UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Luciano Walenty Xavier Cejnog

Rigid registration based on local geometric dissimilarity

Juiz de Fora

2015

UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Luciano Walenty Xavier Cejnog

Rigid registration based on local geometric dissimilarity

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientador: Marcelo Bernardes Vieira

Juiz de Fora

2015

Luciano Walenty Xavier Cejnog

Rigid registration based on local geometric dissimilarity

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Aprovada em 21 de Setembro de 2015.

BANCA EXAMINADORA

Prof. D.Sc. Marcelo Bernardes Vieira - Orientador
Universidade Federal de Juiz de Fora

Prof. D.Sc. Rodrigo Luis de Souza da Silva
Universidade Federal de Juiz de Fora

Prof. D.Sc. Gilson Antônio Giraldi
Laboratório Nacional de Computação Científica

ACKNOWLEDGMENTS

First, I'd like to thank the other members of this project: my friend Fernando and my advisor Marcelo. Without their contribution and patience, this idea wouldn't have gotten that far.

To all my other colleagues on the GCG and on the University, for all the support, the encouraging words and ideas.

To the professors of the UFJF, for teaching me the right way to make research, and for all the lessons.

To my mother, my father, my little brothers and my girlfriend, for the unconditional support.

I also thank FAPEMIG and CAPES for the financial support.

"Research is what I'm doing when I don't know what I'm doing." Wernhen von Braun

RESUMO

Este trabalho visa melhorar um método clássico para o problema de registro rígido, o ICP (iterative Closest Point), fazendo com que a busca dos pontos mais próximos, uma de suas fases principais, considere informações aproximadas da geometria local de cada ponto combinadas à distância Euclidiana originalmente usada. Para isso é necessária uma etapa de pré-processamento, na qual a geometria local é estimada em tensores de orientação de segunda ordem. É definido o CTSF, um fator de similaridade entre tensores. O ICP é alterado de modo a considerar uma combinação linear do CTSF com a distância Euclidiana para estabelecer correspondências entre duas nuvens de pontos, variando os pesos relativos entre os dois fatores. Isso proporciona uma capacidade maior de convergência para ângulos maiores em relação ao ICP original, tornando o método comparável aos que constituem o estado da arte da área.

Para comprovar o ganho obtido, foram realizados testes exaustivos em malhas com características geométricas variadas, para diferentes níveis de ruído aditivo, outliers e em casos de sobreposição parcial, variando os parâmetros do método de estimativa dos tensores. Foi definida uma nova base com malhas sintéticas para os experimentos, bem como um protocolo estatístico de avaliação quantitativa. Nos resultados, a avaliação foi feita de modo a determinar bons valores de parâmetros para malhas com diferentes características, e de que modo os parâmetros afetam a qualidade do método em situações com ruído aditivo, outliers, e sobreposição parcial.

Palavras-chave: Registro rígido. Iterative Closest Point. Tensor de orientação. Dissimilaridade de forma.

ABSTRACT

This work aims to enhance a classic method for the rigid registration problem, the ICP (Iterative Closest Point), modifying one of its main steps, the closest point search, in order to consider approximated information of local geometry combined to the Euclidean distance, originally used. For this, a preprocessing stage is applied, in which the local geometry is estimated in second-order orientation tensors. We define the CTSF, a similarity factor between tensors. Our method uses a linear combination between this factor and the Euclidean distance, in order to establish correspondences, and a strategy of weight variation between both factors. This increases the convergence probability for higher angles with respect to the original ICP, making our method comparable to some of the state-of-art techniques.

In order to comprove the enhancement, exhaustive tests were made in point clouds with different geometric features, with variable levels of additive noise and outliers and in partial overlapping situations, varying also the parameters of the tensor estimative method. A dataset of synthetic point clouds was defined for the experiments, as well as a statistic protocol for quantitative evaluation. The results were analyzed in order to highlight good parameter ranges for different point clouds, and how these parameters affect the behavior of the method in situations of additive noise, outliers and partial overlapping.

Keywords: Rigid registration. Iterative Closest Point. Orientation Tensor. Shape Dissimilarity.

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\overrightarrow{r}$	Vector $\overrightarrow{r} = (x, y, z)$, with $x, y, z \in \mathbb{R}$.
$\|\|\overrightarrow{r}\|\|$	Norm L_2 of a vector $\overrightarrow{r}$.
$\widehat{r}$	Normalized vector $\widehat{r} = (x, y, z)$, with $x, y, z \in \mathbb{R}$ and $\|\|\widehat{r}\|\| = x^2 + y^2 + z^2 = 1$.
$\mathbf{S}_p$	Orientation second-order tensor.
$L_k(p)$	List of neighbors of a point p.
M	Point cloud.
p	A point component of a point cloud.
$L_k^-1(p)$	List of points that contains the point p as a neighbor.

CONTENTS

1 INTRODUCTION

Surface registration is the alignment of 3D surfaces into a common coordinate system. It is an important process on computer geometry applications, inserted into the processing pipeline, such as object recognition on 3D scenes, geometry acquisition for computer vision and medical image applications. Since it is a primary step, alignment errors affect directly the output of those pipelines, and therefore solve appropriately the registration problem is fundamental.

Rigid registration is a constrained version of this problem, which aims to find a rotation and a translation that best align two surfaces. Since the transformation is constrained to 6 degrees of freedom, the problem has a straightforward mathematical definition.

The first successful solution for rigid registration between two point sets was Besl and McKay's Iterative Closest Point (ICP) (BESL; MCKAY, 1992). This method is based on the iterative establishment of correspondences between points on both sets and the estimation of a transformation that minimizes the Root Mean Square (RMS) Euclidean distance between the correspondences.

The original ICP method is able to retrieve optimal alignment when both sets represent completely the same object and there is a small rotation and translation between them. However, in practice, point sets are mostly partially overlapped and subject to the presence of outliers and additive noise on the scanner raw data. Notably, these are non-optimal scenarios in terms of minimizing the L_2 error, and the original ICP tends to perform poorly.

Another issue for the ICP appears when there is a large angular displacement between the point sets. Efficient coarse registration techniques were developed for this kind of situation, but usually require additional preprocessing and a non-trivial parameter setup.

The problem of rigid registration has been widely explored by the literature, and several optimizations of the algorithm ICP, as well as many other algorithms for rigid registration, have been proposed. However, despite the recent advances, there is not a closed-form solution for estimating the best alignment in all situations.

Our proposal is to enhance the quality of the correspondences, altering the closest point criterion to use similarity between geometric features which describe each point on

the mesh. These features are computed from second-order tensors, which encode the local geometry of each point. With a proper coarse-to-fine strategy, the modified algorithm tends to reach a better solution and yield better results at situations where the original algorithm does not perform well.

1.1 PROBLEM DEFINITION

Given two distinct 3D point clouds $M = \{\overrightarrow{m_i} = (x_{m_i}, y_{m_i}, z_{m_i}), |M| = N_m\}$ and $D = \{\overrightarrow{d_j} = (x_{d_j}, y_{d_j}, z_{d_j}), |D| = N_d\}$, rigid registration aims to find the rigid transformation $T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, composed by a rotation and a translation, that best aligns M and D, minimizing a certain error metric E. The transformation T is global, i.e., unique for all points. M and D are at least partially overlapped, so that there are regions of the point clouds which have a similar surface geometry. In general, M and D represent complementary views of a real object.

The computation of the error metric E is generally point-based, and requires the establishment of a set of correspondence pairs between M and D. This set desirably contains points on the shared regions of both meshes and is defined as a base for the transformation estimation. The criteria used to compose this set is an important issue of the solution. For instance, the original version of the Iterative Closest Point algorithm, one of the most traditional fine registration methods, uses an Euclidean L_2 metric as both closest point criterion and error metric.

1.2 OBJECTIVES

The primary objective of this work is to enhance the quality of the correspondences of the ICP algorithm through the use of local geometric features. The secondary objectives are:

- To present a tensor voting process used to estimate the second-order tensors, briefly introducing tensor quantities and concepts;
- To introduce a shape similarity factor capable of expressing geometric similarity between points from the eigenvalues of their tensors;
- To generate a dataset with meshes that make possible a quantitative analysis of the

method performance on situations with additive noise, outliers and partial overlapping, which compound challenging scenarios for the ICP;

- To analyze the performance of the method on the situations mentioned above, highlighting the parameter ranges that yield better results on each situation.

1.3 CONTRIBUTIONS

The main contributions of this work are: 1) a Comparative Tensor Shape Factor (CTSF) to compare the shape of two tensors representing local geometry, and 2) the strategy to vary the relative weight of the CTSF and the Euclidean distance.

Our proposals greatly enhance the registration of point clouds mismatched by wide angle rotations. The secondary but important contributions include: a method for finding smooth normals given a k-neighborhood, a clearly stated comparison protocol to evaluate registration methods, a point cloud database composed by a large amount of events that can be used for future quantitative comparisons with our method.

Furthermore, to our knowledge this is the first work that uses a second-order tensor-based similarity factor to enhance the solution of the rigid registration problem.

1.4 RELATED WORKS

Research on surface registration has always been an interest field on computer vision and computer graphics, since several applications make use of this task. However, with the variety of applications, the notation used on the different approaches is not standardized, enforcing the importance of surveys and benchmarks in order to categorize and compare the methods, independently of the application. Throughout the surveys existent on the literature, we highlight the following works:

- Eggert et al. (1997): compares the performance of four different rigid transformation estimation methods;

- Rusinkiewicz and Levoy (2001): enumerates six possible stages of optimization on the ICP algorithm;

- Salvi et al. (2007): proposes a taxonomy for rigid registration methods, classifying them into coarse and fine, and subclassifying each one of the categories with respect

to different aspects, such as registration strategy, motion estimation / minimization strategy, kind of correspondence / efficient search and robustness;

- Tam et al. (2013): formulates the problem as data fitting, summarizing approaches for rigid and non-rigid registration and grouping them according to its constraints and optimization methods;

- Díez et al. (2015): focuses on coarse registration, proposing a pipelined classification and presenting the current state-of-art methods.

In this work, we will follow the taxonomy of Salvi et al. (2007), and summarize the state-of-art methods on coarse and fine registration separately. They differ in the objective: while coarse registration aims to compute an initial estimation of the rigid motion between two clouds of 3D points, fine registration uses an estimation of motion previously computed in order to converge to a more accurate solution. Among the coarse registration techniques, the methods will be presented according to the pipeline proposed by Díez et al. (2015), and in fine registration methods we chose to follow the optimization steps described in Rusinkiewicz and Levoy (2001), since most methods are modifications of the original ICP, previously mentioned.

1.4.1 COARSE REGISTRATION

Coarse registration is the alignment of two meshes initially in arbitrary positions. In general, coarse registration methods are based on the matching of reliable correspondences. Thus, this problem is closely related to the problem of keypoint detection and description on 3D point clouds, and good keypoint approaches tend to yield good results on 3D coarse registration.

Although quantitative comparison for coarse registration methods is rare, literature provides extensive benchmarks for 3D keypoint detection and description, such as the SHREC database (BRONSTEIN et al., 2010; BOYER et al., 2011). These benchmarks are applied not only for rigid registration, but also in other applications related to 3D keypoint detection, such as 3D pose detection and tracking, symmetry detection and query databases of 3D meshes. The emphasis of the evaluation on those benchmarks is on the repeatability of the algorithms on several scenarios, i.e. “how algorithms cope with certain classes of transformations and what is the strength of the transformations that

can be dealt with" (BOYER et al., 2011). Some of these scenarios are out of the scope of rigid registration, like deformation and partial scaling. Thus, in coarse rigid registration it is not mandatory that a good descriptor should be robust to all the repeatability tests proposed on the benchmarks. In our work, we take into account robustness to high angles, additive noise, outliers and partial overlapping situations.

A recent survey (DÍEZ et al., 2015) on coarse registration proposes a generic pipeline, which consists on three subtasks: a subsampling based on detection of feature points, the matching of descriptors based on the shape of such points and the search strategy. Sampling based on feature points is necessary in order to reduce the time spent, especially when the density of the point sets is high. Strategies to detect feature points on meshes include normal space sampling (RUSINKIEWICZ; LEVOY, 2001), maximally stable volumes (DONOSER; BISCHOF, 2006), heat kernel signatures (SUN et al., 2009), meshDoG (ZAHARESCU et al., 2009), a 3D version of Harris detector (SIPIRAN; BUSTOS, 2011) and intrinsic shape signatures (ZHONG, 2009). Among them, maximally stable volumes and heat kernel signatures reach good results at most cases of the SHREC database, returning fewer points with a high level of distinctiveness.

The description is the key step on the coarse registration process, in which correspondences are established according to a shape function, desirably invariant to translation, rotation and scaling. This shape function can be either a histogram or a signature, based on topological or geometrical features. Several descriptors are addressed on the survey, and among them the survey highlights the robustness of ISS (ZHONG, 2009) and SHOT (TOMBARI et al., 2010), the accuracy of Integral Invariants (POTTMANN et al., 2009) and HKS (SUN et al., 2009), and the computational time of the PCA (CHUNG et al., 1998).

As for the search strategy, the goal is to enhance the search process in order to find the best transformation between the points efficiently. Some of these strategies are algebraic surfaces (TAREL et al., 1998), robust global registration (GELFAND et al., 2005), RANSAC-based methods like the DARCES (CHEN et al., 1999) and the 4PCS (AIGER et al., 2008) and evolutionary game-theoretical frameworks (SANTAMARÍA et al., 2011), which is a recent topic (CIRUJEDA et al., 2015; ALBARELLI et al., 2015). The survey points out the 4PCS as currently the best search strategy.

The 4PCS is a method based on finding small congruent 4-point subsets, which effi-

ciently achieves good results in different situations, even in the presence of a high amount of outliers and additive noise. Although originally designed for coarse registration, 4PCS reaches results comparable to fine registration methods. More recently, a method named Super-4PCS (MELLADO et al., 2014) was proposed, enhancing the 4PCS in terms of speed and achieving linear complexity by the use of an efficient data indexing, while maintaining the alignment quality even in cases of low overlap.

The CTSF, main proposal of this work, consists on a shape comparison factor invariant to rotation, translation and global scaling, and therefore can be classified as a coarse descriptor. Thus, it can be used to establish reliable correspondences. However, although suitable for rigid registration, its use as a detector and its performance on descriptor repeatability tests tends to be limited, since it is theoretically affected by the presence of additive noise and also does not cope with topological alterations and distortions on the mesh, among others.

1.4.2 FINE REGISTRATION

As for the fine registration techniques, the most common approach is the traditional ICP, and the general effort of the literature is on optimizations of this algorithm.

Several optimizations were proposed so far to enhance the performance of the ICP in terms of speed, robustness and accuracy. The survey of Rusinkiewicz and Levoy (RUSINKIEWICZ; LEVOY, 2001) identifies six possible stages of the algorithm in which optimizations can be made:

- Selecting a subset of points (RUSINKIEWICZ; LEVOY, 2001; MASUDA et al., 1996; GELFAND et al., 2003);
- Changing the closest point criterion (GODIN et al., 1994; MEN et al., 2011; SHARP et al., 2002, 1999);
- Weighting the correspondences (RUSINKIEWICZ; LEVOY, 2001; GODIN et al., 1994);
- Rejecting pairs (CHETVERIKOV et al., 2002; DONG et al., 2014);
- Assigning an error metric (BOUAZIZ et al., 2013; TRUCCO et al., 1999; FITZGIBBON, 2003; MAIER-HEIN et al., 2012);

- Minimizing this error metric (BOUAZIZ et al., 2013; FITZGIBBON, 2003; SEGAL et al., 2009; YANG et al., 2013).

Taking into account their taxonomy, our method only modifies the closest point criterion by combining the CTSF and the Euclidean distance. In this aspect, we propose a novel successive approximation strategy that dynamically changes the closest point criterion, varying the relative weight of the shape factor and the Euclidean distance. The weights are altered every time the algorithm reaches a local optimum point, increasing the Euclidean distance influence. This strategy guides the algorithm from a coarse alignment, based on tensor similarity, to a fine alignment, computed from the Euclidean distance.

Many works propose modifications on the closest point criterion, using other information, such as color (GODIN et al., 1994; MEN et al., 2011) or geometric compatibility (SHARP et al., 1999, 2002), to enhance the quality of the correspondences. Godin et al. (1994) proposes a method called Iterative Compatible Closest Point that alters the distance metric to consider also similarity by color, also extensible to other intensity invariants. The work of Sharp et al. (1999, 2002) introduces the use of geometric invariant features such as curvature, moment invariants and spherical harmonics invariants on the ICP, altering the closest point criterion to a linear combination of Euclidean distance and feature compatibility. In the prior work, the matching of the points is a weighted linear combination of the invariant features with the Euclidean distance. In the more recent work, the method has been reformulated such that the features are weighted by a factor α^2, whose value is set at each iteration as the mean squared Euclidean distance from the data points to their closest points. This scheme implicitly reduces the weight of the invariant feature factor as the transformation bring both point clouds closer to each other. However, this work has as limitation the need of a point grid to compute the invariant features, which makes it only applicable on range image alignment. Our work uses a highly similar approach, considering a linear combination between the factors, but we use a different strategy of variation of the weight. Also, the CTSF is computed from a k-neighborhood that can be obtained from raw point data, which extends the application of our method to point clouds in general.

Discarding bad correspondences is a common strategy to deal with outliers and missing data. One of the methods that use it is the Trimmed ICP (CHETVERIKOV et al., 2002), which consists on discarding the $\epsilon\%$ correspondences that have worse Euclidean distance

values. An alternative is to discard correspondences according to a distance threshold instead of a percentage of the mesh. Due to its simplicity, the Trimmed ICP is used as a standard strategy to deal with partial overlapping, and some works adapt it to discard correspondences with different distance metrics (MAIER-HEIN et al., 2012; DONG et al., 2014).

