$$

By the application of equation [A.6](#):

$$R^{-1}C_R Rv = \lambda v \quad (\text{A.9})$$

$$C_R Rv = \lambda Rv \quad (\text{A.10})$$

thus showing that the implication in equation [A.7](#) is true. □

DERIVATION OF THE AVERAGE TARGET REGISTRATION ERROR

An expression for the estimation of the *target registration error* (TRE) at a spatial position \mathbf{x} , given the approximately known position of a set of landmarks L , was presented by Fitzpatrick et al. [102], and is as follows:

$$\langle \text{TRE}^2(\mathbf{x}, L) \rangle = \frac{\langle \text{FLE}^2(L) \rangle}{|L|} \left(1 + \frac{1}{3} \sum_{k=1}^3 \frac{d_k^2}{f_k^2} \right) \quad (\text{B.1})$$

where d_k denotes the distance from the target to the principal axis \vec{v}_k of L , and f_k the root-mean-square distance of the landmarks to their principal axis \vec{v}_k . The *fiducial localization error* (FLE) denotes the error of locating the landmarks.

The *average target registration error* is defined as the average of the TRE for every location within a bounding sphere around the landmarks, centered on their centroid. To obtain an expression for the target registration error, we first describe the expression for the TRE with a target given in spherical coordinates (r, θ, ϕ) , assuming, without any loss of generality, that the centroid of the landmarks lie at the origin:

$$\langle \text{TRE}^2(r, \theta, \phi, L) \rangle = \frac{\langle \text{FLE}^2(L) \rangle}{|L|} \left(1 + \frac{1}{3} \sum_{k=1}^3 \frac{d(r, \theta, \phi, \vec{v}_k)}{f_k^2} \right) \quad (\text{B.2})$$

$$d(r, \theta, \phi, \vec{v}) = \frac{|\vec{v} \times \vec{x}|^2}{|\vec{v}|^2} \quad (\text{B.3})$$

where \vec{x} is a vector in Cartesian coordinates from the position (r, θ, ϕ) to the origin (centroid of the landmarks). The average TRE is estimated by the integration of the TRE expression within a sphere of radius R and dividing it by the volume V of these sphere, as follows:

$$\begin{aligned} \langle \overline{\text{TRE}^2(L)} \rangle &= \frac{1}{V} \int_R \int_0^\pi \int_0^{2\pi} \langle \text{TRE}^2(r, \theta, \phi, L) \rangle \, dr \, d\theta \, d\phi \\ &= \frac{1}{V} \int_R \int_0^\pi \int_0^{2\pi} \frac{\langle \text{FLE}^2(L) \rangle}{|L|} \left(1 + \frac{1}{3} \sum_{k=1}^3 \frac{d(r, \theta, \phi, \vec{v}_k)}{f_k^2} \right) \, dr \, d\theta \, d\phi \\ &= \frac{1}{V} \frac{\langle \text{FLE}^2(L) \rangle}{|L|} \left(\int_R \int_0^\pi \int_0^{2\pi} \, dr \, d\theta \, d\phi + \frac{1}{3} \int_R \int_0^\pi \int_0^{2\pi} \sum_{k=1}^3 \frac{d(r, \theta, \phi, \vec{v}_k)}{f_k^2} \, dr \, d\theta \, d\phi \right) \\ &= \frac{1}{V} \frac{\langle \text{FLE}^2(L) \rangle}{|L|} \left(\frac{4\pi R^3}{3} + \frac{1}{3} \sum_{k=1}^3 \frac{\int_R \int_0^\pi \int_0^{2\pi} d(r, \theta, \phi, \vec{v}_k) \, dr \, d\theta \, d\phi}{f_k^2} \right) \end{aligned} \quad (\text{B.4})$$

We have the expression $\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} d(r, \theta, \phi, \vec{v}_k) dr d\theta d\phi$ left to solve, which is carried out as follows:

$$\begin{aligned}
& \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} d(r, \theta, \phi, \vec{v}) dr d\theta d\phi \\
&= \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \frac{|\vec{v} \times \vec{x}|^2}{|\vec{v}|^2} dr d\theta d\phi \\
&= \frac{1}{|\vec{v}|^2} \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} |\vec{v} \times \vec{x}|^2 dr d\theta d\phi \\
&= \frac{1}{|\vec{v}|^2} \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} (\vec{v}_x \vec{x}_y - \vec{v}_y \vec{x}_x)^2 + \\
&\quad (\vec{v}_y \vec{x}_z - \vec{v}_z \vec{x}_y)^2 + \\
&\quad (\vec{v}_z \vec{x}_x - \vec{v}_x \vec{x}_z)^2 dr d\theta d\phi \\
&= \frac{1}{|\vec{v}|^2} \left(\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} (\vec{v}_x \vec{x}_y - \vec{v}_y \vec{x}_x)^2 dr d\theta d\phi + \right. \\
&\quad \left. \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} (\vec{v}_y \vec{x}_z - \vec{v}_z \vec{x}_y)^2 dr d\theta d\phi + \right. \\
&\quad \left. \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} (\vec{v}_z \vec{x}_x - \vec{v}_x \vec{x}_z)^2 dr d\theta d\phi \right) \text{ (B.5)}
\end{aligned}$$

Three terms in the form $\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} (\vec{v}_i \vec{x}_j - \vec{v}_j \vec{x}_i)^2 dr d\theta d\phi$ are left, which expand to:

$$\begin{aligned}
& \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} (\vec{v}_i \vec{x}_j - \vec{v}_j \vec{x}_i)^2 dr d\theta d\phi \\
&= \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \vec{v}_i^2 \vec{x}_j^2 - 2\vec{v}_i \vec{v}_j \vec{x}_i \vec{x}_j + \vec{v}_j^2 \vec{x}_i^2 dr d\theta d\phi \\
&= \vec{v}_i^2 \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \vec{x}_j^2 dr d\theta d\phi - \\
&\quad 2\vec{v}_i \vec{v}_j \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \vec{x}_i \vec{x}_j dr d\theta d\phi + \\
&\quad \vec{v}_j^2 \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \vec{x}_i^2 dr d\theta d\phi \quad \text{(B.6)}
\end{aligned}$$

We must then solve the integrals for \vec{x}_x^2 , \vec{x}_y^2 , \vec{x}_z^2 , $\vec{x}_x \vec{x}_y$, $\vec{x}_x \vec{x}_z$ and $\vec{x}_y \vec{x}_z$. By converting the vector coordinates back to spherical coordinates, we get:

$$\begin{aligned}
\vec{x}_x &= R \sin \theta \cos \phi \\
\vec{x}_y &= R \sin \theta \sin \phi \\
\vec{x}_z &= R \cos \theta
\end{aligned} \quad \text{(B.7)}$$

By solving the different terms we obtain:

$$\begin{aligned}
\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \bar{x}_x^2 dr d\theta d\phi &= \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} r^2 \sin^2 \theta \cos^2 \phi dr d\theta d\phi \\
&= \frac{\mathbb{R}^3}{3} \int_0^\pi \int_0^{2\pi} \sin^2 \theta \cos^2 \phi d\theta d\phi \\
&= \frac{\mathbb{R}^3}{3} \frac{\pi^2}{2}
\end{aligned} \tag{B.8}$$

$$\begin{aligned}
\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \bar{x}_y^2 dr d\theta d\phi &= \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} R^2 \sin^2 \theta \sin^2 \phi dr d\theta d\phi \\
&= \frac{\mathbb{R}^3}{3} \frac{\pi^2}{2}
\end{aligned} \tag{B.9}$$

$$\begin{aligned}
\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \bar{x}_z^2 dr d\theta d\phi &= \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} R^2 \cos^2 \theta dr d\theta d\phi \\
&= \frac{\mathbb{R}^3}{3} \pi^2
\end{aligned} \tag{B.10}$$

$$\begin{aligned}
\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \bar{x}_x \bar{x}_y dr d\theta d\phi &= \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} R^2 \sin^2 \theta \sin \phi \cos \phi dr d\theta d\phi \\
&= \frac{\mathbb{R}^3}{3} \pi^2 \int_0^{2\pi} \sin \phi \cos \phi d\phi \\
&= 0
\end{aligned} \tag{B.11}$$

$$\begin{aligned}
\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \bar{x}_x \bar{x}_z dr d\theta d\phi &= \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} R^2 \sin \theta \cos \theta \cos \phi dr d\theta d\phi \\
&= 0
\end{aligned} \tag{B.12}$$

$$\begin{aligned}
\int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} \bar{x}_y \bar{x}_z dr d\theta d\phi &= \int_{\mathbb{R}} \int_0^\pi \int_0^{2\pi} R^2 \sin \theta \cos \theta \sin \phi dr d\theta d\phi \\
&= 0
\end{aligned} \tag{B.13}$$

We can now substitute these equations back in Eq. B.5. Before that, let us make an observation: The sum of the distances of all points within a sphere to any axis that passes through the center of the sphere is constant. Therefore, we only have to solve the equation for a single axis, with any direction vector. We assume the direction vector of this axis to be $(1, 0, 0)$. This leaves us with:

$$\begin{aligned}
\frac{1}{|v|^2} (\bar{v}_x^2 \int_{\mathbf{R}} \int_0^\pi \int_0^{2\pi} \bar{x}_y^2 dr d\theta d\phi + \bar{v}_x^2 \int_r \int_0^\pi \int_0^{2\pi} \bar{x}_z^2 dr d\theta d\phi) \\
= \frac{\bar{v}_x^2}{|v|^2} \left(\frac{R^3}{3} \frac{\pi^2}{2} + \frac{R^3}{3} \pi^2 \right) \\
= \frac{R^3}{3} \left(\frac{\pi^2}{2} + \pi^2 \right) \\
= \frac{\pi^2 R^3}{2} \quad (\text{B.14})
\end{aligned}$$