The choice of the error metric and its minimization plays an important role on the quality of the result. The literature contains approaches that explore other minimization methods and other error metrics than the L_2 norm. The RICP (TRUCCO et al., 1999) proposes the use of a Least Median Squares regression to enhance the performance in the presence of outliers and with missing data. The LM-ICP (FITZGIBBON, 2003) uses the traditional Levenberg-Marquardt non-linear optimization algorithm on the parameter space, allowing the use of robust kernels and other error metrics. The authors claim that the results achieved are comparable to the original method in precision and speed. The EM-ICP (HERMANS et al., 2011) models the point clouds as Gaussian Mixture Models, adapting the ICP to an Expectation-Maximization process.

Among the recent approaches, the Sparse ICP (BOUAZIZ et al., 2013) uses L_p norms to change the error metric. The sparsity induced by the use of L_p norms when $p \in [0, 1]$ reduces the influence of outliers on the transformation estimation, but the optimization problem becomes non-convex and the minimization is made by Augmented Lagrangian methods. The Generalized ICP (SEGAL et al., 2009) associates each point to a covariance matrix, computed from a principal component analysis on its nearest neighbors, and assumes that the error between each point and its correspondence is drawn from a Gaussian distribution. The transformation is then estimated using a maximum likelihood estimation process. This method was later expanded (SERVOS; WASLANDER, 2014) to build a Multi-Channel kernel descriptor capable of incorporating features of the images, such as reflectance coefficients and color, to enhance the quality of the correspondences. The Anisotropic ICP (MAIER-HEIN et al., 2012) computes a cross-covariance matrix for each point and uses this information on the closest point computation and on the error minimization, adapting the problem to deal with anisotropic and inhomogeneous errors.

Our work links to the Generalized ICP and Anisotropic ICP since these works also express local geometry information of points on covariance matrices. However, to our knowledge, our method is the first rigid registration approach to treat local covariance

matrices as second-order tensors and use invariant tensor properties to compute similarity between patches on both sets. There is a previous method (REYES et al., 2007), different from the ICP, that uses second-order tensors computed by a voting process to solve the rigid registration problem, but this work differs from ours on the use of a complex geometric algebra formulation, while we use a single factor computed directly from the tensor eigenvalues.

A recent trend is to use Lie algebra properties to optimize in the parameter space. The GoICP (YANG et al., 2013) achieves globally optimal solutions under the L_2 error for any initial position, integrating the ICP minimization with a branch-and-bound search on a 3D motion space. The Trimmed ICP was later extended to a Lie Group parametrization formulation, the LieTrICP (DONG et al., 2014), which improves the accuracy on partial overlapping situations and select automatically the amount of points discarded, reformulating the error function to deal with anisotropic errors through the use of Lie algebra properties. The LieTrICP is able to retrieve anisotropic scale transformations, besides the rigid transformation.

For further details on recent advances, the survey of Tam et al. (2013) treats the registration as a data fitting problem and summarizes many approaches for the problem of rigid and non-rigid registration, classed with respect to the registration constraints and optimization methods. Salvi et al. (2007) classifies registration methods in coarse and fine, comparing results in terms of error, parameter variation and execution time on different examples. Despite the well stated error measurement protocol in various aspects of performance, the results presented on this survey are merely qualitative and lack statistical relevance. Unfortunately, most works on rigid registration present a weak experimental setup, with low statistical backup and results only for few cases.

2 FUNDAMENTALS

Our method has as input the single $3D$ position of each point, not requiring any other information. The basic assumption here is that when two points belonging to different viewpoints of the same object have the same neighborhood disposition, they are likely to represent the same region in two different views of an object. Thus, our method aims to use an estimation of the local geometry disposition of the points, encoded on second-order tensors, in the matching step of the ICP.

This chapter presents basic concepts of second-order tensors and the preprocessing stage of tensor voting, necessary to encode the geometric information of each point.

2.1 SECOND-ORDER TENSORS

A tensor is a generalization of the concept of vector and scalar. Mathematically, we represent a second-order tensor or rank-two tensor by a $n \times n$ matrix associated to a locality on on $\mathbb{R}^n$. In this work, the matrix dimension is 3×3, since our input is a set of $3D$ points, and we use a type of tensors known on the literature as orientation tensors.

An orientation tensor is a special case of a second-order tensor, in which the matrix representative of the tensor is a covariance matrix, therefore symmetric and positive-semidefinite. Since the eigenvalues of the orientation tensors are greater or equal to zero, the covariance matrix can be decomposed in three parts, each one encoding the contribution of the linear, planar and spherical part of the tensor.

$$T = (\lambda_1 - \lambda_2)T_1 + (\lambda_2 - \lambda_3)T_2 + (\lambda_3)T_3 \tag{2.1}$$

The tensor T_1 encodes the direction of the main eigenvector. T_2 represents the plane formed by the two main directions, and T_3 accumulates all the three main directions equally. Based on the eigenvalues λ_1, λ_2 and λ_3, we compute the tensor anisotropy coefficients c_l, c_s and c_p (WESTIN et al., 1997) as shown in Equations 2.2, 2.3 and 2.4, in order to describe the neighborhood geometry. We assume that the eigenvalues are sorted

in descending order, i.e. $\lambda_1 \geq \lambda_2 \geq \lambda_3$.

$$c_l = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3}, \tag{2.2}$$

$$c_p = \frac{2 \cdot (\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3}, \tag{2.3}$$

$$c_s = \frac{3 \cdot \lambda_3}{\lambda_1 + \lambda_2 + \lambda_3}. \tag{2.4}$$

If the tensor presents a high c_l (or linear coefficient), the main direction is predominant in relation to the other two. This is an indication of a high curvature region. In turn, a high c_p (planar coefficient) indicates that the neighborhood has two predominant directions. In this situation, the local geometry can be approximated by a plane, and the point is probably in a low curvature region. If the three eigenvalues have similar magnitudes, the spherical coefficient (c_s) is large, and no information can be inferred about the local geometry.

Figure 2.1 shows superquadric glyphs (KINDLMANN, 2004) that represent tensors with high linear, planar and spherical anisotropy, respectively. The figures on the rest of this work that represent graphically second-order tensors will follow that glyph representation.

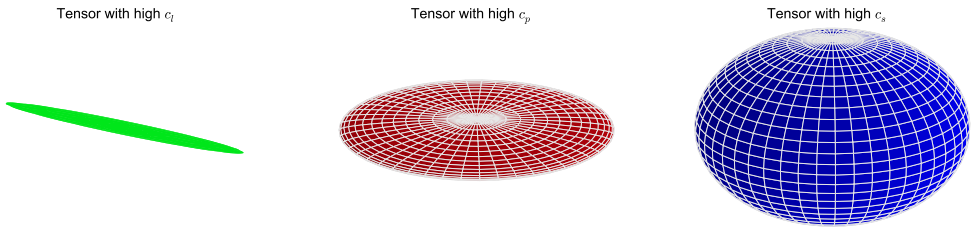


Figure 2.1: Representations of shapes of tensors with high c_l, c_p and c_s, respectively.

Our local geometry estimation method is based on the accumulation of vector votes from a neighborhood, weighted by an influence scalar function. The direct accumulation of vector votes is not appropriate, since vectors in opposite orientations would cancel themselves. Tensors are suitable to accumulate geometric information, since they are able to accumulate those votes regardless of the orientation of the vectors. Thus, two basic operations are used: the transformation of a vector in a stick tensor and the weighted sum of tensors.

A vector $\vec{a}$ can be encoded in an orientation tensor $\mathbf{A}$ through the product $\vec{a} \cdot \vec{a}^T$, which

generates a covariance matrix. This tensor has two null eigenvalues, and its main eigenvector is the direction pointed by $\vec{a}$. Also, the geometry information can be approximated by a stick, since the linear coefficient is high.

Consider, for example, two stick tensors $\mathbf{A}$ and $\mathbf{B}$, which represent vectors $\vec{a}$ and $\vec{b}$. The weighted sum $\mathbf{T}$ of these two tensors encodes the plane generated by $\vec{a}$ and $\vec{b}$.

$$\mathbf{T} = w_1 \cdot \mathbf{A} + w_2 \cdot \mathbf{B} \tag{2.5}$$

The geometry encoded by $\mathbf{T}$ will be planar (higher c_p) if the angle between $\vec{a}$ and $\vec{b}$ is 90°, and linear (higher c_l) if both vectors are colinear. The main direction of the resulting tensor is the line which bisects the angle between $\vec{a}$ and $\vec{b}$, and the eigenvalues are proportional to the colinearity between $\vec{a}$ and $\vec{b}$. The use of weight factors scale the influence of each direction, affecting directly the shape of the resulting tensor.

When more than two vectors encoded in stick tensors are summed, the information encoded on the resulting tensor tends to capture the main orientation of all tensors scaled by their norms and weights. Such influence is represented in the eigenvalues of the covariance matrix. If all the vectors belong to a plane, the tensor will have planar geometry, and if all the vectors point out to the same direction, the tensor will have a high c_l and linear geometry.

$$\mathbf{T} = \sum_i w_i \cdot \mathbf{A}_i \tag{2.6}$$

2.2 TENSOR ESTIMATION

To estimate the local geometry of both point sets, a preprocessing stage based on a tensor voting process is applied, aiming to estimate orientation tensors that represent the local geometry. The basic assumption is that when two points belonging to different viewpoints of the same object have the same neighborhood disposition, they are likely to represent a common region of one object in different views. Thus, the estimation of those orientation tensors generates additional information, that can be used on the ICP algorithm.

This preprocessing method has as input the single $3D$ position of each point, not requiring any other information. Since the method relies on nearest neighbors to compute the tensors, we assume that the input point cloud represents a surface. Volumetric repre-

sentations would add internal points among the neighbor lists, which would affect directly the output of the tensor estimation.

2.2.1 TENSOR VOTING FRAMEWORK

The preprocessing stage used in this work computes a second-order tensor for each point, representing its local geometric disposition. It is based on the method proposed by Vieira et al. (2004), and uses a tensor voting framework based on the well known work of Medioni et al. (2000); Mordohai and Medioni (2006). Tensor voting has many formulations and deals efficiently with a wide number of problems on computer vision and computer graphics, including surface reconstruction, stereo vision, edge detection, data repairing and optical flow, and is acknowledged for its robustness to outliers.

The tensor voting algorithm is the accumulation of the influence of the local neighborhood on a point, which generates a tensor that encodes an estimation of the local distribution of a feature. For each point, a voting field is generated. The point propagates its information as a vote to all his neighbors, and uses the information cast by them in order to generate the tensor.

The general formulation of the tensor voting algorithm is described by:

$$\mathbf{T}_p = \sum_{q \in N(p)} f_{qp} \cdot \mathbf{T}_{qp}, \tag{2.7}$$

where $N(p)$ is the neighborhood of the point, f_{qp} is a scalar field that weights the influence of each neighbor $q \in N(p)$, and $\mathbf{T}_{qp}$ is the vote that each neighbor q casts on the point p.

Our implementation of the tensor voting is composed by two steps. In the first step, a coarse estimation of the main curvature directions is obtained for each point. In the second, the previous estimation is enhanced, increasing the influence of coplanar structures on the neighborhood, and consequently the local planarity of the tensor. This is made by the use of a different distance metric, the distance over an elliptical trajectory. It is possible to perform the second step more than once, in order to enhance continuously the planarity of the tensors. With the second step, the influence of unstructured outliers on the point set tends to be attenuated, since those points are mostly not coplanar with the tangent plane defined by each point's normal vector.

In our case, the neighborhood of each point is represented by the list $L_k(p)$ of its k

nearest-neighbors sorted by their Euclidean distances. Conversely, we represent as $L_k^{-1}(p)$ the list of points that contains p into its k nearest neighbors. The method assumes that these lists are previously computed.

The tensor voting process is analogous to mathematical morphology techniques in 2D images, such as erosion and dilation. Those techniques use structuring elements to modify each pixel according to the information contained on its neighbors, enhancing or reducing the influence of structured information. Each voting step can be seen as an application of a structuring element.

2.2.2 FIRST PASS - RADIAL STRUCTURING ELEMENT

The first step builds for each point p a second-order tensor $\mathbf{T}_p$, which accumulates the weighted sum of stick tensors built from the vectors $\overrightarrow{pq}$, for each neighbor $q \in L_k(p)$. The influence function f_{qp} is a Gaussian decay proportional to the Euclidean distance between p and q with standard deviation σ_p. This deviation is proportional to the closeness of the neighbors, such that the farthest neighbor q_f has influence 0.01.

$$\sigma_p = \sqrt{\frac{||\overrightarrow{pq_f}||^2}{\ln 0.01}}. \tag{2.8}$$

The tensors $\mathbf{T}_{qp}$ are stick tensors built from a tensor product of a normalized vector $\widehat{v}_{qp}$ with itself transposed. In this step, the vote direction $\widehat{v}_{qp}$ is the normalized vector $\widehat{pq}$.

The output of the first step is the set of tensors $\mathbf{T}_p$:

$$\mathbf{T}_p = \sum_{q \in L_k(p)} f_{qp} \cdot \widehat{v}_{qp} \cdot \widehat{v}_{qp}^T = \sum_{q \in L_k(p)} e^{\frac{-||\overrightarrow{pq}||^2}{\sigma_p^2}} \cdot \widehat{pq} \cdot \widehat{pq}^T, \tag{2.9}$$

where the Gaussian influence function f_{qp} is proportional to the Euclidean distance between p and q, with standard deviation σ_p, as in Equation 2.8. The Equation 2.9 corresponds to the application of a 3D isotropic radial structuring element. The third main direction of $\mathbf{T}_p$ is a rough estimation of the normal on points with planar neighborhood.

2.2.3 SECOND PASS - COPLANAR STRUCTURING ELEMENT

The tensors $\mathbf{T}_p$ obtained in the first step are used as input to the second step. Here, another structuring element is applied on the point set, in order to enforce the influence of local coplanar structures. Differently from the first step, here each point p *casts* a stick vote on its neighbors q, based on a vector $\widehat{v}_{pq}$, and its influence on the tensor $\mathbf{S}_q$ is given by a function f_{pq}, proportional to the coplanarity between p and q. This function ensures that points aligned to the tangent plane have higher influence.

To estimate $\widehat{v}_{pq}$ and f_{pq}, we first bring all points to a different coordinate system, where the axes $(\widehat{x}, \widehat{y}, \widehat{z})$ are respectively aligned with the normalized eigenvectors $(\widehat{e_1}, \widehat{e_2}, \widehat{e_3})$ of $\mathbf{T}_p$. The transformation of each neighbor q to this system is obtained by the application of a rotation matrix R_p built from $\widehat{e_1}$, $\widehat{e_2}$ and $\widehat{e_3}$ on the vector $\overrightarrow{pq}$, generating a new point denoted q'. In this system, the correspondent point p' of the point p is the origin. Note that $\overrightarrow{pq}$ is a column vector (3×1).

$$q' = \begin{bmatrix} \widehat{e_1} \\ \widehat{e_2} \\ \widehat{e_3} \end{bmatrix} \cdot (\overrightarrow{pq}).$$

The next step is to express q' in spherical coordinates:

$$\begin{cases} \rho_{q'} = \sqrt{{q'}_x^2 + {q'}_y^2 + {q'}_z^2}, \\ \theta_{q'} = \tan^{-1} \dfrac{{q'}_y}{{q'}_x}, \\ \phi_{q'} = \tan^{-1} \dfrac{{q'}_z}{\sqrt{{q'}_x^2 + {q'}_y^2}}. \end{cases}$$

For each neighbor q', there is a unique ellipsoid E with eccentricity $\tan \alpha_{ellip}$ that is centered over the $\widehat{z'}$-axis and tangent to p' and q'. Figure 2.2 shows ellipses with different α_{ellip} parameters passing through an arbitrary point, in a $2D$ representation.

The coplanarity between p and a neighbor q is proportional to the distance d_e between p' and q' over this ellipsoid, which is given by:

$$d_e(p,q) = d_e(p',q') = \rho_{q'} \cdot \cos \phi_{q'} \cdot \left(1 + \left(2 - \frac{1}{\tan^2 \alpha_{ellip}}\right) \cdot \tan^2 \phi_{q'}\right)^{\frac{\tan^2 \alpha_{ellip}}{2 \cdot \tan^2 \alpha_{ellip} - 1}}. \quad (2.10)$$

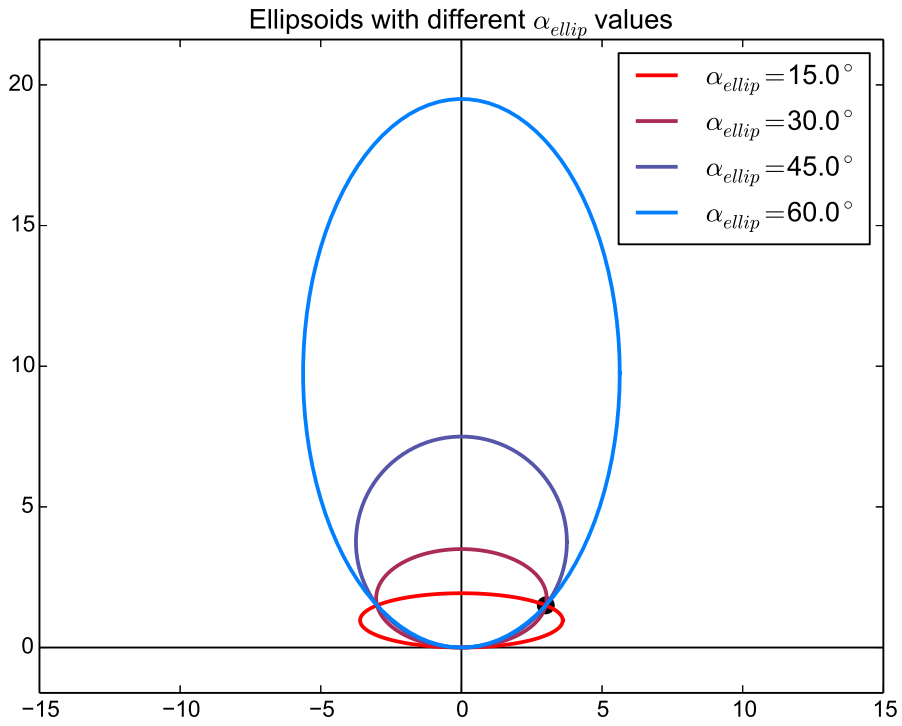


Figure 2.2: Family of ellipses (2D representation) passing through the point $(3.0, 1.5)$ with different values of α_{ellip}.