We now substitute these result in the average TRE expression (Eq. B.4):

$$\begin{aligned}
\langle \overline{\text{TRE}^2(\text{L})} \rangle &= \frac{1}{V} \frac{\langle \text{FLE}^2(\text{L}) \rangle}{|\text{L}|} \left(\frac{4\pi R^3}{3} + \frac{1}{3} \sum_{k=1}^3 \frac{\pi^2 R^3}{f_k^2} \right) \\
&= \frac{3}{4\pi r^3} \frac{\langle \text{FLE}^2(\text{L}) \rangle}{|\text{L}|} \left(\frac{4\pi R^3}{3} + \frac{\pi^2 R^3}{6} \sum_{k=1}^3 \frac{1}{f_k^2} \right) \\
&= \frac{\langle \text{FLE}^2(\text{L}) \rangle}{|\text{L}|} \left(1 + \frac{\pi}{8} \sum_{k=1}^3 \frac{1}{f_k^2} \right) \quad (\text{B.15})
\end{aligned}$$

BIBLIOGRAPHY

- [1] A. M. Abdulkader. *Parallel Algorithms for Labelled Graph Matching*. PhD thesis, Colorado School of Mines, 1998. (Cited on page 83.)
- [2] D. Aiger, N. J. Mitra, and D. Cohen-Or. 4-points congruent sets for robust pairwise surface registration. *ACM TOG*, 27(3):1–10, 2008. (Cited on pages 16, 30, 33, 34, and 39.)
- [3] E. Akleman and J. Chen. Guaranteeing the 2-manifold property for meshes with doubly linked face list. *Int. J. Shape Mod.*, 5(2): 149–177, 1999. (Cited on pages xix, 13, 42, 44, 45, and 68.)
- [4] E. Akleman and J. Chen. Regular meshes. In *Proc. Solid and physical modeling*, pages 213–219, 2005. (Cited on page 42.)
- [5] E. Akleman and J. Chen. Regular mesh construction algorithms using regular handles. In *Proc. Shape Modeling and Applications*, pages 27–36, 2006. (Cited on page 42.)
- [6] E. Akleman and V. Srinivasan. Honeycomb subdivision. In *Proc. Computer and Information Sciences*, pages 137–141, 2002. (Cited on page 42.)
- [7] E. Akleman, J. Chen, and V. Srinivasan. A new paradigm for changing topology during subdivision modeling. In *Proc. Pacific Graphics*, pages 192–201, 2000. (Cited on page 42.)
- [8] E. Akleman, J. Chen, V. Srinivasan, and F. Eryoldas. A new corner cutting scheme with tension and handle-face reconstruction. *International Journal of Shape Modeling*, 7(2):111–121, 2001. (Cited on page 42.)
- [9] E. Akleman, J. Chen, and V. Srinivasan. A minimal and complete set of operators for the development of robust manifold mesh modelers. *Graphical Models*, 65(5):286–304, 2003. (Cited on pages 42, 45, and 46.)
- [10] E. Akleman, V. Srinivasan, and J. Chen. Interactive rind modeling. In *Proc. Int. Conf. Shape Modeling and Application*, pages 23–31, 2003. (Cited on page 42.)

- [11] E. Akleman, V. Srinivasan, Z. Melek, and P. Edmundson. Semiregular pentagonal subdivisions. In *Proc. Shape Modeling*, pages 110–118, 2004. (Cited on page 42.)
- [12] E. Akleman, V. Srinivasan, and E. Mandal. Remeshing schemes for semi-regular tilings. In *Proc. Shape Modeling*, 2005. (Cited on page 42.)
- [13] E. Akleman, V. Srinivasan, J. Chen, D. V. Morris, and S. T. Tett. Topmod3d: An interactive topological mesh modeler. In *Proc. CGI*, 2008. (Cited on pages 42 and 69.)
- [14] M. Alexa, D. Cohen-Or, and D. Levin. As-rigid-as-possible shape interpolation. In *Proc. SIGGRAPH*, pages 157–164, 2000. (Cited on page 35.)
- [15] B. Amberg, S. Romdhani, and T. Vetter. Optimal step nonrigid ICP algorithms for surface registration. In *Proc. CVPR*, 2007. (Cited on page 36.)
- [16] L. Armesto, J. Minguéz, and L. Montesano. A generalization of the metric-based iterative closest point technique for 3D scan matching. In *Proc. ICRA*, pages 1367–1372, 2010. (Cited on page 36.)
- [17] N. Aspert, D. Santa-Cruz, and T. Ebrahimi. MESH: Measuring errors between surfaces using the Hausdorff distance. In *Proc. IEEE ICME*, pages 705–708, 2002. (Cited on page 71.)
- [18] M. Attene, S. Katz, M. Mortara, G. Patane, M. Spagnuolo, and A. Tal. Mesh segmentation - a comparative study. In *Proc. IEEE Shape Mod. & Apps.*, pages 14–25, 2006. (Cited on page 84.)
- [19] O. K.-C. Au, C.-L. Tai, H.-K. Chu, D. Cohen-Or, and T.-Y. Lee. Skeleton extraction by mesh contraction. *ACM TOG*, 27:44:1–44:10, 2008. (Cited on pages 20 and 33.)
- [20] O. K.-C. Au, C.-L. Tai, D. Cohen-Or, Y. Zheng, and H. Fu. Electors voting for fast automatic shape correspondence. *Comp. Graph. Forum*, 29(2):645–654, 2010. (Cited on pages xviii, 21, 30, 33, and 39.)
- [21] M. A. Audette and T. Peters. Level-set surface segmentation and registration for computing intrasurgical deformations. In *Proc. SPIE Med. Img.: Img. Proc.*, 1999. (Cited on pages 3, 9, and 38.)

- [22] M. A. Audette, F. P. Ferrie, and T. M. Peters. An algorithmic overview of surface registration techniques for medical imaging. *MIA*, 4:201–217, 2000. (Cited on pages 15, 37, and 82.)
- [23] P. Bao, T. K. Sinha, C.-C. R. Chen, J. R. Warmath, R. L. Galloway, and A. J. Herline. A prototype ultrasound-guided laparoscopic radiofrequency ablation system. *Surg. Endosc.*, 21(1):74–79, 2007. (Cited on page 3.)
- [24] B. G. Baumgart. Winged edge polyhedron representation. Technical report, Stanford University, Stanford, CA, USA, 1972. CS-TR-72-320. (Cited on pages 13, 41, and 56.)
- [25] B. G. Baumgart. *Geometric Modelling for Computer Vision*. PhD thesis, Stanford University, 1974. (Cited on pages 41 and 56.)
- [26] B. G. Baumgart. A polyhedron representation for computer vision. In *Nat. Comp. Conf.*, pages 589–596, 1975. (Cited on pages 13, 41, and 56.)
- [27] M. Baumhauer, M. Feuerstein, H.-P. Meinzer, and J. Rassweiler. Navigation in endoscopic soft tissue surgery: perspectives and limitations. *J. Endourol.*, 22(4):751–766, 2008. (Cited on page 2.)
- [28] M. Baumhauer, T. Simpfendorfer, D. Müller-Stich, D. Teber, C. Gutt, J. Rassweiler, H.-P. Meinzer, and I. Wolf. Soft tissue navigation for laparoscopic partial nephrectomy. *Int. J. CARS*, 3(3):307–314, 2008. (Cited on page 2.)
- [29] A. F. Beardon. *Algebra and Geometry*. Cambridge University, 1 edition, 2005. (Cited on page 21.)
- [30] E. Bengoetxea. *Inexact Graph Matching Using Estimation of Distribution Algorithms*. PhD thesis, Ecole Nationale Supérieure des Télécommunications (ENST), Paris, France, 2002. (Cited on page 83.)
- [31] A. B. Benincasa, C. L. W., H. S. D., and G. R. L. Feasibility study for image-guided kidney surgery: assessment of required intra-operative surface for accurate physical to image space registrations. *Med. Phys.*, 35(9):4251–4261, 2008. (Cited on pages 3, 9, and 38.)
- [32] M. Berger, L. G. Nonato, V. Pascucci, and C. T. Silva. Fiedler trees for multiscale surface analysis. *Comp. & Graph.*, 34:272–281, 2012. (Cited on page 151.)

- [33] P. J. Besl and R. C. Jain. Invariant surface characteristics for 3d object recognition in range images. *Comp. Vis., Graph., & Img. Proc.*, 33(1):33–80, 1986. ISSN 0734-189X. doi: [http://dx.doi.org/10.1016/0734-189X\(86\)90220-3](http://dx.doi.org/10.1016/0734-189X(86)90220-3). (Cited on pages [83](#), [103](#), [104](#), and [136](#).)
- [34] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE TPAMI*, 14:239–256, 1992. (Cited on pages [17](#), [26](#), [31](#), [35](#), and [36](#).)
- [35] C. Bettschart, A. Kruse, F. Matthews, W. Zemmann, J. A. Obwegeser, K. W. Grätz, and H.-T. Lübbers. Point-to-point registration with mandibulo-maxillary splint in open and closed jaw position. evaluation of registration accuracy for computer-aided surgery of the mandible. *J. Cranio-Maxillofac. Surg.*, 2011. ISSN 1010-5182. In press. (Cited on page [9](#).)
- [36] F. Blais. Review of 20 years of range sensor development. *J. Elect. Img.*, 13(1):231–240, 2004. (Cited on page [10](#).)
- [37] D. K. Blandford, G. E. Blelloch, D. E. Cardoze, and C. Kadow. Compact representations of simplicial meshes in two and three dimensions. *International Journal of Computational Geometry and Applications*, 15(1):135–146, 2005. (Cited on page [41](#).)
- [38] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE TPAMI*, 11:567–585, 1989. (Cited on page [37](#).)
- [39] I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 1 edition, 1996. (Cited on page [34](#).)
- [40] E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Disc. App. Math.*, 123:155–225, 2002. (Cited on page [34](#).)
- [41] F. Bourgeois and J.-C. Lassalle. An extension of the munkres algorithm for the assignment problem to rectangular matrices. *Comm. ACM*, 14(12):802–804, 1971. (Cited on page [91](#).)
- [42] S. Boyd and L. Vandenbergue. *Convex optimization*. Cambridge Univ. Press, 2006. (Cited on page [32](#).)
- [43] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE TPAMI*, 26:1124–1137, 2004. (Cited on page [34](#).)

- [44] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *PNAS*, 103:1168–1172, 2006. (Cited on pages 17 and 34.)
- [45] A. M. Bronstein, M. M. Bronstein, B. Bustos, U. Castellani, M. Crisani, B. Falcidieno, L. J. Guibas, I. Kokkinos, V. Murino, M. Ovsjanikov, G. Patan , I. Sipiran, M. Spagnuolo, and J. Sun. SHREC 2010: robust feature detection and description benchmark. In *Proc. 3DOR*, 2010. (Cited on page 22.)
- [46] A. M. Bronstein, M. M. Bronstein, L. J. Guibas, and M. Ovsjanikov. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM TOG*, 30:1:1–1:20, 2011. (Cited on page 20.)
- [47] R. Burkard, M. Dell’Amico, and S. Martello. *Assignment Problems*. Society for Industrial and Applied Mathematics, 1 edition, 2009. (Cited on pages 26 and 27.)
- [48] R. E. Burkard and E. ela. Linear assignment problems and extensions. *Handbook of Combinatorial Optimization*, 4(1):1–54, 1999. (Cited on page 31.)
- [49] B. Bustos, D. A. Keim, D. Saupe, T. Schreck, and D. V. Vranić. Feature-based similarity search in 3d object databases. *ACM Comput. Surv.*, 37:345–387, 2005. (Cited on page 22.)
- [50] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, and Z. Su. Point cloud skeletons via laplacian based contraction. In *Proc. IEEE SMI*, 2010. (Cited on page 20.)
- [51] D. Cash, M. Miga, S. Glasgow, B. Dawant, L. Clements, Z. Cao, R. Galloway, and W. Chapman. Concepts and preliminary data toward the realization of image-guided liver surgery. *J. Gastrointest. Surg.*, 11(7):844–859, 2007. (Cited on pages 3, 9, and 38.)
- [52] D. M. Cash, M. I. Miga, T. K. Sinha, R. L. Galloway, and W. C. Chapman. Compensating for intraoperative soft-tissue deformations using incomplete surface data and finite elements. *IEEE TMI*, 24(11):1479–1491, 2005. (Cited on pages 3, 9, 13, 36, 38, 39, and 142.)
- [53] U. Castellani, M. Cristani, S. Fantoni, and V. Murino. Sparse points matching by combining 3d mesh saliency with statistical descriptors. *Comp. Graph. Forum*, 27(2):643–652, 2008. (Cited on page 25.)

- [54] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Comp. Aided Geo. Design*, 22(2):121–146, 2005. (Cited on pages [xvii](#), [18](#), [19](#), and [126](#).)
- [55] W. Chang and M. Zwicker. Automatic registration for articulated shapes. In *Proc. SGP*, 2008. (Cited on page [27](#).)
- [56] W. Charlesworth and D. Anderson. Applications of non-manifold topology. In *Proc. Computers in Engineering Conf. and Engineering Database Symp.*, pages 103–112, 1995. (Cited on page [41](#).)
- [57] G. Charpiat, O. Faugeras, and R. Keriven. Approximations of shape metrics and application to shape warping and empirical shape statistics. *Found. Comput. Math.*, 5:1–58, 2005. (Cited on page [30](#).)
- [58] J. Chen. Algorithmic graph embeddings. *Theo. Comp. Sc.*, 181(2):247–266, 1997. (Cited on pages [13](#) and [42](#).)
- [59] X. Chen, A. Golovinskiy, and T. Funkhouser. A benchmark for 3D mesh segmentation. In *Proc. SIGGRAPH*, 2009. (Cited on page [84](#).)
- [60] D. Chetverikov, D. Stepanov, and P. Krsek. Robust euclidean alignment of 3D point sets: the trimmed iterative closest point algorithm. *Img*, 23:299–309, 2005. (Cited on page [36](#).)
- [61] M. K. Chung, K. M. Dalton, and R. J. Davidson. Encoding neuroanatomical information using weighted spherical harmonic representation. In *Proc. IEEE SSP*, pages 146–150, 2007. (Cited on pages [xviii](#) and [21](#).)
- [62] M. K. Chung, R. Hartley, K. M. Dalton, and R. J. Davidson. Encoding cortical surfaces by spherical harmonics. *Statistica Sinica*, 18:1269–1291, 2008. (Cited on page [21](#).)
- [63] P. G. Ciarlet. *An Introduction to Differential Geometry with Applications to Elasticity*. Springer, 1 edition, 2006. (Cited on page [29](#).)
- [64] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum*, 17(2):167–174, 1998. (Cited on page [71](#).)
- [65] L. W. Clements, D. M. Cash, W. C. Chapman, R. L. Galloway, and M. I. Miga. Robust surface registration using salient anatomical features in image-guided liver surgery. In *Proc. SPIE*

- Med. Img.: Vis. Img.-Guided Proc. Disp.*, 2006. (Cited on pages [3](#), [9](#), [36](#), and [38](#).)
- [66] L. W. Clements, W. C. Chapman, B. M. Dawant, R. L. Galloy, and M. I. Miga. Robust surface registration using salient anatomical features for image-guided liver surgery: algorithm and validation. *Med. Phys.*, 35(6):2528–2540, 2008. (Cited on pages [3](#), [9](#), [36](#), [38](#), and [142](#).)
- [67] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM TOG*, 23(3):905–914, 2004. (Cited on page [85](#).)
- [68] B. Combès and S. Prima. Prior affinity measures on matches for ICP-like nonlinear registration of free-form surfaces. In *Proc. ISBI*, pages 370–373, 2009. (Cited on page [36](#).)
- [69] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *Int. J. Patt. Rec.*, 18(3):265–298, 2004. (Cited on page [83](#).)
- [70] M. Cui, J. Femiani, J. Hu, P. Wonka, and A. Razdan. Curve matching for open 2d curves. *Patt. Rec. Letters*, 30:1–10, 2009. (Cited on page [153](#).)
- [71] Y. Cui, S. Schuon, C. Derek, S. Thrun, and C. Theobalt. 3d shape scanning with a time-of-flight camera. In *Proc. CVPR*, 2010. (Cited on page [10](#).)
- [72] B. Dagon, C. Baur, and V. Bettschart. Real-time update of 3d deformable models for computer aided liver surgery. In *ICPR*, pages 1–4, 2008. (Cited on page [13](#).)
- [73] A. Danilchenko and J. M. Fitzpatrick. General approach to first-order error prediction in rigid point registration. *IEEE TMI*, 30(3):679–693, 2011. (Cited on pages [2](#) and [128](#).)
- [74] L. De Floriani and A. Hui. A scalable data structure for three-dimensional non-manifold objects. In *Proc. Eurographics/SIGGRAPH Geometry processing*, pages 72–82, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. (Cited on page [41](#).)
- [75] L. De Floriani, M. M. Mesmoudi, F. Morando, and E. Puppo. Non-manifold decomposition in arbitrary dimensions. In *Proc. Discrete Geometry for Computer Imagery*, pages 69–80, 2002. ISBN 3-540-43380-5. (Cited on page [41](#).)