To avoid numerical instabilities on the computation of $d_e(p, q)$, the angle α_{ellip} must be at least $\tan^{-1} \frac{\sqrt{2}}{2} \approx 35.26°$. Smaller values of α_{ellip} would result in a negative base and a negative exponent, yielding an invalid operation.

From the spherical distance defined in Equation 2.10 and the standard deviation σ_p from Equation 2.8, we define the influence f_{pq}, exerted from p on each neighbor $q \in L_k(p)$. Like in previous step, the farthest point in $L_k(p)$, using Euclidean distance, has influence 0.01. Ideally, σ_p should be built such that the point with the bigger elliptical trajectory has influence of 0.01, but since our work applies the structuring element several times, this operation would require the recalculation of the k-nearest neighbors by the elliptical distance at each pass, which is computationally expensive.

The influence of points misaligned to the tangent plane is constrained by discarding points with $\tan \phi_{q'} > \tan \phi_{max}$.

$$f_{pq} = \begin{cases} e^{\dfrac{-d_e(p,q)}{\sigma_p^2}}, & \phi_{q'} \leq \phi_{max}, \\ 0.0, & \phi_{q'} > \phi_{max}. \end{cases}$$

The vote vector $\widehat{v}_{pq}$ requires the computation of the angle $\beta_{q'}$, which is the angle formed by the x'-axis and the line tangent to E at the point q'. The direction defined by the angle $\beta_{q'}$ is assumed to be an estimation of a vector that belongs to the tangent plane of q'. In function of α_{ellip} and $\phi_{q'}$, $\beta_{q'}$ is:

$$\beta_{q'} = \tan^{-1} \frac{2 \cdot \tan^2 \alpha_{ellip} \cdot \tan \phi_{q'}}{\tan^2 \alpha_{ellip} - \tan^2 \phi_{q'}}.$$

By the replacement of the angle $\phi_{q'}$ for $\beta_{q'}$ and the conversion of q' back to Euclidean coordinates, we obtain a vector $\widehat{v'}_{pq}$.

$$\widehat{v'}_{pq} = (\cos\theta_{q'} \cdot \cos\beta_{q'})\widehat{i} + (\sin\theta_{q'} \cdot \cos\beta_{q'})\widehat{j} + (\sin\beta_{q'})\widehat{k}. \tag{2.11}$$

The vote vector $\widehat{v}_{pq}$ is obtained by the application of the inverse rotation matrix R_p^{-1}:

$$\widehat{v}_{pq} = R_p^{-1} \cdot \widehat{v'}_{pq}.$$

Figure 2.3 shows a cut of the plan $y' = 0$ depicting the vector $\widehat{v}_{pq}$ and the angles $\phi_{q'}$ and $\beta_{q'}$ for an arbitrary point, in a case where $\alpha_{ellip} = 30°$.

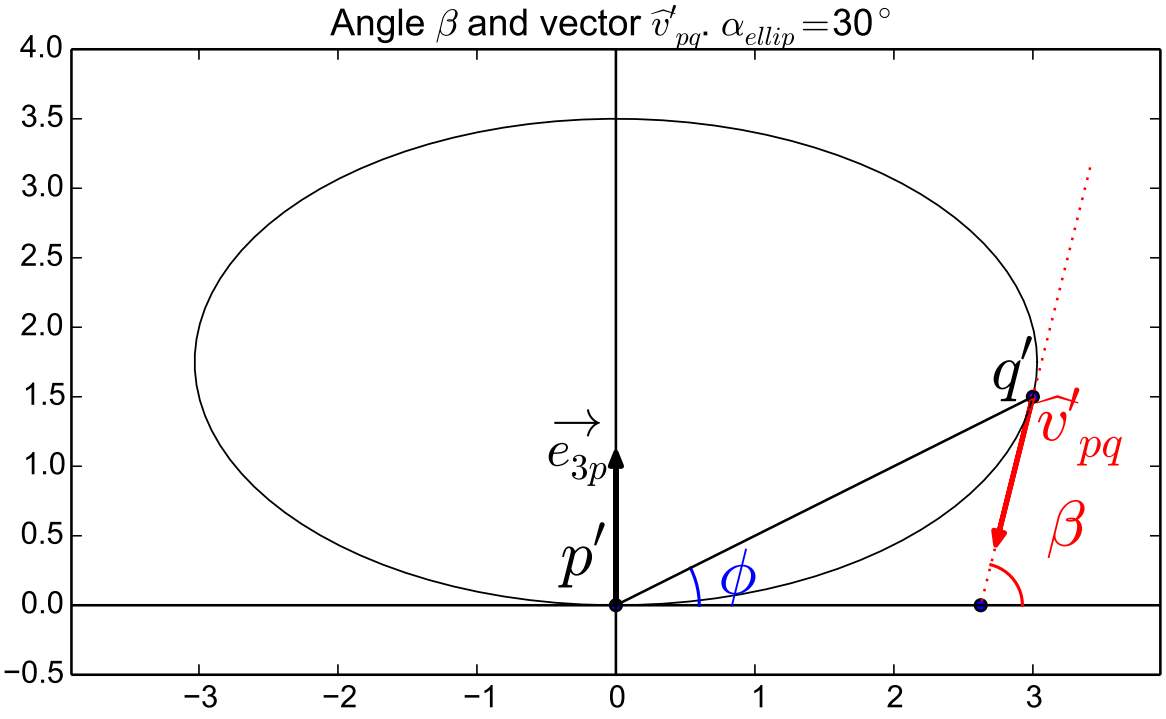


Figure 2.3: $2D$ geometric representation of the angles ϕ, β and vector $\widehat{v}_{pq}$ of an arbitrary point q'. Note that $\widehat{v}_{pq}$ is normalized.

The vote $\mathbf{S}_{pq}$ cast by p on a neighbor q is expressed by:

$$\mathbf{S}_{pq} = f_{pq} \cdot \widehat{v}_{pq} \cdot \widehat{v}_{pq}^{T}.$$

Finally, the resulting tensor $\mathbf{S}_p$ for a point p is composed by the weighted sum of the tensors built from the votes received on the point, cast by all the points that have p as a neighbor:

$$\mathbf{S}_p = \sum_{q \in L_k^{-1}(p)} \mathbf{S}_{qp} = \sum_{q \in L_k^{-1}(p)} f_{qp} \cdot \widehat{v}_{qp} \cdot \widehat{v}_{qp}^{T}. \tag{2.12}$$

2.2.4 IMPROVEMENTS AND DISCUSSION ABOUT THE PARAMETERS

The differences between the original method Vieira et al. (2004) and the one proposed and used in this work are:

1. We propose an iterative reapplication of the second step, through the recalculation of the tensors using with $\mathbf{S_p}$ as input. With this process, the planarity properties of the final tensors are continuously enhanced. The method stops when the average c_p stops improving or a maximum number of iterations is reached. In all experiments, we use a maximum of 100 iterations. In Subsection 2.2.4.1, we show the advantages of this iterative reapplication of the second step.

2. The original method uses all the points of the cloud as neighborhood and sets a default value for the standard deviation, equal for all points. We use a k-neighborhood and set the value of σ proportional to the distance to the k-th neighbor. In Subsection 2.2.4.2, we discuss the effects of using this approach.

3. The original method uses the normal information for surface reconstruction. Thus, the computation of the vector $\widehat{v}_{pq}$ is set to encode the normal in the first main direction of the tensor. Our implementation, in turn, encodes the normal in the third eigenvector, since the region geometry is as important as the normal itself in our work.

4. The original approach treats α_{ellip} and ϕ_{max} as the same variable, while we choose to deattach them. The consequences are discussed in Subsections 2.2.4.3 and 2.2.4.4,

and the effects of different parameters on the rigid registration are shown in the Experimental Results, on Subsection 4.3.2.

Before introducing the CTSF and the proposed method, we chose to show visual results of the tensor estimation method, and discuss the effects of the variation of each parameter of the method on those results. The next Subsections deal with the iterative enhancement process, the size of the neighborhood k and the angles α_{ellip} and ϕ_{max}.

2.2.4.1 Enhancement obtained by the iterative scheme

Here we aim to show the effectiveness of the iterative scheme, by comparing visual results of the tensors obtained after the radial structuring element application, after a pass of the coplanar structuring element and after the iterative coplanarity enhancement. As stated on the beginning of this Section, second-order tensors here are graphically represented by superquadric glyphs, using the parametrization described in Kindlmann (2004). The color indicates the c_p of each tensor, and tensors highlighted in hotter colors have higher planarity coefficients. On all the examples we consider $k = 25\%$ of the points on the original Octopus mesh, i.e. $k = 956$, in order to provide better visual hints about the performance of the algorithm. The parameters α_{ellip} and ϕ_{max} are set with the default value of 45°. We show that the planarity of the tensors is enhanced by the use of the iterative method even in hard situations, such as point clouds with outliers and additive noise. Figure

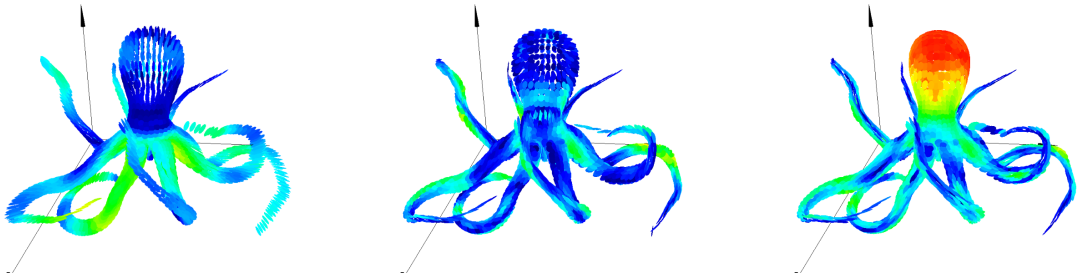
Figure 2.4: Color of the tensors according to its c_p. Reddish tensors are more planar.

Figure 2.5 shows the differences between the tensors obtained after the application of the radial structuring element, after one pass of the second structuring element, and the final result, after six applications of the second structuring element on the original *Octopus* point cloud. In this case, the method stopped after six iterations, because from the sixth to the seventh iteration the average planarity coefficient decreased. The iterative process effectively increases the planarity of the tensors of the point cloud, which is particularly notable on the head of the octopus, the more planar region of this point cloud.

Figure 2.6 shows an example of tensor estimation with a large amount of outliers. In this case, we added 200% of the original number of points of the Octopus point set, gener-

ated by an uniform distribution inside a sphere with radius $r = 2$, on a normalized point cloud for which the biggest side is 1. The difference of the planarity coefficients between Figures 2.8b and 2.8c indicates that the iterative enhancement is effective in acquiring better geometry information of planar regions even with unrealistic outlier amounts, which is highlighted on the head of the Octopus. The method achieves good results with unstructured outliers inside the sphere. The presence of structured outliers, in turn, would affect the result since the tensors would encode structures generated by the outliers.

Additive noise is naturally is a hard scenario for the tensor estimation, since the smoothness of the neighborhood is compromised. In our work, the noise was synthetically added to each point and is generated by two random variables: a normalized direction, generated by a random isotropic distribution on a sphere, and a magnitude factor generated by a Gaussian random variable. The scale factor is generated on an interval for which the biggest value is proportional to the larger bounding box side by a factor δ, in order to control the additive noise strength. Results obtained show that for $\delta = 0.01$ the

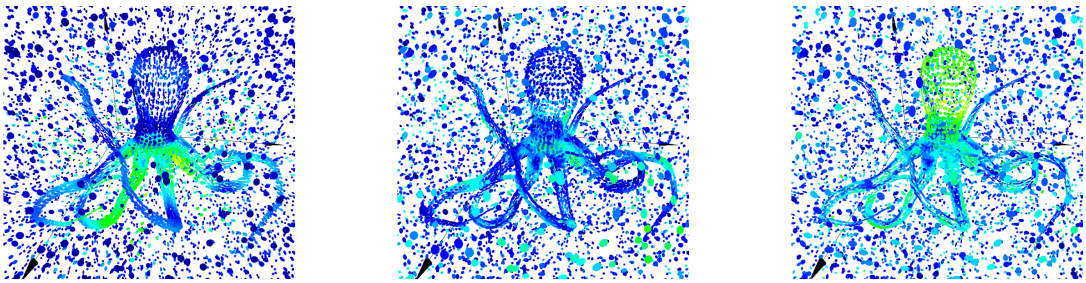


(a) After the radial structuring element.

(b) After one pass of the coplanar structuring element.

(c) Final result, after six passes of the coplanar structuring element

Figure 2.5: Orientation tensors for the *Octopus* with $k = 25\%$, $\alpha_{ellip} = \phi_{max} = 45°$.



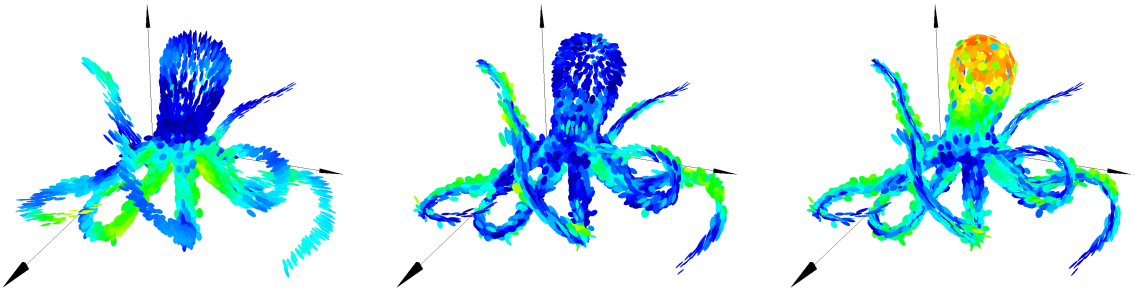
(a) After the radial structuring element.

(b) After one pass of the coplanar structuring element.

(c) Final result, after four passes of the coplanar structuring element

Figure 2.6: Orientation tensors for the *Octopus* with 200% of outliers, $k = 25\%, \alpha_{ellip} = \phi_{max} = 45°$, and 200% of outliers.

method is still capable of encoding planarity information. As we can note in Figure 2.7, tensors on the head of the octopus still have large planarity coefficient values. However, for $\delta = 0.05$ our method is unable to reach good results even after the iterative enhancement application, as shown in Figure 2.8.

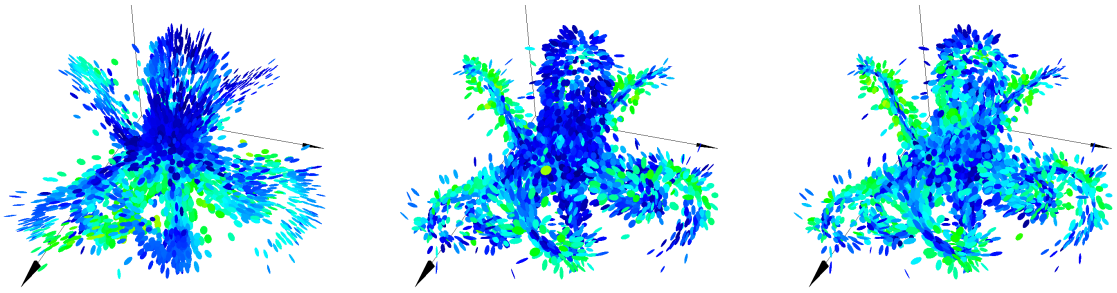


(a) After the radial structuring element.

(b) After one pass of the coplanar structuring element.

(c) Final result, after six passes of the coplanar structuring element

Figure 2.7: Orientation tensors for the *Octopus* with $k = 25\%, \alpha_{ellip} = \phi_{max} = 45°$. In this case, the scale of the additive noise is up to 1% of the biggest bounding box side.



(a) After the radial structuring element.

(b) After one pass of the coplanar structuring element.

(c) Final result, after four passes of the coplanar structuring element

Figure 2.8: Orientation tensors for the *Octopus* with $k = 25\%, \alpha_{ellip} = \phi_{max} = 45°$. In this case, the scale of the additive noise is up to 5% of the biggest bounding box side.

2.2.4.2 Size of the neighborhood

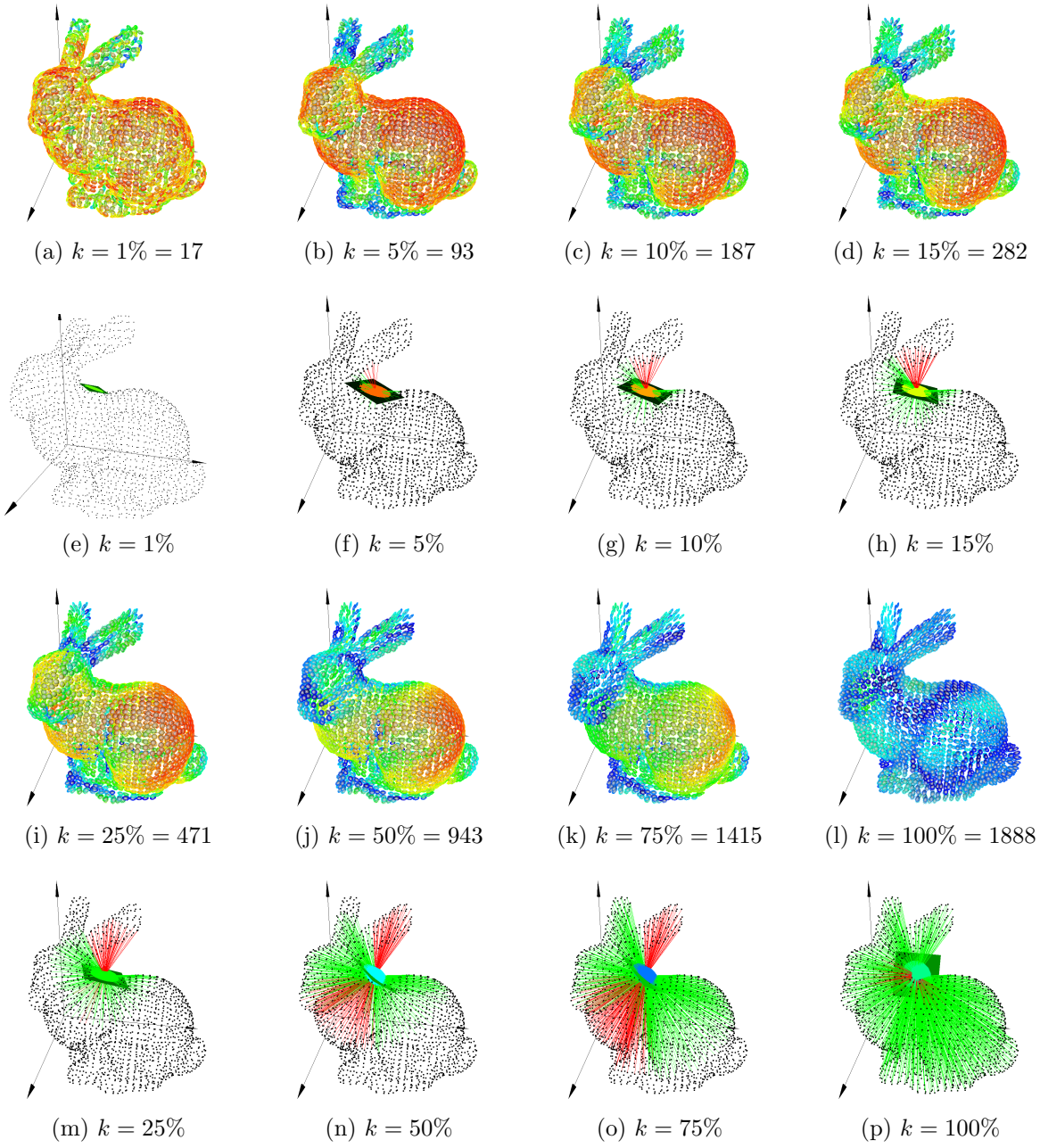
The number of neighbors k is an important parameter on the tensor estimation process. This value affects the scale of the neighborhood and the computation of σ_p. Figure 2.9 shows examples of output tensors of the method for different values of k on the Bunny point cloud and the neighborhood of a point for each value of k. If the value of k is very small, the tensor may have insufficient information of the local geometry, as shown

in the result with $k = 1\%$. In counterpart, for $k = 100\%$, the resulting tensor encodes information on distant points of the set. This explains the fact that the planarity of the tensors is generally lower with the increase of k. Subfigure 2.9p shows that larger neighborhoods contain points which not necessarily are desirable. As example, a point on the back of the bunny consider neighbors on its ear. However, those points have small influence on the tensor estimation, and some of them are disconsidered through the ϕ_{max} angle constraint, which is represented by the red edges at the subfloats (e)-(h) and (m)-(p). On larger neighborhoods, more points are disconsidered through the constraint.

As the size of the neighborhood increases, the k-th nearest neighbor gets further from the point, which increases the standard deviation of the Gaussian influence (Eq. 2.8). In this case, the decay caused by the difference of distances is attenuated for close neighbors, which makes their influence more similar. The practical consequence is a tendency of the algorithm to produce tensors that are smoother in relation to its neighbors, representing the relative distribution of the point with respect to a bigger region of the mesh.

The case where $k = 100\%$ can be seen as a degenerate case, in which using all the points as neighbors yield a plane that is not tangent to the surface. However, our latter results at registration suggests that this is not necessarily bad for our application. This is an interesting issue and will be discussed with more details on Chapter 4.

Our formulation of considering σ_p dependent of the k-th nearest neighbor makes the method sensitive to the density of the point cloud. If this density is not homogeneous, σ_p will have different values on different regions of the point cloud, and a decay based on a distance threshold can produce more uniform results. It is easy to adapt our approach to consider a distance threshold, but this would require a parameter calibration for each point cloud, which would be highly sensitive to its scale. On the other hand, the use of a fixed number of neighbors or percentual value of the number of points makes easier the parameter tuning and analysis.



(a) $k = 1\% = 17$ (b) $k = 5\% = 93$ (c) $k = 10\% = 187$ (d) $k = 15\% = 282$

(e) $k = 1\%$ (f) $k = 5\%$ (g) $k = 10\%$ (h) $k = 15\%$

(i) $k = 25\% = 471$ (j) $k = 50\% = 943$ (k) $k = 75\% = 1415$ (l) $k = 100\% = 1888$

(m) $k = 25\%$ (n) $k = 50\%$ (o) $k = 75\%$ (p) $k = 100\%$

Figure 2.9: (a)-(d), (i)-(l): Orientation tensors for the *Bunny* point cloud with larger values of k, set as percentuals of the number of points. Tensors highlighted in hotter colors have higher planarity coefficients. (e)-(h), (m)-(p): Neighborhood and estimated tensor of a point on the back of the Bunny, varying the amount of neighbors. Neighbors with green edges are below the angle $\phi_{max} = 45°$ constraint, and neighbors with red edges are disconsidered for having $\phi > \phi_{max}$. Note that as k gets bigger, the number of red tensors (i.e. with higher c_p) diminish.

2.2.4.3 Angle ϕ_{max}

The restriction on the angle ϕ_{max} constrains the influence of points misaligned to the tangent plane, defined by the normal of p. Higher values of ϕ_{max} tend to produce smoother surfaces, while smaller values allow more details at cost of robustness to outliers (VIEIRA et al., 2004). The choice of 45° is a mid term of both cases. The Figure 2.10 represents a cut on the plan $y' = 0$, showing influence isolines between the origin and points over their respective elliptical trajectories.

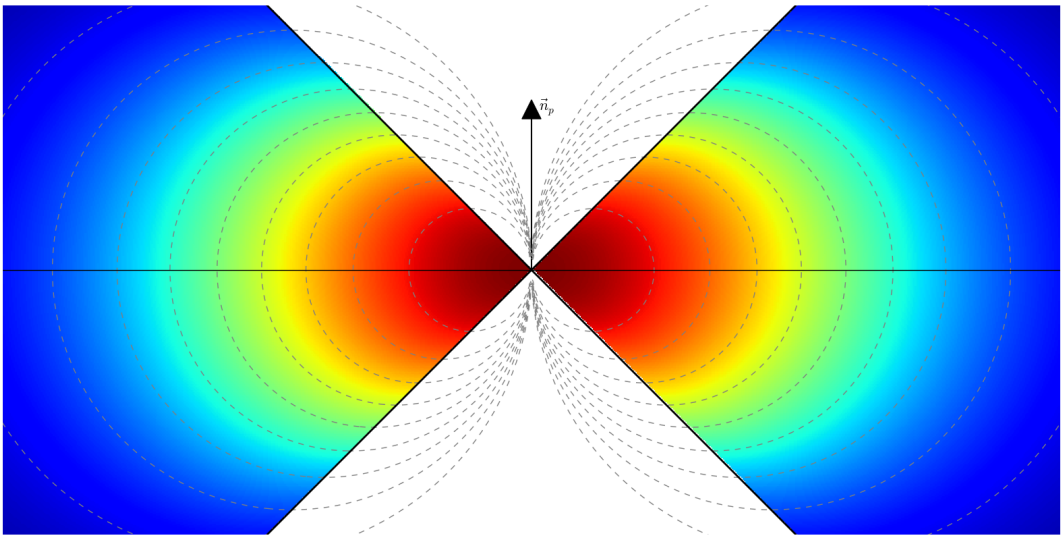


Figure 2.10: Influence over the elliptical trajectory, cut on the plan $y' = 0$: hotter colors means greater influence. $\phi_{max} = \alpha_{ellip} = 45°$

The original approach sets $\phi_{max} = \alpha_{ellip}$ in all cases. In our work, we assume that these values can be different but highly recommend that the angle ϕ_{max} should be smaller than α_{ellip}. The direction of the vote vector $\widehat{v}_{pq}$ is reverted for points with $\phi > \alpha_{ellip}$, since the curvature sign changes in that case. Figure 2.11 shows three examples of vote vectors for points with $\phi = 30°$, 45° and 60°, in a case where $\alpha_{ellip} = 45°$. The vector $\widehat{v}_{p_2q'}$ indicates a divergent direction in relation to the central point, and should not be considered.

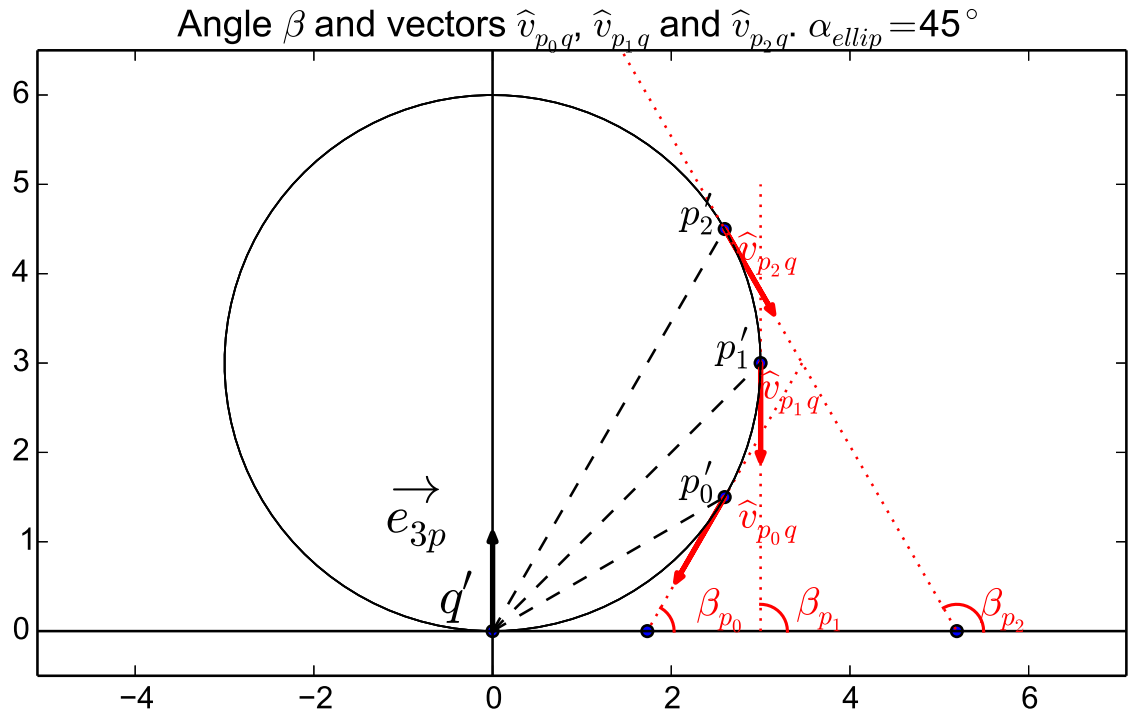
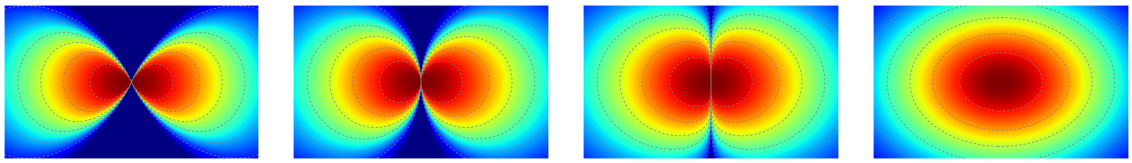


Figure 2.11: $2D$ geometric representation of the angles ϕ, β and vector $\widehat{v}_{pq}$ of an arbitrary point q'. Note that $\widehat{v}_{pq}$ is normalized.

2.2.4.4 Angle α_{ellip}

The parameter α_{ellip} impacts on the calculation of the angle $\beta_{q'}$ and the influence factor f_{pq}. The distance d_e is a combination between the Euclidean distance and the coplanarity with the tangent plane. Smaller α_{ellip} values give more influence to the coplanarity. As α_{ellip} gets closer to 90°, the influence field converges to a sphere, which is the first step structuring element, based solely on the Euclidean distance. Figure 2.12 represent the influence fields elements with different α_{ellip} values, respectively 36°, 45°, 60° and 90°.



(a) $\alpha_{ellip} = 36°$ (b) $\alpha_{ellip} = 45°$ (c) $\alpha_{ellip} = 60°$ (d) $\alpha_{ellip} = 90°$

Figure 2.12: Influence over the elliptical trajectory for different values of α_{ellip}, respectively 36°, 45°, 60° and 90°, on a cut on the plan $y' = 0$: hotter colors means greater influence. Note that when $\alpha_{ellip} = 90°$, the structuring element adquires the shape of a sphere.

3 PROPOSED METHOD

After the computation of the tensors, we have a local estimation of the neighborhood distribution for each point of the point cloud. Points that have a similar neighborhood disposition tend to have associated tensors with similar eigenvalues. Conversely, if we can measure dissimilarity between eigenvalues, we can express numerically the dissimilarity between the neighborhood distribution of two points, and this can be used to enhance the quality of correspondences between two meshes.

For this, we introduce a a shape dissimilarity factor between tensors, named Comparative Tensor Shape Factor (CTSF), and propose an adaptation to the matching step of the ICP in order to use this factor.

In the next Sections we describe the computation of the dissimilarity between the tensors, and the coarse-to-fine weighting scheme used on the ICP.

3.1 COMPARATIVE TENSOR SHAPE FACTOR - CTSF

We define a factor to compare the normalized shapes called Comparative Tensor Shape Factor (CTSF). The CTSF between two tensors $\mathbf{S}_1$ and $\mathbf{S}_2$ is given by:

$$CTSF\left(\mathbf{S}_1, \mathbf{S}_2\right) = \sum_{k=1}^{3} \left(\lambda_k^{\widehat{\mathbf{S}}_1} - \lambda_k^{\widehat{\mathbf{S}}_2}\right)^2. \tag{3.1}$$

where $\widehat{\mathbf{S}}_i$ is the normalized form of a tensor $\mathbf{S}_i$, using the L_2 matrix norm, and $\lambda_k^{\widehat{\mathbf{S}}_m}$ is the k-th greatest eigenvalue of the tensor $\widehat{\mathbf{S}}_m$. Bigger values of CTSF indicate tensors with dissimilar shapes.

Geometrically, a second-order tensor can be represented as an ellipsoid with axes proportional to its eigenvalues. Two ellipsoids have the same shape if the proportion of their axes, i.e. matrix eigenvalues, is the same. In this aspect, the CTSF represents the difference of shape proportions between two ellipsoids.

By normalizing the tensors of both Model and Data point sets, we can compare the local geometry despite the densities involved. As a consequence, the difference of magnitude of the original tensors is irrelevant in our method.

The eigenvalues of a tensor are invariant to rigid transformations, due to the isometric

nature of these transformations. Therefore, the CTSF is suitable to the ICP matching step as a compatibility factor between two points, particularly when combined with a tensor estimation algorithm that provides consistent input tensors. Figure 3.1 shows some cases of high and low CTSF between two tensors.

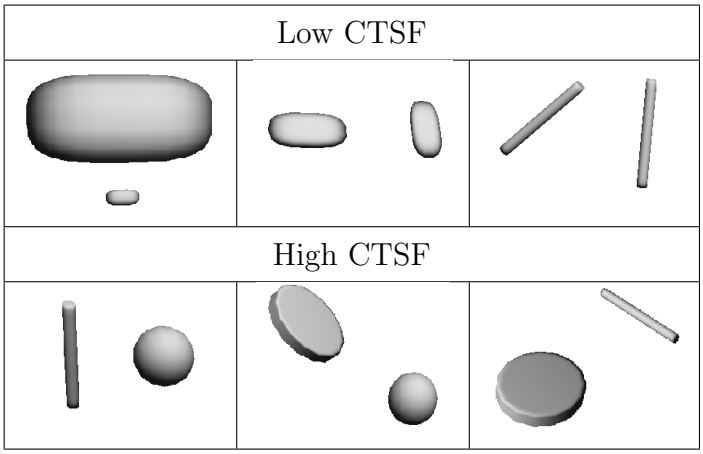


Table 3.1: Examples of how the CTSF is affected by the geometry of planar tensors. Note that the CTSF is invariant to the to orientation of the tensors and to their magnitude, due to the normalization.