- [76] L. De Floriani, D. Greenfieldboyce, and A. Hui. A data structure for non-manifold simplicial d-complexes. In *Proc. Eurographics/SIGGRAPH Geometry processing*, pages 83–92, New York, NY, USA, 2004. ACM. ISBN 3-905673-13-4. doi: <http://doi.acm.org/10.1145/1057432.1057444>. (Cited on page 41.)
- [77] L. De Floriani, P. Magillo, E. Puppo, and D. Sobrero. A multi-resolution topological representation for non-manifold meshes. *Computer-Aided Design*, 36(2):141 – 159, 2004. ISSN 0010-4485. (Cited on page 41.)
- [78] A. Delaunoy, K. Fundana, E. Prados, and A. Heyden. Convex multi-region segmentation on manifolds. In *Proc. IEEE Comp. Vis.*, pages 662–669, 2009. (Cited on pages xxii and 85.)
- [79] H. Delingette. Triangular springs for modeling nonlinear membranes. *IEEE Trans. on Vis. Comp. Graph.*, 14:329–341, 2008. (Cited on page 29.)
- [80] M. Dell’Amico and S. Martello. The k-cardinality assignment problem. *Disc. App. Math.*, 76:103–121, 1997. (Cited on page 32.)
- [81] M. Dell’Amico and T. Paolo. Algorithms and codes for dense assignment problems: the state of the art. *Disc. App. Math.*, 100: 14–48, 2000. (Cited on pages 83 and 91.)
- [82] H. Desaulnier and N. Stewart. An extension of manifold boundary representation to r-sets. *ACM Trans. Graph.*, 11(1):40–60, 1992. (Cited on page 41.)
- [83] R. Detry and J. Piater. Continuous surface-point distributions for 3d object pose estimation and recognition. In *Proc. ACCV*, pages 572–585, 2010. (Cited on page 12.)
- [84] T. K. Dey, K. Li, C. Luo, P. Ranjan, I. Safa, and Y. Wang. Persistent heat signature for pose-oblivious matching of incomplete models. In *Proc. SGP*, 2010. (Cited on page 22.)
- [85] T. K. Dey, P. Ranjan, and Y. Wang. Convergence, stability, and discrete approximation of laplace spectra. In *Proc. ACM-SIAM Symp. on Disc. Alg.*, pages 650–663, 2010. (Cited on page 21.)
- [86] P. Di Lena and L. Margara. Optimal global alignment of signals by maximization of Pearson correlation. *Inf. Process. Lett.*, 110: 679–686, 2010. (Cited on page 153.)
- [87] P. Dierckx. *Curve and Surface Fitting with Splines*. Oxford Univ. Press, 1995. (Cited on page 37.)

- [88] S. DiMaio, T. Kapur, K. Cleary, S. Aylward, P. Kazanzides, K. Vosburgh, R. Ellis, J. Duncan, K. Farahani, H. Lemke, T. Peters, W. Lorensen, D. Gobbi, J. Haller, L. Clarke, S. Pizer, R. Taylor, R. Galloway, G. Fichtinger, N. Hata, K. Lawson, C. Tempany, and F. Jolesz. Challenges in image-guided therapy system design. *NeuroImage*, 37:144–151, 2007. (Cited on page 2.)
- [89] A. Dubrovina and R. Kimmel. Matching shapes by eigendecomposition of the Laplace-Beltrami operator. In *Proc. 3DPVT*, 2010. (Cited on pages [xix](#), [28](#), [29](#), [32](#), and [39](#).)
- [90] O. Duchenne, F. Bach, I. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. In *Proc. CVPR*, pages 1980–1987, 2009. (Cited on page [83](#).)
- [91] P. Dumpuri, L. W. Clements, B. M. Dawant, and M. I. Miga. Model-updated image-guided liver surgery: Preliminary results using surface characterization. *Prog. Biophys. Mol. Bio.*, 103(2-3):197 – 207, 2010. (Cited on pages [3](#), [9](#), [38](#), and [142](#).)
- [92] N. Dyn, D. Levine, and J. A. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Trans. Graph.*, 9:160–169, 1990. (Cited on page [73](#).)
- [93] I. Eckstein, J.-P. Pons, Y. Tong, C.-C. J. Kuo, and M. Desbrun. Generalized surface flows for mesh processing. In *Proc. SGP*, pages 183–192, 2007. (Cited on pages [17](#), [30](#), [35](#), [39](#), and [127](#).)
- [94] J. Eckstein and D. P. Bertsekas. An alternating direction method for linear programming. Technical Report LIDS-P;1967, Lab. for Inf. and Decision Sys., MIT, 1990. (Cited on page [34](#).)
- [95] R. S. J. Estépar, A. Brun, and C.-F. Westin. Robust generalized total least squares iterative closest point registration. In *Proc. MICCAI*, volume 3216, pages 234–241, 2004. (Cited on page [36](#).)
- [96] R. Ewers, K. Schicho, G. Undt, F. Wanschitz, M. Truppe, R. Seemann, and A. Wagner. Basic research and 12 years of clinical experience in computer-assisted navigation technology: A review. *Int. J. Oral Max. Surg.*, 34(1):1–8, 2005. (Cited on page [2](#).)
- [97] J. Feldmar and N. Ayache. Rigid, affine and locally affine registration of smooth surfaces. Technical Report 2220, INRIA, 1994. (Cited on page [37](#).)
- [98] M. Feuerstein, T. Mussack, S. M. Heining, and N. Navab. Intraoperative laparoscope augmentation for port placement and

resection planning in minimally invasive liver resection. *IEEE TMI*, 27(3):355–369, 2008. (Cited on page 3.)

- [99] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981. (Cited on page 33.)
- [100] J. M. Fitzpatrick. The role of registration in accurate surgical guidance. *J. Eng. Med.*, 224(5):607–622, 2010. (Cited on page 7.)
- [101] J. M. Fitzpatrick and J. B. West. The distribution of target registration error in rigid-body point-based registration. *IEEE TMI*, 20(9):917–927, 2001. (Cited on page 2.)
- [102] J. M. Fitzpatrick, J. B. West, and C. R. Maurer. Predicting error in rigid-body point-based registration. *IEEE TMI*, 17(5):694–702, 1998. (Cited on pages xxv, 2, 126, 127, 128, 129, and 157.)
- [103] A. Flint, A. Dick, and A. van den Hengel. Thrift: Local 3d structure recognition. In *Proc. DICTA*, 2007. (Cited on page 24.)
- [104] J. D. Foley, A. van Dam, S. K. Feiner, and J. H. Hughes. *Computer Graphics - Principles and Practice*. Addison-Wesley, Reading, Massachusetts, USA, 2 edition, 1990. (Cited on page 37.)
- [105] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *Proc. ECCV*, 2004. (Cited on pages 19, 21, and 24.)
- [106] T. Funkhouser and P. Shilane. Partial matching of 3d shapes with priority-driven search. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 131–142, 2006. (Cited on pages 21, 27, 33, and 39.)
- [107] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs. A search engine for 3d models. *ACM TOG*, 22(1):83–105, 2003. ISSN 0730-0301. doi: <http://doi.acm.org/10.1145/588272.588279>. (Cited on page 21.)
- [108] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. In *Proc. CVPR*, 2007. (Cited on page 10.)
- [109] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph.*, 25(1):130–150, 2006. (Cited on page 24.)

- [110] J. A. Gallian. A dynamic survey of graph labeling. *Elect. J. Combinatorics*, 16:1–219, 2009. (Cited on page 34.)
- [111] I. Garg, L. W. Clements, and R. L. Galloway. A computational approach to pre-align point cloud data for surface registration in image guided liver surgery. In *Proc. SPIE Med. Img.: Vis. Img.-guided Proc.*, 2007. (Cited on pages 3, 9, and 38.)
- [112] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *Proc. SIGGRAPH*, pages 209–216. ACM Press/Addison-Wesley Publishing Co., 1997. (Cited on page 41.)
- [113] T. Gatzke, C. Grimm, M. Garland, and S. Zelinka. Curvature maps for local shape comparison. In *Proc. IEEE Shape Mod. and Apps.*, pages 244–253, 2005. (Cited on pages 18 and 39.)
- [114] K. Gebal, J. A. Bærentzen, H. Aanæs, and R. Larsen. Shape analysis using the auto diffusion function. *Comp. Graph. Forum*, 28(5):1405–1413, 2009. (Cited on page 25.)
- [115] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann. Robust global registration. In *Proc. SGP*, pages 197–206, 2005. (Cited on pages 18, 20, 25, 27, 32, 34, 39, 125, 130, and 133.)
- [116] P. Glomb. Detection of interest points on 3d data: Extending the harris operator. In *Computer Recognition Systems 3*, volume 57 of *Advances in Intelligent and Soft Computing*, pages 103–111. Springer, 2009. (Cited on page 25.)
- [117] M. S. Gockenbach. *Understanding And Implementing the Finite Element Method*. Soc. for Industrial and App. Math., 2006. (Cited on page 13.)
- [118] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Trans. on Pat. Anal. and Mach. Intel.*, 18:377–388, 1996. ISSN 0162-8828. (Cited on page 90.)
- [119] M. W. Graham and W. E. Higgins. Globally optimal model-based matching of anatomical trees. In *Proc. SPIE Medical Imaging*, pages 373–387, 2006. (Cited on pages 98 and 113.)
- [120] M. W. Graham and W. E. Higgins. Optimal graph-theoretic approach to 3D anatomical tree matching. In *Proc. Biomedical Imaging*, pages 109–112, 2006. (Cited on pages 98 and 113.)

- [121] J. L. Gross and J. Yellen, editors. *Handbook of Graph Theory*. Discrete Mathematics and its Applications. CRC Press, 1 edition, 2003. (Cited on pages [42](#) and [52](#).)
- [122] L. J. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and the computation of voronoi diagrams. *ACM TOG*, 4(2):74–123, 1985. (Cited on pages [13](#), [41](#), and [56](#).)
- [123] E. Gursoz, Y. Choi, and F. Prinz. Vertex-based representation of non-manifold boundaries. In *Geometric Modeling for Product Engineering*, pages 107–130. Elsevier Science Publishers, 1990. (Cited on page [41](#).)
- [124] M. F. Hansen, M. R. Blas, and R. Larsen. Mahalanobis distance based iterative closest point. In *Proc. SPIE Med. Img.*, 2007. (Cited on page [36](#).)
- [125] R. Hartley and A. Zisserman. *Multiple View Geometry*. Cambridge Univ. Press, 2 edition, 2003. (Cited on pages [12](#) and [116](#).)
- [126] P. Heider, A. Pierre-Pierre, R. Li, and C. Grimm. Local shape descriptors, a survey and evaluation. In *Proc. 3DOR*, 2011. (Cited on page [22](#).)
- [127] T. Heimann and H.-P. Meinzer. Statistical shape models for 3d medical image segmentation: a review. *Med. Img. Anal.*, 13(4):543–563, 2009. (Cited on page [10](#).)
- [128] A. Herline, J. Herring, J. Stefansic, W. Chapman, R. Galloway, and B. Dawant. Surface registration for use in interactive image-guided liver surgery. In *Proc. MICCAI*, volume 1679, pages 892–899, 1999. (Cited on pages [3](#), [9](#), and [38](#).)
- [129] M. Higashi, H. Yatomi, Y. Mizutani, and S. Murabata. Unified geometric modeling by non-manifold shell operation. In *Proc. Solid modeling and applications*, pages 75–84, New York, NY, USA, 1993. ACM Press. (Cited on page [41](#).)
- [130] P. Hildebrand, M. Kleemann, U. J. Roblick, L. Mirow, C. Bürk, and H.-P. Bruch. Technical aspects and feasibility of laparoscopic ultrasound navigation in radiofrequency ablation of unresectable hepatic malignancies. *J. Laparoendosc. Adv. Surg. Tech. A.*, 17(1):53–57, 2007. (Cited on page [3](#).)
- [131] H. T. Ho and D. Gibbins. Curvature-based approach for multi-scale feature extraction from 3D meshes and unstructured point clouds. *IET Comput. Vis.*, 3(4):201–212, 2009. (Cited on pages [xviii](#), [25](#), and [125](#).)

- [132] D. S. Hochbaum and D. B. Shmoys. A best possible heuristic for the k-center problem. *Math. of Op. Research*, 10(2):180–184, 1985. (Cited on page 35.)
- [133] J.-S. Hong, T. Dohi, M. Hasizume, K. Konishi, and N. Hata. A motion adaptable needle placement instrument based on tumor specific ultrasonic image segmentation. In *Proc. MICCAI*, pages 122–129, 2002. (Cited on page 3.)
- [134] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am.*, 4:629–642, 1987. (Cited on pages 37, 103, and 136.)
- [135] A. Hostettler, S. Nicolau, C. Forest, L. Soler, and Y. Rémond. Real time simulation of organ motions induced by breathing: First evaluation on patient data. In *Proc. ISBMS*, pages 9–18, 2006. (Cited on page 13.)
- [136] C. Hughes, M. Glavin, E. Jones, and P. Denny. Review of geometric distortion compensation in fish-eye cameras. In *Proc. ISSC*, pages 162–167, 2008. (Cited on page 12.)
- [137] T. Igarashi, T. Moscovich, and J. F. Hughes. As-rigid-as-possible shape manipulation. *ACM TOG*, 24:1134–1141, July 2005. (Cited on page 35.)
- [138] H. Ishikawa. Higher-order clique reduction in binary graph cut. In *Proc. IEEE CVPR*, pages 2993–3000, 2009. (Cited on page 34.)
- [139] N. Iyer, S. Jayanti, K. Lou, Y. Kalyanaraman, and K. Ramani. Three-dimensional shape searching: state-of-the-art review and future trends. *Comput. Aided Des.*, 37:509–530, 2005. (Cited on page 22.)
- [140] B. Jian and B. C. Vemuri. Robust point set registration using gaussian mixture models. *IEEE TPAMI*, 33:1633–1645, 2011. (Cited on page 36.)
- [141] A. E. Johnson and M. Hebert. Surface matching for object recognition in complex 3-d scenes. *Img. and Vis. Comp.*, 16:635–651, 1998. (Cited on page 19.)
- [142] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE TPAMI*, 21(5): 433–449, 1999. (Cited on pages xviii, 19, and 20.)

- [143] T. R. Jones, F. Durand, and M. Desbrun. Non-iterative, feature-preserving mesh smoothing. In *SIGGRAPH*, pages 943–949, New York, NY, USA, 2003. ACM. ISBN 1-58113-709-5. doi: <http://doi.acm.org/10.1145/1201775.882367>. (Cited on page 106.)
- [144] J. Jost. *Riemannian Geometry and Geometric Analysis*. Springer, 5 edition, 2008. (Cited on page 17.)
- [145] T. Jost and H. Hügli. A multi-resolution ICP with heuristic closest point search for fast and robust 3D registration of range images. In *Proc. 3DIM*, volume 0, pages 427–433, 2003. (Cited on page 36.)
- [146] S. Kaneko, T. Kondo, and A. Miyamoto. Robust matching of 3D contours using iterative closest point algorithm improved by M-estimation. *Pa*, 36:2041–2047, 2003. (Cited on page 36.)
- [147] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Proc. SGP*, pages 156–164, 2003. (Cited on page 21.)
- [148] N. Kehtarnavaz and S. Mohan. A framework for estimation of motion parameters from range image. *Computer Vision, Graphics, and Image Processing*, 45(1):88–105, 1989. ISSN 0734-189X. doi: [http://dx.doi.org/10.1016/0734-189X\(89\)90072-8](http://dx.doi.org/10.1016/0734-189X(89)90072-8). (Cited on pages 18 and 83.)
- [149] L. Kettner. Using generic programming for designing a data structure for polyhedral surfaces. *Computational Geometry - Theory and Applications*, 13:65–90, 1999. (Cited on page 56.)
- [150] K. Khairy and J. Howard. Spherical harmonics-based parametric deconvolution of 3d surface images using bending energy minimization. *Med. Img. Anal.*, 12(2):217–227, 2008. (Cited on page 21.)
- [151] A. Khamene, J. K. Warzelhan, S. Vogt, D. Elgort, C. Chedf’Hotel, J. L. Duerk, J. S. Lewin, F. K. Wacker, and F. Sauer. Characterization of internal organ motion using skin marker positions. In *Proc. MICCAI*, pages 526–533, 2004. (Cited on page 3.)
- [152] M. F. Khan, S. Dogan, A. Maataoui, S. Wesarg, J. Gurung, H. Ackermann, M. Schiemann, G. Wimmer-Greinecker, and T. J. Vogl. Navigation-based needle puncture of a cadaver using a hybrid tracking navigational system. *Invest. Radiol.*, 41(10):713–720, 2006. (Cited on page 2.)

- [153] A. Khodakovsky, P. Alliez, M. Desbrun, and P. Schröder. Near-optimal connectivity encoding of 2-manifold polygon meshes. *Graph. Models*, 64:147–168, 2002. (Cited on page 12.)
- [154] T. Kilgus, T. R. dos Santos, A. Seitel, K. Yung, A. M. Franz, A. Groch, I. Wolf, H.-P. Meinzer, and L. Maier-Hein. Generation of triangle meshes from time-of-flight data for surface registration. In *Proc. BVM*, pages 189–193, 2011. (Cited on page 12.)
- [155] J. J. Koenderink and A. J. van Doorn. Surface shape and curvature scales. *Img. & Vis. Comp.*, 10(8):557–565, 1992. ISSN 0262-8856. doi: [http://dx.doi.org/10.1016/0262-8856\(92\)90076-F](http://dx.doi.org/10.1016/0262-8856(92)90076-F). (Cited on pages xxii, 18, 74, 86, 88, 111, 125, and 143.)
- [156] A. Kolb, E. Barth, R. Koch, and R. Larsen. Time-of-flight sensors in computer graphics. In *Proc. Eurographics - STAR*, pages 119–134, 2009. (Cited on pages 10 and 75.)
- [157] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts—a review. *IEEE TPAMI*, 29:1274–1279, 2007. (Cited on page 34.)
- [158] V. Kraevoy and A. Sheffer. Cross-parameterization and compatible remeshing of 3d models. In *Proc. ACM SIGGRAPH*, pages 861–869, 2004. (Cited on page 85.)
- [159] V. Kraevoy and A. Sheffer. Mean-value geometry encoding. *Int. J. Shape Mod.*, 12(1):29–46, 2006. (Cited on pages 29 and 35.)
- [160] J. Krücker, S. Xu, N. Glossop, A. Viswanathan, J. Borgert, H. Schulz, and B. J. Wood. Electromagnetic tracking for thermal ablation and biopsy guidance: Clinical evaluation of spatial accuracy. *J. Vasc. Interv. Radiol.*, 18(9):1141–50, 2007. (Cited on page 2.)
- [161] J. Krücker, S. Xu, N. Glossop, W. F. Pritchard, J. Karanian, A. Chiesa, and B. J. Wood. Evaluation of motion compensation approaches for soft tissue navigation. In *Proc. SPIE Med. Img.: Vis. Img.-Guided Proc. Mod.*, 2008. (Cited on page 2.)
- [162] R. Ladner, K. Shaw, and M. Abdelguerfi. 3d synthetic environment representation using the "non-manifold 3d winged-edge" data structure. In *Proc. Interoperating Geographic Information Systems*, pages 305–314, London, UK, 1999. Springer-Verlag. ISBN 3-540-65725-8. (Cited on page 41.)