Mathematically, we map points and their associated tensors into a space in which the axes are the normalized eigenvalues $\lambda_1, \lambda_2, \lambda_3$, in which $\lambda_1 \geq \lambda_2 \geq \lambda_3$, and $\sum_{k=1}^{3} \lambda_k^2 = 1$. Thus, the representation of each point on this space is a normalized vector. Since the eigenvalues are assumed to be ordered, the domain of the vectors is a region inside a unit sphere, where $\lambda_1 \in [\frac{\sqrt{3}}{3}, 1]$, $\lambda_2 \in [0, \frac{\sqrt{2}}{2}]$, $\lambda_3 \in [0, \frac{\sqrt{3}}{3}]$. The CTSF is the squared Euclidean distance between two vectors in this domain. The maximum distance on this space is the one between a stick and a ball tensor.

Other eigenvalue comparison factors can be defined, such as the difference between the anisotropy coefficients. We chose the direct comparison between the eigenvalues for the simplicity and generalization capacity of this factor.

This measurement can also be generalized as a shape dissimilarity factor for second-order tensors with higher dimensions, which can be useful in problems other than rigid registration. In this case, each eigenvalue λ_i is in the interval $[0, \frac{\sqrt{i}}{i}]$ for $i > 1$. The CTSF for N-dimensional tensors is defined as:

$$CTSF\left(\mathbf{S}_1, \mathbf{S}_2\right) = \sum_{k=1}^{N} \left(\lambda_k^{\widehat{\mathbf{S}}_1} - \lambda_k^{\widehat{\mathbf{S}}_2}\right)^2.$$

3.2 MODIFIED ITERATIVE CLOSEST POINT

Assuming that both point sets of the ICP are different visions from the same object or surface and that the sampling rate of the sets is the same, the geometric neighborhood disposition of a point $m_i \in M$ is the same of its correspondent $d_j \in D$. Thus, the tensor $\mathbf{S}_{m_i}$ has the same shape of $\mathbf{S}_{d_j}$, and the CTSF between them tends to zero.

The modification on the ICP proposed by this work aims to benefit from the invariance to rigid transformations of the CTSF in order to provide better correspondences on situations where the closest point provided by Euclidean distance can be inaccurate. However, if only shape information is used in the matching step, the alignment obtained is coarse, and small details on the adjustment of the sets are compromised. Thereby, we consider the distance in the matching step as a combination of the CTSF and the Euclidean distance, using a weighting strategy to control their relative influence.

Our method uses a weighting factor w to combine CTSF and Euclidean distance (ED), and controls the variation of the parameter when the algorithm reaches a local optimum of the error function. Let i be the number of local optima reached. We define the matching function as:

$$d(p,q) = ED(p,q) + w_i \cdot CTSF(\mathbf{S}_p, \mathbf{S}_q), \tag{3.2}$$

$$w_i = w_0 \cdot b^i, b < 1 \text{ and } 0 \leq w_i < w_0 \tag{3.3}$$

where the parameter b impacts on the variation of the relative weight between CTSF and Euclidean distance at each ICP loop, and w_0 is the initial influence of the Euclidean distance, most likely a small value. In practice, the weight of the CTSF is divided by b every time a local optimum is reached. The algorithm stops when $w_i \approx 0$, so that only Euclidean distance is considered on its last loop. With this process we are able to guide the solution in order to give more importance to the tensor dissimilarity in the first iterations, coarsely recovering the transformation for higher angle displacements, and to reach fine alignment when the sets are close enough, condition in which the matching based on Euclidean distance (original ICP formulation) tends to work better. We set $w_0 = 10000$ and $b = 0.1$ as default values of the parameters, but show results with different values of b. Our method is summarized on Algorithm 1.

The CTSF and the Euclidean distance are measurements of distinct magnitudes, since

Algorithm 1: Modified Version of the ICP algorithm

Require: M, D, k, a, w_0
Tensor Estimation of M
Tensor Estimation of D
$i \leftarrow 0$
$w_i \leftarrow w_0$
while $w_i > 10^{-6}$ **do**
    $C \leftarrow$ Nearest Neighbors(M, D, w_i, $1 - w_i$)
    $T \leftarrow$ Transformation Estimation (D, C)
    if RMS($T \cdot D$, M) < RMS(D, M) **then**
        update T
    else
        $i \leftarrow i + 1$
        $w_i \leftarrow b^{(\log_a w_0) - i}$
    end if
end while
return T

one of them is a distance factor and the other one is a shape dissimilarity factor.

In the optimization aspect, each step of the weight variation is an execution of the ICP algorithm with a different relative weight associated to the CTSF on the matching of points. The weight variation works as a coarse-to-fine successive approximation method. Each step of the process is a scale reduction of the CTSF in comparison to the Euclidean distance.

The value of b controls the reduction on the weight factor of each step of the process. With smaller values, the method perform fewer steps, and in most cases fewer iterations. For values near 1, the method performs more iterations and possibly reaches better results, since the landscape of the function is changed more smoothly. Therefore, an increase on b implies on an increase of the precision of the transformation estimation and bigger chances of convergence. The downside is that a linear increase on the value of b implies on an exponential increase on the number of iterations in this formulation.

Since our algorithm alters the matching step of the Iterative Closest Point, it can be used alongside any minimization strategy. In our experimental setup, we modify both the original ICP and the Sparse ICP (BOUAZIZ et al., 2013) to use the CTSF in the matching step, combined with the weight variation strategy.

4 EXPERIMENTAL RESULTS

In fine rigid registration literature, in order to highlight the good performance of the proposed method, most works present only qualitative comparisons with other methods. These comparisons are usually made with a small number of point clouds and on situations predetermined by the author. This type of evaluation is generally inconclusive, because the behavior observed in a single case may not describe the overall performance of the method. The quantitative evaluation of methods on many randomly generated events would be a more suitable form to describe the general performance. However, this type of analysis is hard, since the time spent on each execution is big, especially on larger point sets. Furthermore, large initial rotations are not usually a matter of concern for fine registration methods since they assume that both sets are coarsely pre-aligned.

Our work presents an experimental setup for quantitative evaluation of registration methods. For this, we generate a large dataset composed of transformations between two point clouds that include cases with wide angle initialization, additive noise, outliers and partial overlapping. With this dataset, we can characterize the behavior of the methods with different parameter values and on different situations.

In this chapter we describe the experimental setup of this work, presenting the point clouds used on the dataset, the procedures for generation of additive noise and outliers, the methods used on the evaluation and the results obtained.

4.1 DATASET

The dataset used in our work is composed by synthetically generated events that deal with usual issues on rigid registration. We consider an event as an alignment situation between two point clouds. Our dataset has two types of events: the first type contains point clouds with outliers and additive noise and the second deals with partially overlapped point clouds, also synthetically generated.

The base point clouds used in the generation of the events were sampled versions of

the Bunny[1], the Happy Buddha[1], the Octopus[2] and the Genus[3]. The sampling process was necessary due to the large number of executions performed on the experimental setup, in order to reduce the time spent on each event. For the Bunny point cloud we use the smallest zippered version available online. The other sampled point clouds were obtained through a Poisson-disk sampling algorithm (CORSINI et al., 2012) available on MeshLab[4].

Those point sets were chosen because they have different geometric features: the majority of points of the Bunny have a smooth neighborhood, except the points on the ears and on the feet, which are high curvature regions. The Happy Buddha, despite being composed mostly by small planar patches, has the geometry similar to a dumbbell, presenting a radial symmetry that make it a harder point set, particularly on wider angles. The Octopus, on the other side, has mostly high curvature regions. The Genus is a point cloud composed mostly by planar regions, but can be challenging due to its symmetry axes, increasing the chance of misalignment. Figure 4.1 shows the point clouds that compose our dataset.

We scale the point sets to a bounding box whose biggest side is 1.0, keeping the original aspect ratio. This normalization is the first step of the event generation and is applied before the rigid transformation, the addition of outliers and additive noise and the generation of the subsets on the partial overlapping case. This way, the error measurements for all the point clouds are on a similar order of magnitude.

For wide angle result evaluation, we sample the rotation angle interval $[0° - 180°]$ at each $15°$. The dataset has the same number of events for each angle and parameter configuration. In each event, an axis-angle rotation is applied in one of the point clouds, through the generation of a random unit vector $\vec{v}$ following an isotropic distribution in an unit sphere. One of the meshes is rotated around $\vec{v}$. The transformation does not include a translation part. We also consider this a more challenging scenario for the ICP, since there is a bigger chance of reaching a wrong local minima, specially on wide angles where the sets are incorrectly prealigned.

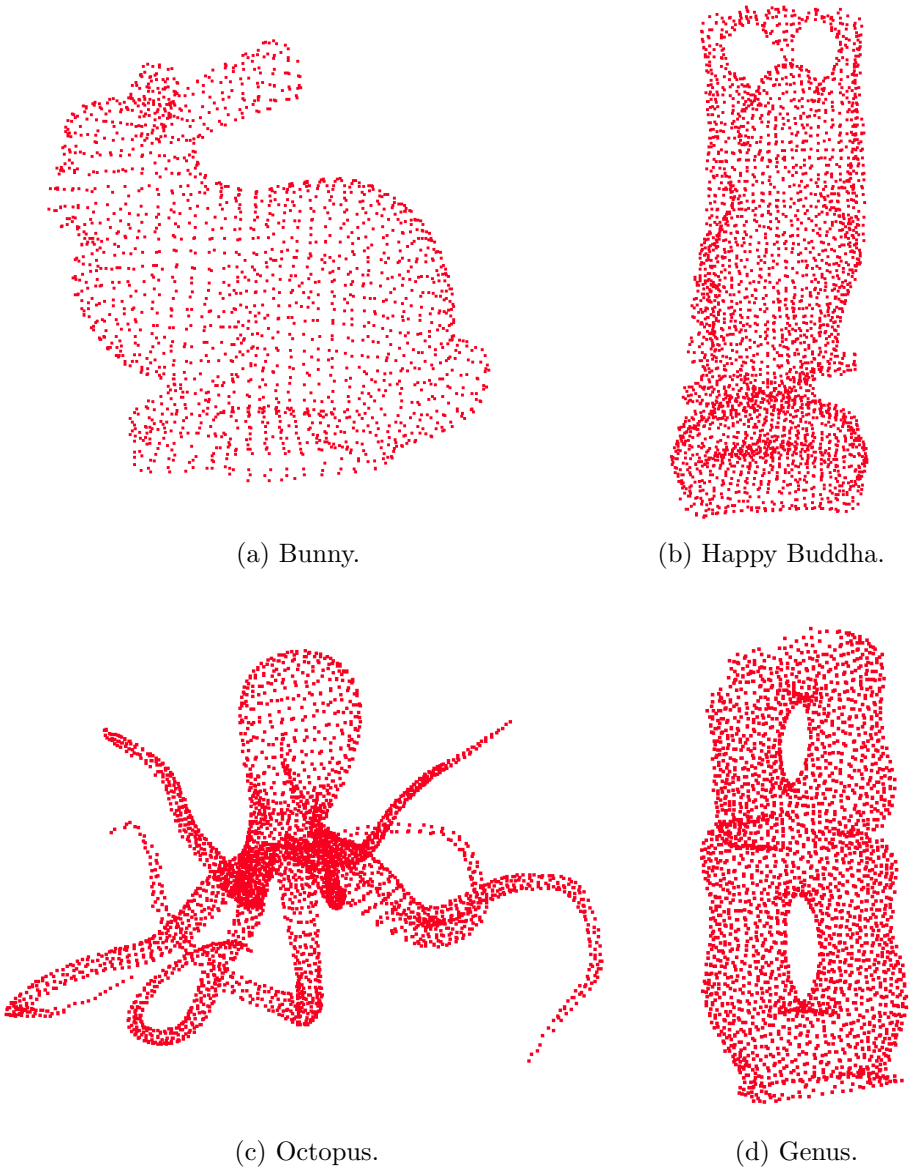
Additive noise, outliers and partial overlapping are generated randomly by synthetic

[1]Provided by Stanford University Computer Graphics Laboratory on `http://graphics.stanford.edu/data/3Dscanrep/` .

[2]Provided courtesy of INRIA by the AIM@SHAPE-VISIONAIR Shape Repository.

[3]Provided by École Polytechnique Fédérale de Lausanne Computer Graphics and Geometry Laboratory on `http://lgg.epfl.ch/statues_dataset.php`

[4]`http://meshlab.sourceforge.net`



(a) Bunny. (b) Happy Buddha.

(c) Octopus. (d) Genus.

Figure 4.1: The point sets used on the experiment: (a): Bunny (containing 1889 points). (b): Happy Buddha (3118 points). (c): Octopus (3822 points). (d): Genus (2711 points).

processes, with a different seed for each process. All the random variables are generated by a Mersenne Twister pseudo-random number generator. The next Subsections detail the synthetic algorithms for generation of events with additive noise, outliers and partial overlapped point clouds.

4.1.1 ADDITIVE NOISE AND OUTLIERS

The first type of event of the dataset addresses realistic non-optimal scenarios with additive noise and outliers. The point sets are fully overlapped, that is, all the points on the base mesh are used to generate both Model and Data meshes, which are distinguished by

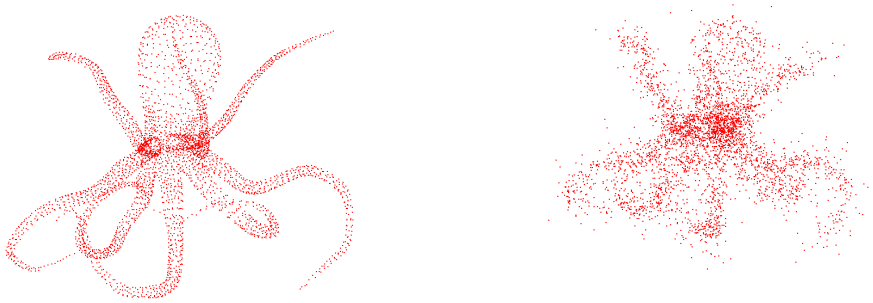
the application of a rigid transformation and the addition of outliers and additive noise on both. The seeds used for generation of outliers and additive noise are distinct for each point cloud, therefore the two point clouds generated in one event have different outliers and a different perturbation by additive noise.

Outliers are randomly generated by a uniform distribution in a sphere with radius 2.0, when the bigger side of the normalized bounding box has size 1.0.

For additive noise generation, we perturb each point $\overrightarrow{p_i}$ adding a vector $\overrightarrow{r_i}$. The direction of $\overrightarrow{r_i}$ is generated by an isotropic distribution over a unit sphere, and its magnitude is obtained through a Gaussian random variable and scaled by a parameter δ, that controls the strength of the noise. This scheme of synthetic additive noise generation is not as usual as the addition of a perturbation on the normal direction of each point, but was chosen for this work because it yields more challenging scenarios. The general equation for additive noise is shown in Equation 4.1.

$$\overrightarrow{p_i} = \overrightarrow{p_i} + \delta \cdot \mathcal{N}(0,1) \cdot \overrightarrow{r_i} \tag{4.1}$$

We generate events with outliers in a quantity ω proportional to a percentage of the points in the cloud, having events with $\omega = 0\%$, 5% and 20%, and a variable perturbation factor δ for the additive noise (0, 0.01 and 0.05). Higher amounts of outliers and additive noise hardly represent realistic situations. For δ values bigger than 0.05, the additive noise degenerates the mesh such that most of its geometrical information is lost. Figure 4.2 illustrates the effect of additive noise and outliers.

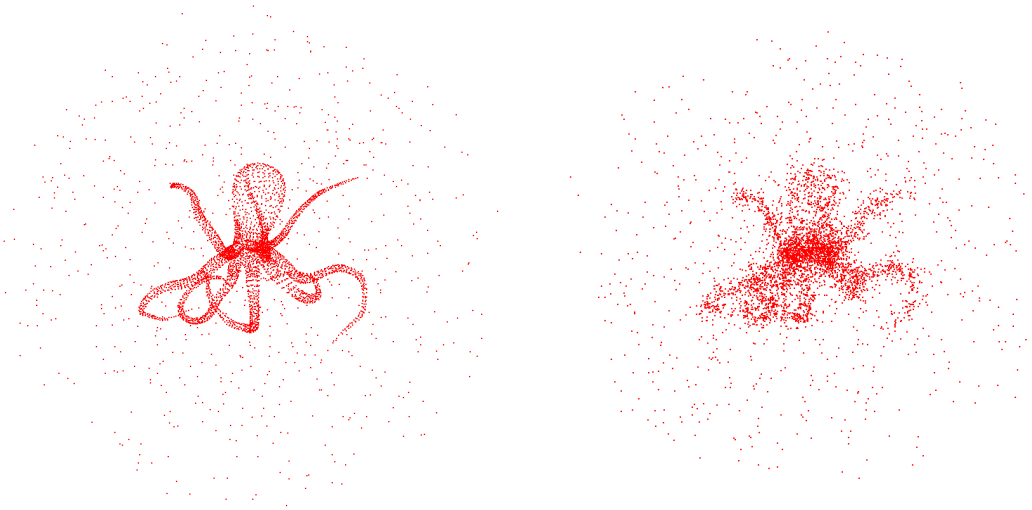


(a) Original Octopus.

(b) Octopus with additive noise ($\delta = 0.05$).

Figure 4.2: Examples of application of additive noise and outliers to the Bunny point set.

Figure 4.4 details the generation of synthetic events with outliers and additive noise.



(c) Octopus with $\omega = 20\%$ of Outliers.

(d) Octopus with additive noise and outliers combined ($\delta = 0.05$ and $\omega = 20\%$)

Figure 4.2: Examples of application of additive noise and outliers to the Bunny point set.

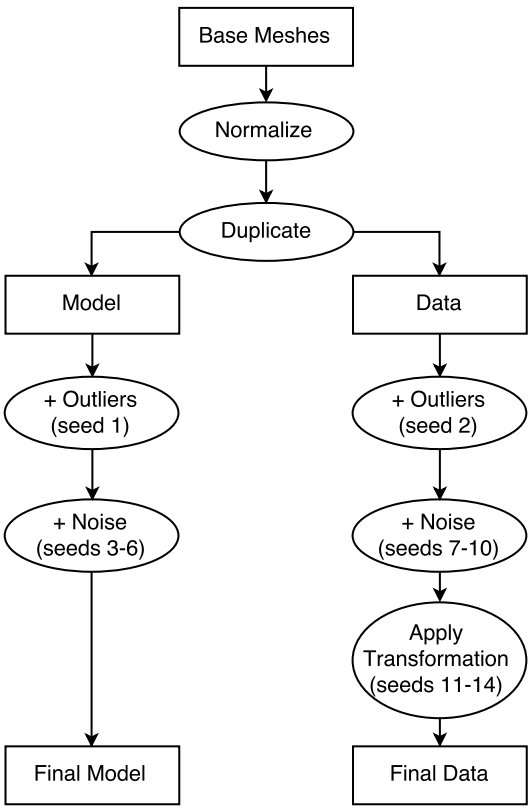


Figure 4.3: Pipeline of generation of events with noise and outliers.

4.1.2 PARTIAL OVERLAPPING

The second type of event addresses partial overlapping situations, simulating the process of object reconstruction. For that, we generate subsets of the base meshes that are continuous in relation to the nearest neighbors list, controlling the percentage amount of overlapping (β) and non-overlapping points (α) between those subsets, relative to the number of points on the base mesh. Table 4.1 shows the parameter combinations for α and β.

α	12.5%				25%			
β	75%	50%	25%	12.5%	50%	37.5%	25%	12.5%

Table 4.1: Variation of the parameters for the second type of event: percentage of points on the overlapping region (α) and on the non-overlapping region (β).

The number of non-overlapping points α is the same for both point clouds. We generate events with different levels of overlapping, varying the number of points in the shared region β. With higher values, both point clouds have more points in common. When both point clouds have $\alpha = 25\%$ of unique points that are not on the other cloud (non-overlapping), the maximum overlapping amount (β) would be 50%. As the rate between the overlapping points and the non-overlapping points of the meshes diminish, it is expected to be harder for the methods to align the point sets.

The first step for synthetic subset generation is a region growing algorithm based on a breadth search on the k-nearest-neighbors list, initiated on a random point to generate the overlapping region. In sequence, another breadth search is made on points on the boundary of the overlapping region for the generation of the non-overlapping regions of both point clouds, until the number of points desired is reached. To make sure the percentages are right, the number of points of the generated subsets is checked. In case it is wrong, the subsets are discarded and two new subsets is generated.

Figure 4.4 details the generation of synthetic events with partial overlapping.

Figure 4.5 shows examples of partially overlapped point clouds, with the variation of the parameters α and β.

Like in the first type of event, we also sample the rotation angle interval at each 15°.

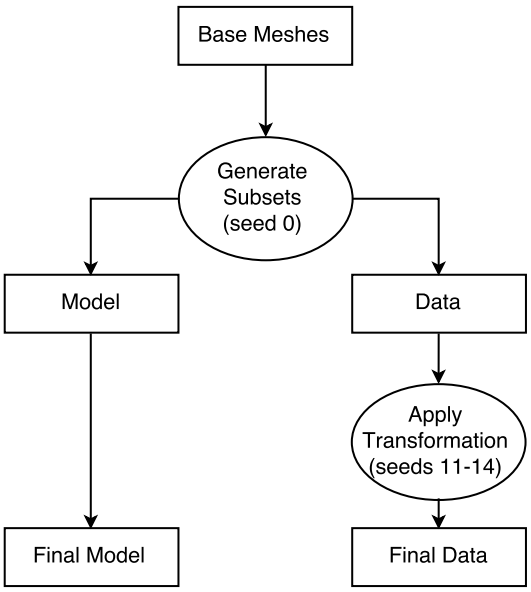
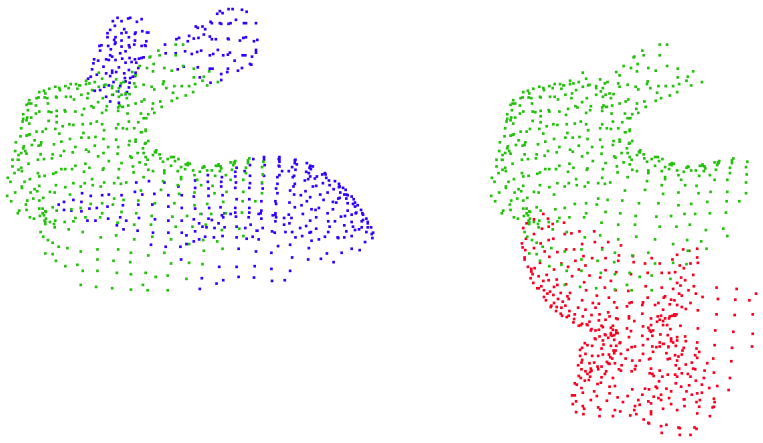
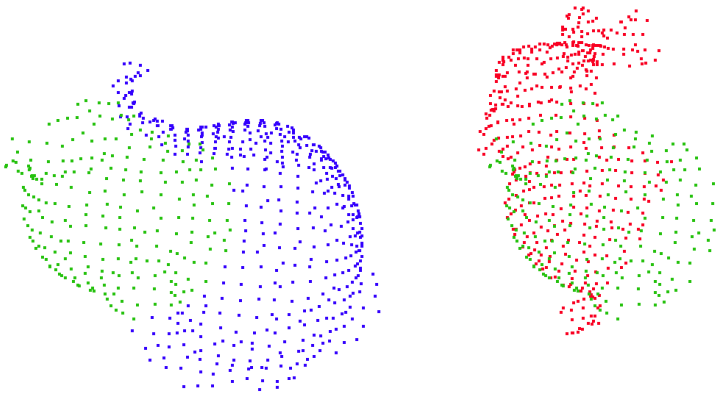


Figure 4.4: Pipeline of generation of events with partial overlapping.



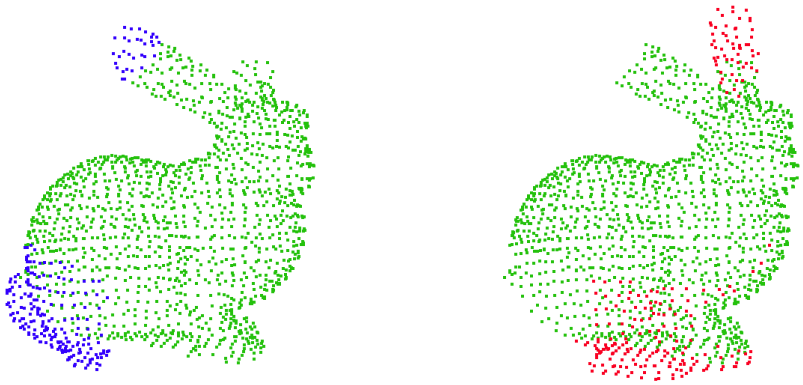
(a) (b)

Figure 4.5: Partial overlapping examples: green points indicate the overlapping region, and blue and red points indicate the non-overlapping region. (a) and (b): $\alpha = 25\%, \beta = 25\%$, average difficulty.



(c) (d)

Figure 4.5: (c) and (d): $\alpha = 12.5\%, \beta = 25\%$, the hardest parameter setup, since the overlapping region is smaller than the non-overlapping regions.



(e) (f)

Figure 4.5: (e) and (f): $\alpha = 12.5\%, \beta = 75\%$, the easiest parameter setup, with the largest proportion between the size of the overlapping region and the non-overlapping region.

4.2 ERROR MEASUREMENT PROTOCOL

In the fine registration literature, many error measurements are presented, such as residual errors, convergence regions and rotation and translation error. Since most works present qualitative results, visual results are shown in order to link these error measurements with the convergence of the methods on the chosen examples. Many works also present data on the number of iterations and the time spent on the execution, in order to highlight their computational efficiency.

In this work, we use two types of error measurements: the ground-truth root mean squared error (GT-RMS), which is the average Euclidean distance between the points on the first point cloud and their correct correspondences on the second, and the labeled error, which is the number of correct correspondences between both clouds.

The GT-RMS error is only measured between inliers, since outliers do not have correct correspondences. For the same reason, in partial overlapping events, only points on the overlapping region are considered. As a consequence, this error measurement can only be used with synthetically generated point clouds. It is important to emphasize the difference between the RMS used on the ICP, measured between the point and the correspondence estimated by the method, and the GT-RMS, between the point and its known correct correspondence. The GT-RMS is used only for error evaluation and cannot be used in the ICP, since it requires a ground-truth of previously known correspondences.

The labeled error can be used to evaluate cases of exact correspondences, since there is a guarantee that each point has a correspondent on the other mesh. In additive noise situations, the exactness of this measurement is lower, because the original position of the points is modified.

For the quantitative analysis, we have two clear result patterns - success and failure. If an algorithm fails to reach alignment on an event, it does not matter whether the final error obtained is big or not. Mean and standard deviation of the GT-RMS can be affected by the magnitude of the error in failure cases, and this fact can lead to wrong conclusions about the method performance. Therefore, the convergence rate can be better analyzed through a success/failure histogram. As a global success rate, we count the number of successful experiments through a threshold segmentation on the GT-RMS error and the labeled error, whose limit values are set based on previous qualitative observations and depend of the amount of additive noise and outliers.

For events with fully overlapped point clouds, the threshold criterion considered is simple on situations without additive noise: 95% of the correspondences must be correct, and the GT-RMS error is at maximum 10^{-2}. Outliers would not alter the GT-RMS criterion neither the labeled error, so the threshold criterion stands the same.

As stated before, additive noise can affect the exactness of the error measurements. The number of correct labeled correspondences tends to decay even on successful cases, but it is still possible to distinguish success from failure by those thresholds. With additive noise, we define a minimal labeled error of 100 correct correspondences combined with a maximal RMS error of 0.1, so that it is highly unlikely that a failure event is not identified by those threshold values. However, with high levels of additive noise, this threshold segmentation becomes harder, and some successful events might be classified as a failure. Table 4.2 shows the threshold values considered on our evaluation.

Situation	**Max GT-RMS**	**Min labeled**
Point clouds without additive noise	10^{-2}	$N_p \cdot 95\%$
Point clouds with additive noise	10^{-1}	100 points

Table 4.2: Threshold values used for events with additive noise and outliers

On the partial overlapping events, both labeled error and RMS error were calculated considering only the ground-truth correspondences of the overlapping region. We consider an event successful when its GT-RMS error is lower than a threshold of 0.05 and it has more than 90% of the points on the overlapping region with correct labeled correspondences. The list of ground-truth correspondences is computed on the subset generation algorithm.

Measurements of number of iterations were also taken from each method execution. In Section 4.5, we discuss the variation on the number of iterations in function of the weight step b of the ICP-CTSF.

4.3 RESULTS WITH ADDITIVE NOISE AND OUTLIERS

4.3.1 SETUP DETAILS

To characterize the behavior of our method with different parameter values, we separate 15 events of the first type for each value of angle, additive noise and outliers, performing a

preliminary step with 36 combinations of method parameters, i.e. the size of the nearest neighbor list (k), the ellipsoid angle α_{ellip} and the cut angle ϕ_{max}. The parameter b is set as 0.1 in this step, since this parameter is independent from the others, and its behavior is known: bigger values reaches better results performing more iterations. Thus, we use a value that is a midterm between performance and time spent.

In this preliminary step, the correspondences are always established using the CTSF-based matching and the transformation is estimated using Horn quaternion method (HORN, 1987). The difference between the methods is in the information encoded on tensors that are used as input for the ICP.

Table 4.3 shows the parameter configurations evaluated on the preliminary step.

k	α_{ellip}	ϕ_{max}
1%	30°	36°
	30°	45°
	45°	45°
	30°	60°
	45°	60°
	60°	60°
5%	30°	36°
	30°	45°
	45°	45°
	30°	60°
	45°	60°
	60°	60°
25%	30°	36°
	30°	45°
	45°	45°
	30°	60°
	45°	60°
	60°	60°
50%	30°	36°
	30°	45°
	45°	45°
	30°	60°
	45°	60°
	60°	60°
75%	30°	36°
	30°	45°
	45°	45°
	30°	60°
	45°	60°
	60°	60°
100%	30°	36°
	30°	45°
	45°	45°
	30°	60°
	45°	60°
	60°	60°

Table 4.3: Combination of parameters k, α_{ellip} and ϕ_{max}.

Thereupon, we use the parameter values that presented better results and execute more 30 events to compare the behavior of our method with other approaches. We raise the value of the parameter b to 0.75 in order to achieve better results, since the amount of methods to be executed is lower. In this step, we execute the original ICP and the Sparse ICP (BOUAZIZ et al., 2013), with both matching functions based on CTSF and Euclidean distance.

The first step of the experiment is not executed with the Sparse ICP because it would yield a high computational cost. Yet, since all the parameters evaluated affect only the tensor estimation, the correspondences provided by the CTSF are the same. Thus, parameters that yield better results on the Original ICP with the CTSF should also reach better results on other methods using the CTSF.

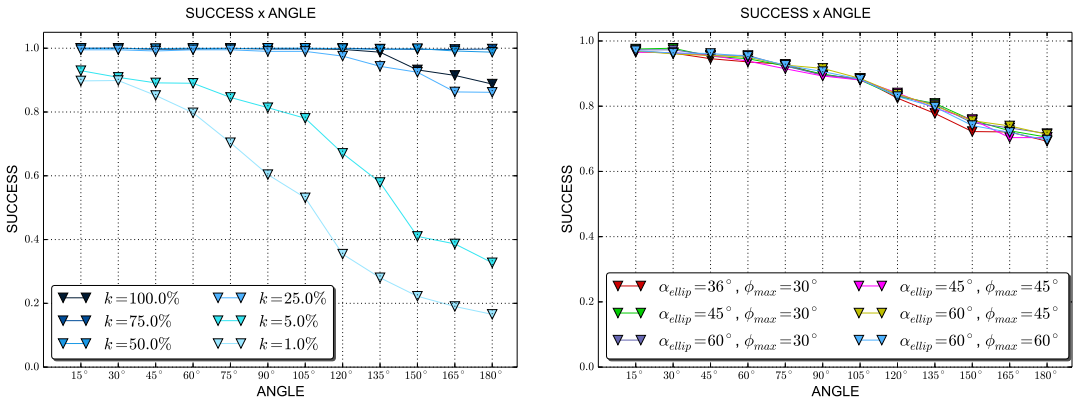
For the Sparse ICP, we use the norm parameter $p = 0.4$, which offers a good trade-off between performance and robustness, according to its authors. The error sequence of the Sparse ICP, unlike the original ICP, is not monotonically descent, so the standard method stops when a fixed number of ICP iterations is made, whose default value is 100, or when the error reaches below a stopping threshold. In order to adapt the CTSF weighting scheme to the Sparse ICP method, for each weight value we execute 100 iterations of the method or until the error is smaller than the threshold parameter. The decay of the weight parameter is the same as described in Algorithm 1. The stopping threshold error parameter is set to 10^{-5}. The other parameters are the default parameters of the source code made available by the authors[5].

4.3.2 FIRST STEP - BEST PARAMETER SEARCH

In the first step of the experimental setup for outliers and additive noise we highlight the parameter combination with the best overall performance for each point cloud, and some of the observed tendencies. We perform 15 experiments for each configuration, although ideally more results should be generated in order to obtain full statistical backup.

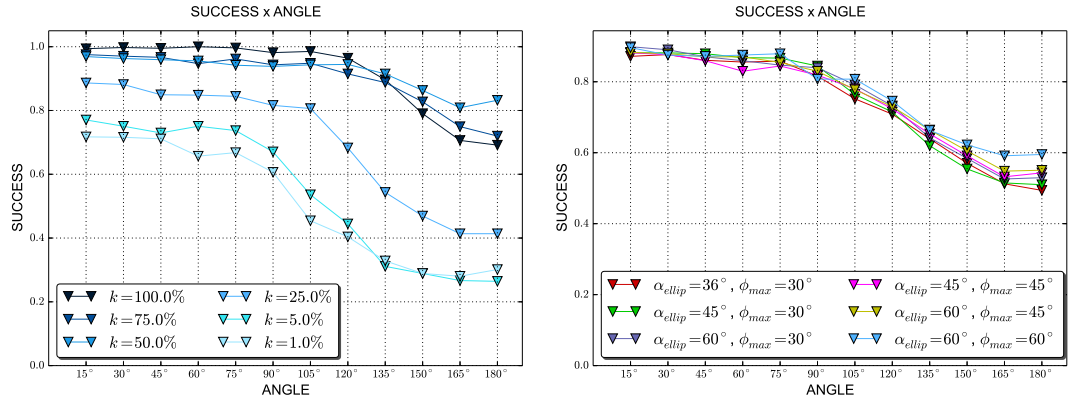
The results are grouped by the presence/absence of outliers and additive noise. The last row is the overall success rate of the method. Since the experiments are paired, the same conditions were tested for all the methods. Tables 4.4, 4.5 and 4.6 presents the success rate for each parameter configuration on each point cloud, and Figures 4.6, 4.7, 4.8 and 4.9 show the average success rate per angle, grouping by neighborhood size and angle parameter combinations for each point cloud. In each table, are highlighted the best result for each group of events and the best overall result.