- [163] T. Lange, N. Papenberg, S. Heldmann, J. Modersitzki, B. Fischer, H. Lamecker, and P. Schlag. 3d ultrasound-ct registration of the liver using combined landmark-intensity information. *Int. J. CARS*, 4:79–88, 2009. ISSN 1861-6410. (Cited on page 3.)
- [164] D. Lanman and G. Taubin. Build your own 3d scanner: 3d photography for beginners. In *ACM SIGGRAPH courses*, pages 1–87, 2009. (Cited on page 10.)
- [165] E. L. Lawler and D. E. Wood. Branch-and-bound methods: A survey. *Operations Research*, 14:699–719, 1966. (Cited on page 32.)
- [166] C. H. Lee, A. Varshney, and D. W. Jacobs. Mesh saliency. *ACM TOG*, 24:659–666, 2005. (Cited on page 24.)
- [167] J.-D. Lee, C.-H. Huang, T.-C. Huang, H.-Y. Hsieh, and S.-T. Lee. Medical augment reality using a markerless registration framework. *Exp. Sys. Apps.*, 2011. ISSN 0957-4174. In press. (Cited on page 9.)
- [168] Y. Lee, S. Lee, A. Shamir, D. Cohen-Or, and H.-P. Seidel. Mesh scissoring with minima rule and part salience. *Comp. Aid. Geom. Design*, 22(5):444–465, 2005. (Cited on pages xxii and 84.)
- [169] E. B. Levy, J. Tang, D. Lindisch, N. Glossop, F. Banovac, and K. Cleary. Implementation of an electromagnetic tracking system for accurate intrahepatic puncture needle guidance: Accuracy results in an in vitro model. *Acad. Radiol.*, 14(3):344–354, 2007. (Cited on page 2.)
- [170] E. B. Levy, H. Zhang, D. Lindisch, B. J. Wood, and K. Cleary. Electromagnetic tracking-guided percutaneous intrahepatic portosystemic shunt creation in a swine model. *J. Vasc. Interv. Radiol.*, 18(2):303–307, 2007. (Cited on page 2.)
- [171] X. J. Li and I. Guskov. 3d object recognition from range images using pyramid matching. In *Proc. ICCV*, pages 1–6, 2007. (Cited on page 12.)
- [172] P. Lienhardt. Topological models for boundary representation: a comparison with n-dimensional generalized maps. *CAD*, 23(1):59–82, 1991. (Cited on page 41.)
- [173] Y. Lipman and T. Funkhouser. Möbius voting for surface correspondence. *ACM TOG*, 28(3):1–12, 2009. (Cited on pages 28, 30, 33, 34, and 39.)

- [174] Y. Lipman, O. Sorkine, D. Levin, and D. Cohen-Or. Linear rotation-invariant coordinates for meshes. In *Proc. SIGGRAPH*, pages 479–487, 2005. (Cited on pages 29 and 35.)
- [175] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proc. Computer graphics and interactive techniques*, pages 163–169. ACM Press, 1987. (Cited on page 42.)
- [176] H.-T. Lübbers, F. Matthews, W. Zemann, K. W. Grätz, J. A. Obwegeser, and M. Bredell. Registration for computer-navigated surgery in edentulous patients: A problem-based decision concept. *J. Cranio-Maxillofac. Surg.*, 39(6):453 – 458, 2011. (Cited on page 9.)
- [177] L. Maier-Hein. *Computer-assisted needle insertion - Motion compensation and guidance*. Südwestdeutscher Verlag für Hochschulschriften AG Co. KG, 2009. (Cited on page 3.)
- [178] L. Maier-Hein, F. Pianka, S. Müller, U. Rietdorf, A. Seitel, A. Franz, I. Wolf, B. Schmied, and H.-P. Meinzer. Respiratory liver motion simulator for validating image-guided systems ex-vivo. *Int. J. CARS*, 2:287–292, 2008. (Cited on pages xxv, 134, and 135.)
- [179] L. Maier-Hein, A. Tekbas, A. M. Franz, R. Tetzlaff, S. A. Müller, F. Pianka, I. Wolf, H.-U. Kauczor, B. M. Schmied, and H.-P. Meinzer. On combining internal and external fiducials for liver motion compensation. *Comp. Aid. Surg.*, 13(16):369–376, 2008. (Cited on page 3.)
- [180] L. Maier-Hein, A. Tekbas, A. Seitel, F. Pianka, S. A. Müller, S. Satz, S. Schawo, B. Radeleff, R. Tetzlaff, A. M. Franz, B. P. Müller-Stich, I. Wolf, H.-U. Kauczor, B. M. Schmied, and H.-P. Meinzer. In vivo accuracy assessment of a needle-based navigation system for CT-guided radiofrequency ablation of the liver. *Med. Phys.*, 35(12):5385–5396, 2008. (Cited on page 2.)
- [181] L. Maier-Hein, M. Schmidt, A. M. Franz, T. R. dos Santos, A. Seitel, B. Jähne, J. M. Fitzpatrick, and H. P. Meinzer. Accounting for anisotropic noise in fine registration of time-of-flight range data with high-resolution surface data. In *Proc. MICCAI*, pages 251–258, 2010. (Cited on pages 73 and 132.)
- [182] L. Maier-Hein, T. R. dos Santos, A. M. Franz, and H. P. Meinzer. Iterative closest point algorithm with anisotropic weighting and

its application to fine surface registration. In *Proc. SPIE Med. Img.: Img. Proc.*, 2011. (Cited on page 73.)

- [183] M. Mäntylä. *Introduction to Solid Modeling*. Computer Science Press, Rockville, Maryland, USA, 1988. (Cited on pages 13, 41, 42, and 56.)
- [184] M. Markert, A. Koschany, and T. Lueth. Tracking of the liver for navigation in open surgery. *Int. J. CARS*, 5(3):229–235, 2010. (Cited on page 3.)
- [185] R. Marmulla, T. Lüth, J. Mühling, and S. Hassfeld. Automated laser registration in image-guided surgery: evaluation of the correlation between laser scan resolution and navigation accuracy. *Int. J. Oral Maxillofac. Surg.*, 33:642–648, 2004. (Cited on page 9.)
- [186] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley-Interscience, 1996. (Cited on page 35.)
- [187] F. Mémoli. On the use of gromov-hausdorff distances for shape comparison. In *Proc. EG Symp. on Pto.-based Graph.*, 2007. (Cited on page 30.)
- [188] J. H. Metzen, T. Kröger, A. Schenk, S. Zidowitz, H.-O. Peitgen, and X. Jiang. Matching of anatomical tree structures for registration of medical images. *Image and Vision Computing*, 27:923–933, 2009. (Cited on pages 98 and 113.)
- [189] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Vis. & Math. III*, pages 35–57. Springer-Verlag, 2003. (Cited on pages xvii, 17, 19, 74, 86, 122, 127, and 140.)
- [190] M. J. Mohlenkamp. A fast transform for spherical harmonics. *J Fourier Anal. and Apps.*, 5(2/3):159–184, 1999. (Cited on page 20.)
- [191] G. Mori, S. Belongie, and J. Malik. Shape contexts enable efficient retrieval of similar shapes. In *Proc. IEEE CVPR*, 2001. (Cited on page 24.)
- [192] D. Münch, B. Combès, and S. Prima. A modified ICP algorithm for normal-guided surface registration. In *SPIE Med. Img.: Vis., Img-Guided Proc.*, volume 7623, 2010. (Cited on page 36.)
- [193] J. Mundy. Object recognition in the geometric era: A retrospective. In *Toward Category-Level Object Recognition*, volume

4170, pages 3–28. Springer Berlin / Heidelberg, 2006. (Cited on page 11.)

- [194] S. A. Nicolau, X. Pennec, L. Soler, and N. Ayache. Clinical evaluation of a respiratory gated guidance system for liver punctures. In *Proc. MICCAI*, pages 77–85, 2007. (Cited on page 2.)
- [195] S. A. Nicolau, X. Pennec, L. Soler, X. Buy, A. Gangi, N. Ayache, and J. Marescaux. An augmented reality system for liver thermal ablation: design and evaluation on clinical cases. *MIA*, 13(3):494–506, 2009. (Cited on page 2.)
- [196] J. Novatnack and K. Nishino. Scale-dependent 3d geometric features. In *Proc. IEEE ICCV*, 2007. (Cited on page 25.)
- [197] P. Oberle. 3d laser scanner: Laser surveying, 3d imaging, 3d scanning, lidar, Sep. 2010. URL <http://knol.google.com/k/paul-oberle/3d-laser-scanner/13av68kkt1k9q/2>. (Cited on page 10.)
- [198] J. Olesch, B. Beuthien, S. Heldmann, N. Papenberg, and B. Fischer. Fast intra-operative nonlinear registration of 3D-CT to tracked, selected 2D-ultrasound slices. In *Proc. SPIE Med. Img.: Vis. Img.-guided Proc. Mod.*, 2011. (Cited on page 3.)
- [199] R. E. Ong, S. D. Herrell III, M. I. Miga, and R. L. Galloway. A kidney deformation model for use in non-rigid registration during image-guided surgery. In *Proc. SPIE Med. Img. Vis., Img-Guided Proc., Mod.*, 2008. (Cited on page 13.)
- [200] L. Oswald. Recent development of the iterative closest point algorithm. Technical report, Auton. Sys. Lab., Swiss Federal Institute of Technology, 2010. (Cited on page 36.)
- [201] C. Ozhasoglu, C. B. Saw, H. Chen, S. Burton, K. Komanduri, N. J. Yue, S. M. Huq, and D. E. Heron. Synchrony - Cyberknife respiratory compensation technology. *Med. Dosim.*, 33(2):117–123, 2008. (Cited on page 3.)
- [202] C. Papazov and D. Burschka. Deformable 3d shape registration based on local similarity transforms. *Comp. Graph. Forum*, 30(5):1493–1502, 2011. (Cited on pages 17 and 35.)
- [203] P. M. Pardalos, F. Rendl, and H. Wolkowicz. The quadratic assignment problem: A survey and recent developments. In *Proc. DIMACS Work. on QAP*, pages 1–42, 1994. (Cited on page 32.)