[5]Available at `https://code.google.com/p/sparseicp/`



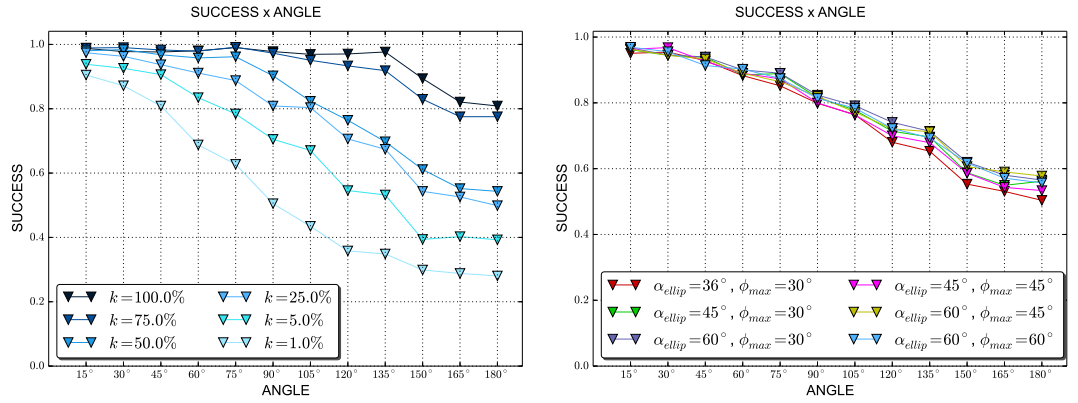
(a) Avg. success rate grouped by k (b) Avg. success rate by α_{ellip} and ϕ_{max}

Figure 4.6: Overall success by k and by angle combination for the Bunny.



(a) Avg. success rate grouped by k (b) Avg. success rate by α_{ellip} and ϕ_{max}

Figure 4.7: Overall success by k and by angle combination for the Happy Budha.



(a) Avg. success rate grouped by k (b) Avg. success rate by α_{ellip} and ϕ_{max}

Figure 4.8: Overall success by k and by angle combination for the Octopus.

$k = 100.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	96.94%	**100.00%**	91.94%	95.74%
45°	30°	**100.00%**	97.22%	**100.00%**	94.31%	96.85%
45°	45°	**100.00%**	99.17%	**100.00%**	96.39%	98.21%
60°	30°	**100.00%**	97.22%	**100.00%**	95.83%	97.53%
60°	45°	**100.00%**	**100.00%**	**100.00%**	97.78%	99.01%
60°	60°	**100.00%**	99.44%	**100.00%**	95.97%	98.09%
$k = 75.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	**100.00%**	**100.00%**	98.75%	99.44%
45°	30°	**100.00%**	99.44%	**100.00%**	99.72%	99.75%
45°	45°	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
60°	30°	**100.00%**	**100.00%**	**100.00%**	99.58%	99.81%
60°	45°	**100.00%**	**100.00%**	**100.00%**	99.86%	99.94%
60°	60°	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
$k = 50.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	**100.00%**	**100.00%**	99.17%	99.63%
45°	30°	**100.00%**	**100.00%**	**100.00%**	99.86%	99.94%
45°	45°	**100.00%**	**100.00%**	**100.00%**	99.03%	99.57%
60°	30°	**100.00%**	**100.00%**	**100.00%**	99.44%	99.75%
60°	45°	**100.00%**	**100.00%**	**100.00%**	99.72%	99.88%
60°	60°	**100.00%**	**100.00%**	**100.00%**	99.31%	99.69%
$k = 25.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	99.72%	96.67%	90.00%	94.75%
45°	30°	**100.00%**	**100.00%**	98.33%	95.28%	97.53%
45°	45°	**100.00%**	99.44%	97.78%	90.00%	94.94%
60°	30°	**100.00%**	**100.00%**	99.17%	96.94%	98.46%
60°	45°	**100.00%**	99.72%	98.33%	91.11%	95.62%
60°	60°	**100.00%**	98.89%	96.94%	90.14%	94.69%
$k = 5.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	88.61%	66.39%	56.25%	70.56%
45°	30°	**100.00%**	87.22%	65.83%	56.67%	70.31%
45°	45°	**100.00%**	90.28%	63.33%	55.28%	69.81%
60°	30°	**100.00%**	85.83%	67.22%	56.39%	70.19%
60°	45°	**100.00%**	88.89%	69.44%	55.97%	71.17%
60°	60°	**100.00%**	90.28%	66.11%	53.33%	69.57%
$k = 1.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	75.83%	40.56%	35.69%	52.84%
45°	30°	**100.00%**	78.61%	46.39%	35.28%	54.57%
45°	45°	99.44%	76.94%	41.94%	35.14%	53.09%
60°	30°	**100.00%**	76.39%	47.50%	34.86%	54.14%
60°	45°	**100.00%**	77.50%	45.56%	36.94%	54.88%
60°	60°	**100.00%**	79.44%	48.61%	35.42%	55.31%

Table 4.4: Success on outliers and noise situations - Bunny. Darker cells indicate higher convergence rates.

$k = 100.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	92.78%	99.72%	82.22%	90.43%
45°	30°	**100.00%**	86.11%	**100.00%**	77.08%	86.73%
45°	45°	**100.00%**	97.22%	97.22%	90.56%	94.57%
60°	30°	**100.00%**	88.61%	**100.00%**	77.92%	87.65%
60°	45°	**100.00%**	95.28%	99.17%	88.47%	93.64%
60°	60°	**100.00%**	99.17%	99.44%	**93.61%**	**96.85%**
$k = 75.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	85.83%	99.72%	85.56%	90.37%
45°	30°	**100.00%**	90.83%	98.89%	86.53%	91.73%
45°	45°	**100.00%**	84.72%	96.11%	77.64%	85.80%
60°	30°	**100.00%**	88.06%	99.72%	83.75%	90.06%
60°	45°	**100.00%**	97.22%	94.44%	84.17%	91.11%
60°	60°	**100.00%**	94.72%	99.44%	83.75%	91.48%
$k = 50.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	95.83%	96.94%	78.33%	88.77%
45°	30°	**100.00%**	96.94%	99.17%	82.22%	91.23%
45°	45°	**100.00%**	99.17%	98.89%	84.58%	92.72%
60°	30°	**100.00%**	98.06%	**100.00%**	86.67%	93.64%
60°	45°	**100.00%**	**100.00%**	96.39%	81.53%	90.99%
60°	60°	**100.00%**	**100.00%**	98.61%	87.92%	94.32%
$k = 25.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	85.83%	70.28%	50.69%	68.33%
45°	30°	**100.00%**	88.89%	75.56%	54.17%	71.73%
45°	45°	**100.00%**	89.44%	73.61%	46.81%	68.15%
60°	30°	**100.00%**	87.78%	77.50%	57.08%	73.21%
60°	45°	**100.00%**	93.33%	77.50%	50.97%	71.73%
60°	60°	**100.00%**	86.11%	76.94%	50.00%	69.57%
$k = 5.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	80.56%	45.00%	31.81%	53.15%
45°	30°	**100.00%**	84.17%	44.72%	29.58%	52.90%
45°	45°	**100.00%**	87.50%	45.83%	30.28%	54.20%
60°	30°	**100.00%**	85.28%	48.89%	33.61%	55.86%
60°	45°	**100.00%**	88.89%	44.72%	30.69%	54.44%
60°	60°	**100.00%**	90.56%	45.28%	31.81%	55.43%
$k = 1.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	90.00%	31.39%	26.11%	49.69%
45°	30°	**100.00%**	86.67%	38.06%	26.53%	50.62%
45°	45°	**100.00%**	92.78%	33.61%	27.08%	51.23%
60°	30°	**100.00%**	86.67%	35.00%	26.81%	50.06%
60°	45°	**100.00%**	93.06%	36.11%	25.56%	51.17%
60°	60°	**100.00%**	95.28%	38.89%	29.03%	53.83%

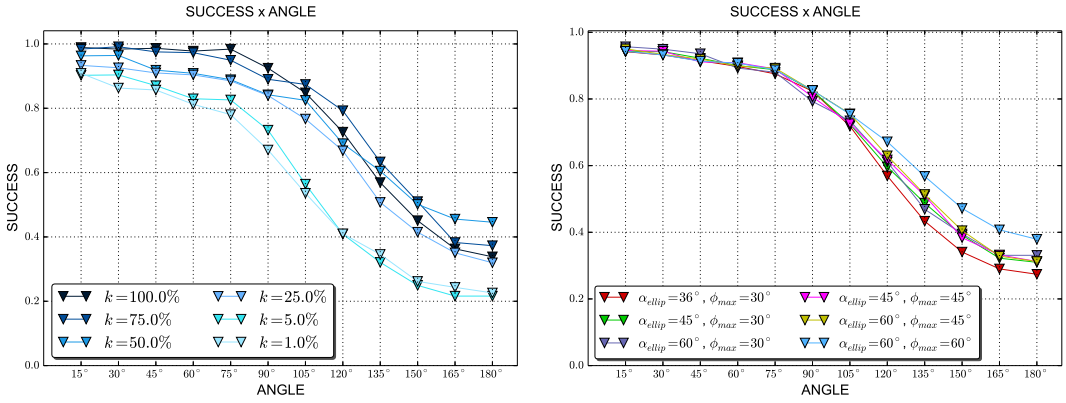
Table 4.5: Success on outliers and noise situations - Happy. Darker cells indicate higher convergence rates.

$k = 100.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	91.11%	**100.00%**	92.36%	94.63%
45°	30°	**100.00%**	86.94%	**100.00%**	92.78%	93.89%
45°	45°	**100.00%**	88.89%	**100.00%**	92.64%	94.26%
60°	30°	**100.00%**	90.28%	**100.00%**	92.50%	94.51%
60°	45°	**100.00%**	90.56%	**100.00%**	**93.33%**	**94.94%**
60°	60°	**100.00%**	91.39%	**100.00%**	91.39%	94.26%
$k = 75.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	94.44%	98.33%	84.86%	91.67%
45°	30°	**100.00%**	93.33%	99.17%	89.44%	93.64%
45°	45°	**100.00%**	92.50%	95.56%	83.61%	90.06%
60°	30°	**100.00%**	92.78%	98.89%	90.56%	93.95%
60°	45°	**100.00%**	94.17%	99.44%	90.00%	94.14%
60°	60°	**100.00%**	93.06%	96.94%	84.58%	90.93%
$k = 50.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	83.06%	83.61%	64.72%	76.91%
45°	30°	**100.00%**	89.72%	85.56%	71.11%	81.67%
45°	45°	**100.00%**	91.39%	83.33%	67.50%	79.94%
60°	30°	**100.00%**	90.83%	88.89%	76.39%	85.00%
60°	45°	**100.00%**	90.56%	83.61%	71.39%	81.54%
60°	60°	**100.00%**	93.33%	83.61%	71.53%	82.22%
$k = 25.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	85.56%	74.44%	61.11%	73.83%
45°	30°	**100.00%**	87.50%	81.39%	65.56%	77.78%
45°	45°	**100.00%**	85.56%	75.56%	61.67%	74.32%
60°	30°	**100.00%**	90.83%	83.06%	68.89%	80.37%
60°	45°	**100.00%**	91.11%	79.44%	64.86%	77.84%
60°	60°	**100.00%**	90.83%	77.22%	65.42%	77.53%
$k = 5.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	87.78%	58.33%	46.53%	64.26%
45°	30°	**100.00%**	88.33%	62.22%	49.58%	66.60%
45°	45°	**100.00%**	91.94%	60.83%	50.97%	67.72%
60°	30°	**100.00%**	90.56%	62.22%	49.44%	67.04%
60°	45°	**100.00%**	93.33%	58.61%	51.25%	67.65%
60°	60°	**100.00%**	93.61%	58.89%	52.36%	68.27%
$k = 1.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	82.50%	36.94%	30.69%	51.30%
45°	30°	**100.00%**	85.28%	38.06%	31.39%	52.47%
45°	45°	**100.00%**	91.94%	39.44%	32.92%	54.94%
60°	30°	**100.00%**	84.72%	37.50%	33.06%	52.96%
60°	45°	**100.00%**	94.72%	37.50%	29.44%	53.58%
60°	60°	**100.00%**	**95.28%**	39.72%	32.27%	55.44%

Table 4.6: Success on outliers and noise situations - Octopus. Darker cells indicate higher convergence rates.

$k = 100.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	79.44%	74.72%	63.61%	73.64%
45°	30°	**100.00%**	80.56%	75.56%	64.86%	74.63%
45°	45°	**100.00%**	79.72%	81.94%	63.75%	75.37%
60°	30°	**100.00%**	75.28%	80.28%	61.94%	73.21%
60°	45°	**100.00%**	84.44%	82.22%	65.14%	77.10%
60°	60°	**100.00%**	91.67%	**86.11%**	**73.06%**	**83.09%**
$k = 75.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	85.28%	73.89%	68.06%	76.73%
45°	30°	**100.00%**	90.56%	83.33%	70.97%	81.30%
45°	45°	**100.00%**	85.83%	73.89%	64.03%	75.06%
60°	30°	**100.00%**	93.06%	83.89%	69.58%	81.36%
60°	45°	**100.00%**	85.56%	69.44%	63.61%	73.83%
60°	60°	**100.00%**	89.44%	79.72%	66.25%	78.15%
$k = 50.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	77.78%	77.50%	56.11%	70.56%
45°	30°	**100.00%**	81.39%	80.56%	58.19%	72.96%
45°	45°	**100.00%**	85.00%	83.06%	61.67%	75.86%
60°	30°	**100.00%**	86.11%	79.44%	65.56%	77.04%
60°	45°	**100.00%**	82.22%	81.39%	62.64%	75.31%
60°	60°	**100.00%**	91.11%	82.50%	65.14%	78.64%
$k = 25.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	80.28%	57.50%	50.42%	64.14%
45°	30°	**100.00%**	87.22%	60.83%	52.92%	67.53%
45°	45°	**100.00%**	93.33%	67.50%	58.75%	72.96%
60°	30°	**100.00%**	86.94%	62.50%	53.61%	68.15%
60°	45°	**100.00%**	**96.39%**	68.89%	57.78%	73.52%
60°	60°	**100.00%**	95.56%	72.22%	59.72%	74.94%
$k = 5.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	80.00%	49.44%	41.67%	58.40%
45°	30°	**100.00%**	79.44%	49.72%	39.72%	57.47%
45°	45°	**100.00%**	78.33%	49.72%	42.50%	58.46%
60°	30°	**100.00%**	83.33%	45.83%	41.25%	58.15%
60°	45°	**100.00%**	81.39%	49.72%	45.00%	60.25%
60°	60°	**100.00%**	81.11%	50.28%	42.64%	59.26%
$k = 1.00\%$						
α_{ellip}	ϕ_{max}	Clean	Outliers	Noise	Noise + outliers	Overall
36°	30°	**100.00%**	83.89%	41.39%	40.83%	57.10%
45°	30°	**100.00%**	81.11%	45.28%	43.75%	58.64%
45°	45°	**100.00%**	85.56%	43.61%	37.78%	56.60%
60°	30°	**100.00%**	77.22%	41.94%	41.53%	56.05%
60°	45°	**100.00%**	86.39%	45.28%	40.28%	58.27%
60°	60°	**100.00%**	87.22%	47.50%	40.42%	59.01%

Table 4.7: Success on outliers and noise situations - Genus. Darker cells indicate higher convergence rates.



(a) Avg. success rate grouped by k (b) Avg. success rate by α_{ellip} and ϕ_{max}

Figure 4.9: Overall success by k and by angle combination for the Genus.

In general, for all point clouds, the results in cases with outliers were consistent, independently of the value of k. The only point cloud in which a substantial decay can be observed is the Bunny, in which for $k = 25\%$, 50%, 75% and 100% the convergence rates were near 100% for almost all cases, and this was not observed for $k = 1\%$ and 5%. However, in the other point clouds there was no tendency observed. On the Octopus, the best individual result for outliers was with $k = 1\%$, for instance. The invariance of the convergence rate in relation to the value of k is explained by the robustness of the tensor estimation method with respect to outliers. The coplanar structuring element is enough to generate different tensors for inliers and outliers, such that outliers do not compromise the quality of the correspondences of the method.

For additive noise, however, another tendency was observed, and higher values of k tend to reach better results. Additive noise affects directly the structuring of the points on the surface. Therefore, tensors are likely to lose their planarity with the presence of additive noise and the precision of the estimation becomes lower. With more neighbors, even if planarity information is lost, the tensors still encode roughly the geometric disposition of the neighborhood, and this can be effectively used for enhancement of the correspondences. Naturally, stronger levels of additive noise will result in lower convergence rates, since the structure of the surface is heavily damaged. We also observe a threshold on the value of k, different for each point cloud, above which all the values tested presented similar results: $k = 25\%$ on the Bunny, $k = 50\%$ on the Happy Budha and the Genus (which obtained lower convergence rates), and $k = 75\%$ on the Octopus.

When combining additive noise and outliers, the behavior observed with only additive noise was dominant, and bigger values of k reached better results. Thus, the use of bigger values of k is recommended if additive noise is present in the point cloud.