- [204] M. Pauly, R. Keiser, and M. Gross. Multi-scale feature extraction on point-sampled surface. In *Proc. Eurographics*, 2003. (Cited on page 25.)
- [205] G. P. Penney, P. J. Edwards, A. P. King, J. M. Blackall, P. G. Batchelor, and D. J. Hawkes. A stochastic iterative closest point algorithm (stochastICP). In *Proc. MICCAI*, pages 762–769, 2001. (Cited on page 36.)
- [206] T. M. Peters. Image-guidance for surgical procedures. *Phys. Med. Biol.*, 51(14):R505–R540, 2006. (Cited on page 2.)
- [207] A. Petrelli and L. Di Stefano. On the repeatability of the local reference frame for partial shape matching. In *Proc. ICCV*, 2011. (Cited on pages 20, 122, 140, and 141.)
- [208] D. L. Pham, C. Xu, , and J. L. Prince. Current methods in medical image segmentation. *Annual Review of Biomedical Engineering*, 2:315–337, 2000. (Cited on page 10.)
- [209] U. Pinkall, S. D. Juni, and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2: 15–36, 1993. (Cited on page 34.)
- [210] K. Polthier. Computational aspects of discrete minimal surfaces. In *Proc. of the Clay Math. Inst. Summer School*, 2005. (Cited on page 34.)
- [211] H. Pottmann, J. Wallner, Q.-X. Huang, and Y.-L. Yang. Integral invariants for robust geometry processing. *Comput. Aided Geom. Des.*, 26(1):37–60, 2009. (Cited on page 20.)
- [212] T. P. Rauth, P. Q. Bao, R. L. Galloway, J. Bieszczad, E. M. Friets, D. A. Knaus, D. B. Kynor, and A. J. Herline. Laparoscopic surface scanning and subsurface targeting: Implications for image-guided laparoscopic liver surgery. *Surgery*, 142(2):207 – 214, 2007. (Cited on pages 3, 9, and 38.)
- [213] D. Raviv, A. Dubrovina, and R. Kimmel. Hierarchical matching for non-rigid shapes. In *Proc. SSVM*, 2011. (Cited on pages 28, 32, 35, 39, 118, 127, and 144.)
- [214] M. Reuter, F.-E. Wolter, and N. Peinecke. Laplace-beltrami spectra as "shape-dna" of surfaces and solids. *Comp. Aid. Design*, 38 (4):342–366, 2006. (Cited on page 22.)

- [215] M. Reuter, S. Biasotti, D. Giorgi, G. Patanè, and M. Spagnuolo. Discrete laplace-beltrami operators for shape analysis and segmentation. *Comp. & Grap.*, 33(3):381–390, 2009. (Cited on page 21.)
- [216] J. Rossignac and D. Cardoze. Matchmaker: Manifold breps for non-manifold r-sets. In *Proc. Solid Modeling and Applications*, pages 31–34, 1999. (Cited on page 41.)
- [217] J. Rossignac and M. O’Connor. Sgc: A dimension-idependent model for pointsets with internal structures and incomplete boundaries. In *Geometric Modeling for Product Engineering*, pages 145–180. Elsevier Science Publishers, 1990. (Cited on page 41.)
- [218] G. Rote. Computing the minimum hausdorff distance between two point sets on a line under translation. *Inf. Process. Lett.*, 38: 123–127, 1991. (Cited on page 30.)
- [219] P. M. Roth and M. Winter. Survey of appearance-based methods for object recognition. Technical report, Inst. for Computer Graphics and Vision, Graz University of Technology, Austria, ICG-TR-01/08, 2008. (Cited on page 11.)
- [220] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *Proc. 3D Dig. Img. and Mod.*, 2001. (Cited on page 36.)
- [221] R. M. Rustamov. Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In *Proc. SGP*, pages 225–233, 2007. (Cited on page 22.)
- [222] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *Proc. IEEE IROS*, 2010. (Cited on page 12.)
- [223] Y. Sahillioğlu and Y. Yemez. Coarse-to-fine combinatorial matching for dense isometric shape correspondence. *Computer Graphics Forum*, 30(5):1461–1470, 2011. (Cited on pages 17, 28, 33, 35, 38, and 39.)
- [224] S. Sahni and T. Gonzalez. P-complete approximation problems. *J. ACM*, 23:555–565, 1976. (Cited on pages 27 and 32.)
- [225] J. Salvi, J. Pagès, and J. Battle. Pattern codification strategies in structured light systems. *Patt. Rec.*, 37:827–849, 2004. (Cited on page 10.)

- [226] J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe. Inter-surface mapping. In *Proc. ACM SIGGRAPH*, pages 870–877, 2004. (Cited on page 85.)
- [227] A. Schweikard, G. Glosser, M. Bodduluri, M. J. Murphy, and J. R. Adler. Robotic motion compensation for respiratory movement during radiosurgery. *Comp. Aid. Surg.*, 5:263–277, 2000. (Cited on page 3.)
- [228] A. Seitel, T. R. dos Santos, S. Mersmann, J. Penne, A. Groch, K. Yung, R. Tetzlaff, H.-P. Meinzer, and L. Maier-Hein. Adaptive bilateral filter for image denoising and its application to in-vitro time-of-flight data. In *Proc. SPIE Med. Img.: Vis. Img.-guided Proc. Disp.*, 2011. (Cited on page 75.)
- [229] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. CVPR*, 2006. (Cited on page 10.)
- [230] A. Selinger. Analysis and applications of feature-based object recognition. Technical report, Univ. of Rochester. Comp. Sc. Dep., 2001. UR CSD / TR755. (Cited on page 11.)
- [231] A. Shamir. A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27(6):1539–1556, 2008. (Cited on page 84.)
- [232] Y. Shan, B. Matei, H. S. Sawhney, R. Kumar, D. Huber, and M. Hebert. Linear model hashing and batch ransac for rapid and accurate object recognition. In *Proc. IEEE CVPR*, pages 121–128, 2004. (Cited on page 25.)
- [233] A. Sharma, R. Horaud, J. Cech, and E. Boyer. Topologically-robust 3d shape matching based on diffusion geometry and seed growing. In *Proc. CVPR*, 2011. (Cited on pages 35, 39, and 144.)
- [234] J. Shlens. A tutorial on principal component analysis. Technical report, Institute for Nonlinear Science, UCSD, 2005. (Cited on page 17.)
- [235] F. G. M. Silva and A. J. P. Gomes. Oversimplified euler operators for a non-oriented, non-manifold b-rep data structure. In *Proc. First International Symposium on Advances in Visual Computing (ISVC 2005)*, pages 25–34, 2005. (Cited on page 41.)
- [236] S. Silva, J. Madeira, and B. S. Santos. POLYMECO - a polygonal mesh comparison tool. In *Proc. IEEE IV*, pages 842–847, 2005. (Cited on page 71.)

- [237] D. Sindram, I. H. McKillop, J. B. Martinie, and D. A. Iannitti. Novel 3-D laparoscopic magnetic ultrasound image guidance for lesion targeting. *HPB*, 12(10):709–716, 2010. (Cited on page 3.)
- [238] I. Sipiran and B. Bustos. A robust 3d interest points detector based on harris operator. In *Proc. 3DOR*, pages 7–14, 2010. (Cited on page 25.)
- [239] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Proc. SGP*, pages 109–116, 2007. (Cited on page 35.)
- [240] V. Srinivasan and E. Akleman. Connected & manifold Sierpinsky polyhedra. In *Proc. ACM Symp. on Solid Modeling and App.*, pages 261–266, 2004. (Cited on page 43.)
- [241] V. Srinivasan, E. Akleman, and J. Chen. Interactive construction of multi-segment curved handles. In *Proc. Pacific Graphics 2002*, 2002. (Cited on page 42.)
- [242] V. Srinivasan, E. Akleman, and J. Keyser. Topological construction of 2-manifold meshes from arbitrary polygonal data. Technical report, Texas A&M University, Tamu College Station, Texas, USA, 2004. (Cited on pages 45 and 48.)
- [243] J. Stam. Evaluation of loop subdivision surfaces. In *Proc. SIGGRAPH*, 1998. (Cited on page 73.)
- [244] J. Stam and C. Loop. Quad/triangle subdivision. *Computer Graphics Forum*, 22(1):79–86, 2003. (Cited on page 50.)
- [245] A. J. Stoddart, M. Petrou, and J. Kittler. Probabilistic relaxation as an optimizer. In *Proc. BMVC*, pages 613–622, 1995. (Cited on page 32.)
- [246] N. Sugano. Computer-assisted orthopedic surgery. *J. Orthop. Sci.*, 8(3):442–448, 2003. (Cited on page 2.)
- [247] R. W. Sumner and J. Popović. Deformation transfer for triangle meshes. *ACM TOG*, 23:399–405, 2004. (Cited on page 35.)
- [248] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Proc. SGP*, 2009. (Cited on pages 22 and 25.)
- [249] Y. Sun and M. A. Abidi. Surface matching by 3d point’s fingerprint. In *Proc. ICCV*, volume 2, pages 263–269, 2001. (Cited on page 19.)

- [250] Y. Sun, P. J., A. Koschan, D. L. Page, and M. A. Abidi. Point fingerprint: A new 3-d object representation scheme. *IEEE Trans on Sys., Man, and Cyb.*, 33(4):712–717, 2003. (Cited on page 19.)
- [251] A. Tagliasacchi, H. Zhang, and D. Cohen-Or. Curve skeleton extraction from incomplete point cloud. *ACM TOG*, 28:71:1–71:9, 2009. (Cited on page 20.)
- [252] J. Tang, G.-S. Wu, F.-Y. Zhang, and M.-M. Zhang. Fast approximate geodesic paths on triangle mesh. *Int. J. Aut. & Comp.*, 4(1): 8–13, 2007. (Cited on page 131.)
- [253] J. Tangelder and R. Veltkamp. A survey of content based 3d shape retrieval methods. In *Proc. IEEE SMI*, pages 145–156, 2004. (Cited on page 22.)
- [254] G. Taubin. A signal processing approach to fair surface design. In *Proc. ACM Comp. Graph. & Interact. Tech.*, pages 351–358, 1995. (Cited on page 21.)
- [255] A. Tevs, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. Isometric registration of ambiguous and partial data. In *Proc. CVPR*, pages 1185–1192, 2009. (Cited on pages 33, 35, 38, and 39.)
- [256] R. Toldo, U. Castellani, and A. Fusiello. Visual vocabulary signature for 3D object retrieval and partial matching. In *Proc. 3DOR*, 2009. (Cited on page 20.)
- [257] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *Proc. ECCV*, pages 347–360, 2010. (Cited on pages xxiv, xxv, 19, 121, 132, 133, 134, and 140.)
- [258] F. Tombari, S. Salti, and L. Di Stefano. A combined texture-shape descriptor for enhanced 3d feature matching. In *Proc. IEEE ICIP*, 2011. (Cited on page 19.)
- [259] L. Torresani, V. Kolmogorov, and C. Rother. Feature correspondence via graph matching: Models and global optimization. In *Proc. ECCV*, pages 596–609, 2008. (Cited on pages 34 and 83.)
- [260] B. Vallet and B. Lévy. Spectral geometry processing with manifold harmonics. In *Proc. EG*, pages 251–260, 2008. (Cited on page 21.)

- [261] O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. A survey on shape correspondence. In *Proc. EG STAR*, 2010. (Cited on page 31.)
- [262] V. Vidal, C. Wolf, C. Dupont, and G. Lavoué. An iterative approach for global triangular mesh regularization. Technical Report RR-LIRIS-2009-032, LIRIS, University of Lyon, 2009. (Cited on page 72.)
- [263] C. Wang, M. M. Bronstein, A. M. Bronstein, and N. Paragios. Discrete minimum distortion correspondence problems for non-rigid shape matching. In *Proc. SSVM*, 2010. (Cited on pages 28, 34, 35, 39, 118, and 127.)
- [264] Y. Wang. Measuring similarity between curves on 2-manifolds via minimum deformation area. Technical report, Dept. of Comp. Sc. & Eng., Ohio State Univ., 2008. (Cited on page 153.)
- [265] C. Weber, S. Hahmann, and H. Hagen. Sharp feature detection in point clouds. In *Proc. IEEE SMI*, pages 175–186, 2010. (Cited on page 25.)
- [266] S. Weber, M. Markert, and T. Lüth. Surface tracking of organs for registration in soft tissue surgery. In *Proc. MICCAI IGSTI*, 2008. (Cited on page 3.)
- [267] Z. Wei, L. Gardi, D. B. Downey, and A. Fenster. Oblique needle segmentation and tracking for 3D TRUS guided prostate brachytherapy. *Med. Phys.*, 32(9):2928–2941, 2005. (Cited on page 3.)
- [268] K. Weiler. The radial edge data structure: A topological representation for non-manifold geometric boundary modeling. In *Geometric Modeling for CAD Applications*. Elsevier Science Publishers, 1988. (Cited on page 41.)
- [269] W. Wein, A. Khamene, D.-A. Clevert, O. Kutter, and N. Navab. Simulation and fully automatic multimodal registration of medical ultrasound. In *Proc. MICCAI*, pages 136–143, 2007. (Cited on page 3.)
- [270] T. Werner. Revisiting the linear programming relaxation approach to gibbs energy minimization and weighted constraint satisfaction. *IEEE TPAMI*, 32:1474–1488, 2010. (Cited on page 34.)

- [271] T. Windheuser, U. Schlickewei, F. R. Schmidt, and D. Cremers. Geometrically consistent elastic matching of 3d shapes: A linear programming solution. In *Proc. ICCV*, 2011. (Cited on pages 18, 29, 34, and 39.)
- [272] O. Wirjadi. Survey of 3D image segmentation methods. Technical Report 123, Fraunhofer-Institut für Techno- und Wirtschaftsmathematik, 2007. (Cited on page 10.)
- [273] B. J. Wood, J. Kruecker, N. Abi-Jaoudeh, J. K. Locklin, E. Levy, S. Xu, L. Solbiati, A. Kapoor, H. Amalou, and A. M. Venkatesan. Navigation systems for ablation. *J. Vasc. Interv. Radiol.*, 21(8): S257–S263, 2010. (Cited on page 2.)
- [274] H.-Y. Wu, H. Zha, T. Luo, X.-L. Wang, and S. Ma. Global and local isometry-invariant descriptor for 3d shape comparison and partial matching. In *Proc. CVPR*, pages 438–445, 2010. (Cited on page 22.)
- [275] S. M. Yamany and A. A. Farag. Surfacing signatures: An orientation independent free-form surface representation scheme for the purpose of objects registration and matching. *IEEE TP*, 24: 1105–1120, 2002. (Cited on page 24.)
- [276] M.-H. Yang. *Object Recognition*, pages 1936–1939. 2009. (Cited on page 11.)
- [277] Z. Yaniv and K. Cleary. Image-guided procedures: A review. Technical report, Georgetown University, 2006. (Cited on page 2.)
- [278] L. A. Zager and G. C. Verghese. Graph similarity scoring and matching. *App. Math. Let.*, 21:86–94, 2008. (Cited on pages 89 and 90.)
- [279] A. Zaharescu, E. Boyer, K. Varanasi, and R. P. Horaud. Surface feature detection and description with applications to mesh matching. In *Proc. CVPR*, pages 373–380, 2009. (Cited on pages 19 and 24.)
- [280] R. Zass and A. Shashua. Probabilistic graph and hypergraph matching. In *Proc. CVPR*, 2008. (Cited on page 83.)
- [281] Y. Zeng, C. Wang, Y. Wang, X. Gu, D. Samaras, and N. Paragios. Dense non-rigid surface registration using high-order graph matching. In *Proc. CVPR*, pages 382–389, 2010. (Cited on pages xix, 17, 18, 28, 29, 30, 34, 35, 36, 38, 39, 83, 131, and 144.)

- [282] Y. Zeng, C. Wang, Y. Wang, X. Gu, D. Samaras, and N. Paragios. A generic local deformation model for shape registration. Technical Report RR-7676, INRIA, 2011. (Cited on pages [18](#), [28](#), [29](#), [30](#), and [34](#).)
- [283] C. Zhang and Z. Zhang. Calibration between depth and color sensors for commodity depth cameras. In *Proc. IEEE ICME*, pages 1–6, 2011. (Cited on page [12](#).)
- [284] E. Zhang, K. Mischikow, and G. Turk. Feature-based surface parameterization and texture mapping. *ACM TOG*, 24(1):1–27, 2005. (Cited on page [85](#).)
- [285] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. van Kaick, and A. Tagliasacchi. Deformation-driven shape correspondence. In *Proc. SGP*, pages 1431–1439, 2008. (Cited on pages [xviii](#), [17](#), [24](#), [29](#), [33](#), [38](#), [39](#), [119](#), and [130](#).)
- [286] H. Zhang, O. van Kaick, and R. Dyer. Spectral mesh processing. *Comp. Graph. Forum*, 29(6):1865–1894, 2010. (Cited on page [21](#).)
- [287] S. Zhang, J. Huang, and D. Metaxas. Robust mesh editing using Laplacian coordinates. *Graph. Models*, 73:10–19, 2011. (Cited on page [35](#).)
- [288] X. Zhang, J. Liu, Z. Li, and M. Jaeger. Volume decomposition and hierarchical skeletonization. In *Proceedings ACM SIG-GRAPH VRCAI*, pages 17:1–17:6, 2008. (Cited on page [20](#).)
- [289] K. Zhou, J. Snyder, B. Guo, and H.-Y. Shum. Iso-charts: Stretch-driven mesh parameterization using spectral analysis. In *Proc. SGP*, pages 45–54, 2004. (Cited on page [85](#).)
- [290] D.-X. Zhuang, Y.-X. Liu, J.-S. Wu, C.-J. Yao, Y. Mao, C.-X. Zhang, M.-N. Wang, W. Wang, and L.-F. Zhou. A sparse intraoperative data-driven biomechanical model to compensate for brain shift during neuronavigation. *Americ. J. Neurorad.*, 32:395–402, 2010. (Cited on page [13](#).)
- [291] V. Zobel, J. Reininghaus, and I. Hotz. Generalized heat kernel signatures. *J. WSCG*, 19(3):93–100, 2011. (Cited on page [22](#).)