The angle parameters α_{ellip} and ϕ_{max} have only a slight influence on the results, such that there is no dominant parameter combination. As discussed in Subsections 2.2.4.3 and 2.2.4.4, higher values of ϕ_{max} yield smoother surfaces, while smaller values allow more details at cost of robustness to outliers. The parameter α_{ellip} controls the coplanar influence constraint, in which smaller values give more influence to coplanar points. A slight trend is observed for cases with outliers, in which methods with $\alpha_{ellip} = 60^\circ$ perform better in general.

The graphics on Figures 4.6, 4.7, 4.8 and 4.9 show that in fact the method is more

sensitive to variations on the parameter k, especially for higher angle displacements.

For the Bunny, the graphics show that the overall performance of the methods with $k \geq 25\%$ is above 90% for all α_{ellip} and ϕ_{max} values. As stated on Section 4.1, the Bunny is mostly composed of low curvature regions, but has the feet and the ears with high curvature. Due to absence of symmetry, the correspondences yielded by the CTSF were accurate, with a low amount of false-positives, and therefore it was an easier point cloud, with higher convergence rates.

More specifically, $k = 75\%$ obtained better results, above 99% for all parameter combinations. For $k = 100\%$, the convergence rate for outliers was slightly lower. The overall performance of small values of k was negatively impacted by additive noise.

The chosen parameter combination for the second step was $k = 75\%$, $\alpha_{ellip} = 60°$, $\phi_{max} = 60°$, that yielded success in 100% of the events. However, the choice could either be for $k = 75\%$, $\alpha_{ellip} = 45°$, $\phi_{max} = 45°$, that also reached convergence in all cases.

On the Happy Budha point cloud, high convergence rates were reached for $k \geq 50\%$. Although the performance in cases with only outliers was not impacted, lower values of k had problems dealing with additive noise. For instance, $k = 1\%$, $\alpha_{ellip} = 60°$, $\phi_{max} = 60°$ converged in 95.28% of cases with only outliers, but the same parameter combination reaches 29.03% combining noise and outliers.

Overall, the best value of k for the Happy Budha was $k = 100\%$. If compared to the results of the Bunny, $k = 25\%$ presented lower convergence rates, which means that in order to correctly represent the geometry of the point cloud more points are needed on the Happy Budha. The graphics show that $k = 50\%$ obtained better results with higher angles, but $k = 75\%$ and $k = 100\%$ obtained better results for lower angles.

For the angles combination, $\alpha_{ellip} = 60°$ and $\phi_{max} = 60°$ presented a little higher convergence rate for higher angles, if compared to other parameter configurations. The parameter combination chosen was $k = 100\%$, $\alpha_{ellip} = 60°$, $\phi_{max} = 60°$, with an overall convergence rate of 96.85%.

On the Octopus, the differences of performance between the values of k were more distinguishable in cases with additive noise, in which bigger values of k reached better results. For cases with only outliers, the performance was similar between the different parameters. The graphics illustrate the levels of performance for each k value and the little impact of the parameters α_{ellip} and ϕ_{max} on the result. The combination chosen was

$k = 100\%$, $\alpha_{ellip} = 60°$, $\phi_{max} = 45°$, with overall performance of 94.94%.

The Genus was the hardest point cloud of the dataset. Although the method is capable to recover the alignment for all original cases, the performance was strongly affected by additive noise, impacting negatively the overall success rates. The graphics show a much higher decay for higher angles, and this is explained by the high number of local optima caused by the symmetry.

The performance differences between the different angle values for α_{ellip} and ϕ_{max} are higher. For additive noise, $\alpha_{ellip} = 60°$, $\phi_{max} = 60°$ was the better setup in smaller k's and in $k = 100\%$. Better results with outliers were reached for $k = 25\%$, with 96.39% of success rate in the best case, but the performance was counterbalanced by the rates with additive noise. The best overall success rate was 83.09%, with $k = 100\%$, $\alpha_{ellip} = 60°$, $\phi_{max} = 60°$. This combination of parameters performed particularly well with additive noise, reaching 86.11% in cases with only additive noise and 73.09% with both outliers and noise, smaller rates if compared to the results on the other point clouds.

4.3.3 SECOND STEP - BEST RESULT SEARCH

In this step, we used the best parameters obtained on the first step for more 30 executions on each point cloud. The methods evaluated were the Original ICP and the Sparse ICP, with matching functions based on the CTSF and Euclidean distance. The value of b was raised to 0.75, in order to enhance the results of the method. This also increases the execution time of the method, and the time spent on most executions of the Sparse ICP combined with the CTSF was very high, if compared to other methods.

For this step, we detail the success rate for each level of outlier and additive noise. The tables show that the success rates of the CTSF methods are effectively enhanced for high angles. This was expected, since such results are similar to the ones obtained on the first step for this parameter combination, and fine methods were not designed to cope with bad initialization situations. However, with small angles (up to 30°) the convergence rates are reasonable for the original methods with Euclidean distance-based matching. This way, the results obtained are coherent, and with the proper coarse initialization the methods should be able to reach better convergence rates.

The Sparse ICP with the CTSF obtained poor convergence rates particularly on the Genus point cloud, which was pointed out in the first step as the hardest point cloud of the dataset. The graphic shows that the Sparse ICP was affected by high angles, probably because of the number of a local optima easily reachable from angles bigger than 90°.

ICP	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	29.44%	29.44%	21.11%	26.67%
$\sigma = 0.01$	16.94%	21.94%	14.17%	17.69%
$\sigma = 0.05$	16.67%	18.61%	13.06%	16.11%
Overall	21.02%	23.33%	16.11%	20.15%
Sparse ICP	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	44.44%	45.56%	40.56%	43.52%
$\sigma = 0.01$	41.11%	40.28%	35.28%	38.89%
$\sigma = 0.05$	32.78%	33.61%	28.89%	31.76%
Overall	39.44%	39.81%	34.91%	38.06%
ICP-CTSF	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	100.00%	99.72%	100.00%	99.91%
$\sigma = 0.01$	100.00%	100.00%	100.00%	100.00%
$\sigma = 0.05$	100.00%	100.00%	100.00%	100.00%
Overall	100.00%	99.91%	100.00%	99.97%
Sparse ICP-CTSF	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	100.00%	95.00%	100.00%	98.33%
$\sigma = 0.01$	100.00%	100.00%	100.00%	100.00%
$\sigma = 0.05$	100.00%	100.00%	98.33%	99.44%
Overall	100.00%	98.33%	99.44%	99.26%

Table 4.8: Success per additive noise and outlier level, second step - Bunny. Darker cells indicate higher convergence rates.

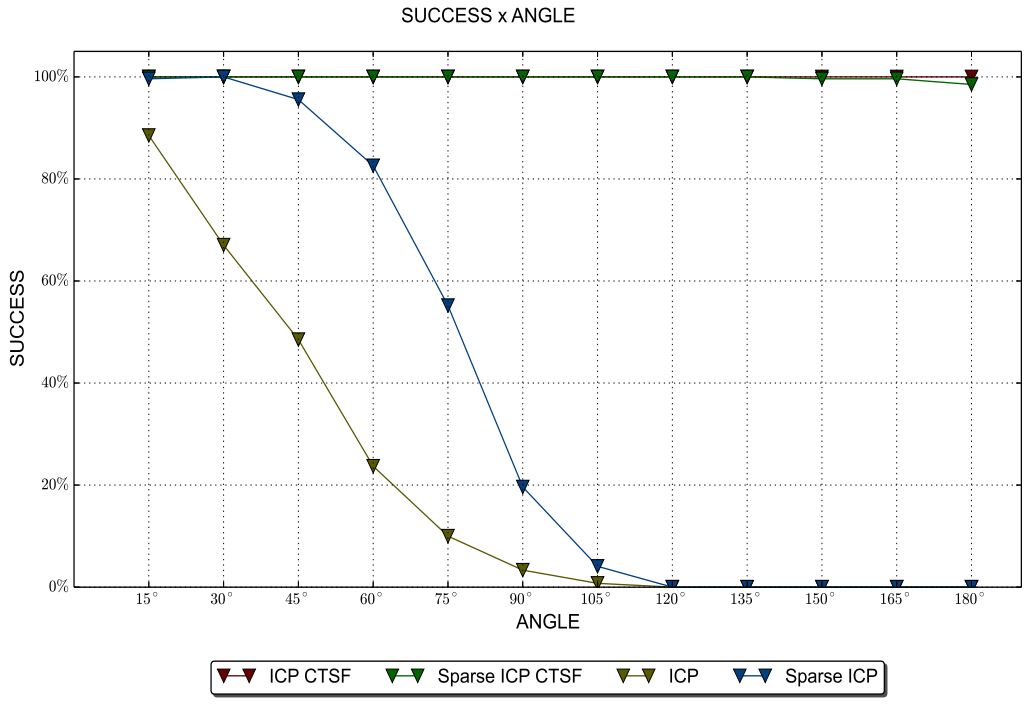


Figure 4.10: Overall success by method on the Bunny.

ICP	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	40.00%	40.56%	33.06%	37.87%
$\sigma = 0.01$	22.50%	30.00%	21.39%	24.63%
$\sigma = 0.05$	33.61%	36.39%	31.39%	33.80%
Overall	32.04%	35.65%	28.61%	32.10%
Sparse ICP	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	48.06%	49.17%	33.33%	43.52%
$\sigma = 0.01$	47.22%	47.22%	33.61%	42.69%
$\sigma = 0.05$	41.39%	43.06%	36.11%	40.19%
Overall	45.56%	46.48%	34.35%	42.13%
ICP-CTSF	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	100.00%	100.00%	100.00%	100.00%
$\sigma = 0.01$	100.00%	100.00%	95.28%	98.43%
$\sigma = 0.05$	99.44%	98.61%	92.50%	96.85%
Overall	99.81%	99.54%	95.93%	98.43%
Sparse ICP-CTSF	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	100.00%	86.94%	75.00%	87.31%
$\sigma = 0.01$	100.00%	98.89%	76.67%	91.85%
$\sigma = 0.05$	91.11%	87.22%	73.33%	83.89%
Overall	97.04%	91.02%	75.00%	87.69%

Table 4.9: Success per additive noise and outlier level - Happy Budha. Darker cells indicate higher convergence rates.

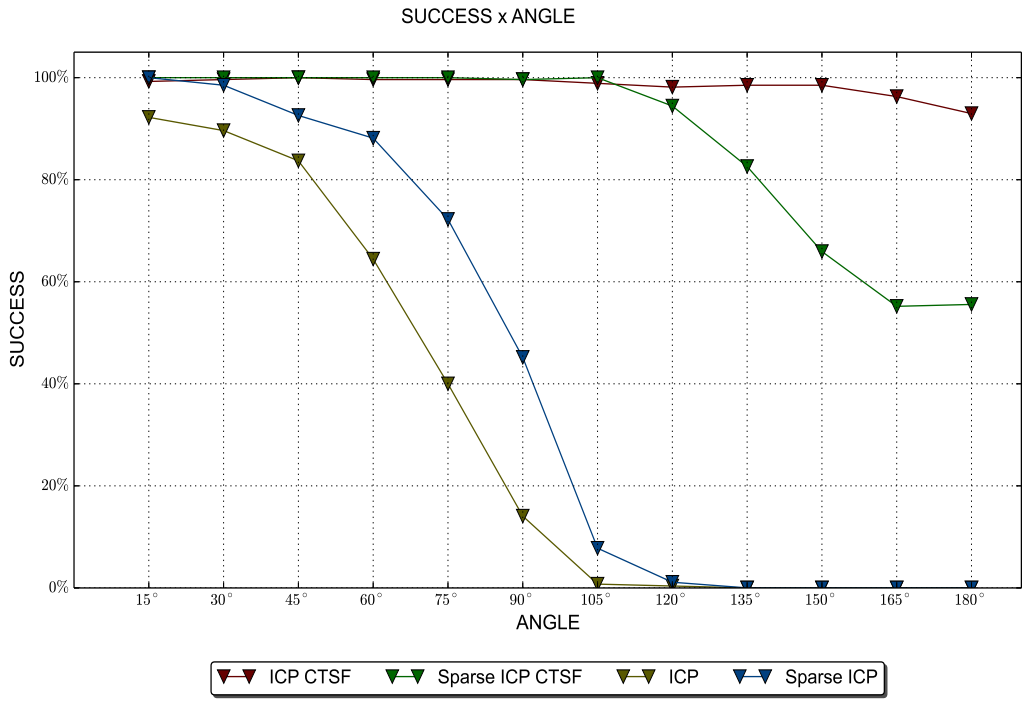


Figure 4.11: Overall success by method on the Happy Budha.

ICP	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	14.44%	16.94%	8.89%	13.43%
$\sigma = 0.01$	11.39%	14.72%	6.11%	10.74%
$\sigma = 0.05$	9.72%	14.17%	6.11%	10.00%
Overall	11.85%	15.28%	7.04%	11.39%
Sparse ICP	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	26.39%	28.33%	29.44%	28.06%
$\sigma = 0.01$	26.11%	23.06%	26.39%	25.19%
$\sigma = 0.05$	20.00%	24.44%	25.28%	23.24%
Overall	24.17%	25.28%	27.04%	25.49%
ICP CTSF	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	100.00%	100.00%	100.00%	100.00%
$\sigma = 0.01$	96.94%	100.00%	100.00%	98.98%
$\sigma = 0.05$	90.56%	98.32%	94.66%	94.50%
Overall	95.83%	99.44%	98.23%	97.83%
Sparse ICP CTSF	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	100.00%	99.17%	95.28%	98.15%
$\sigma = 0.01$	100.00%	98.61%	92.44%	97.03%
$\sigma = 0.05$	94.17%	91.94%	89.66%	91.93%
Overall	98.06%	96.57%	92.47%	95.70%

Table 4.10: Success per additive noise and outlier level - Octopus. Darker cells indicate higher convergence rates.

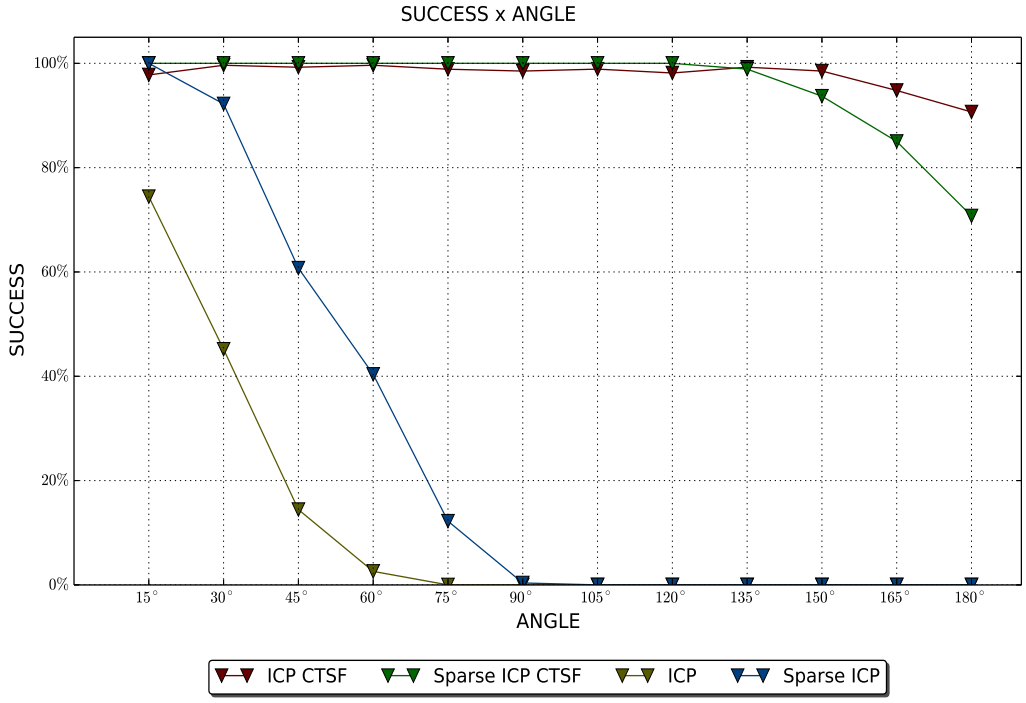


Figure 4.12: Overall success by method on the Octopus.

ICP	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	36.94%	36.11%	34.72%	35.93%
$\sigma = 0.01$	30.56%	32.50%	23.89%	28.98%
$\sigma = 0.05$	31.01%	29.72%	31.94%	30.89%
Overall	32.84%	32.78%	30.19%	31.93%
Sparse ICP	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	50.83%	51.94%	50.28%	51.02%
$\sigma = 0.01$	45.83%	48.06%	49.17%	47.69%
$\sigma = 0.05$	39.44%	39.44%	43.06%	40.65%
Overall	45.37%	46.48%	47.50%	46.45%
ICP CTSF	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	100.00%	99.17%	82.78%	93.98%
$\sigma = 0.01$	99.44%	95.83%	76.67%	90.65%
$\sigma = 0.05$	90.25%	81.11%	65.28%	78.87%
Overall	96.57%	92.04%	74.91%	87.84%
Sparse ICP CTSF	0.00%	5.00%	20.00%	**Overall**
$\sigma = 0.00$	99.17%	63.31%	56.39%	72.98%
$\sigma = 0.01$	71.39%	63.33%	58.31%	64.37%
$\sigma = 0.05$	65.27%	61.11%	55.83%	60.72%
Overall	78.64%	62.58%	56.84%	66.03%

Table 4.11: Success per additive noise and outlier level - Genus. Darker cells indicate higher convergence rates.

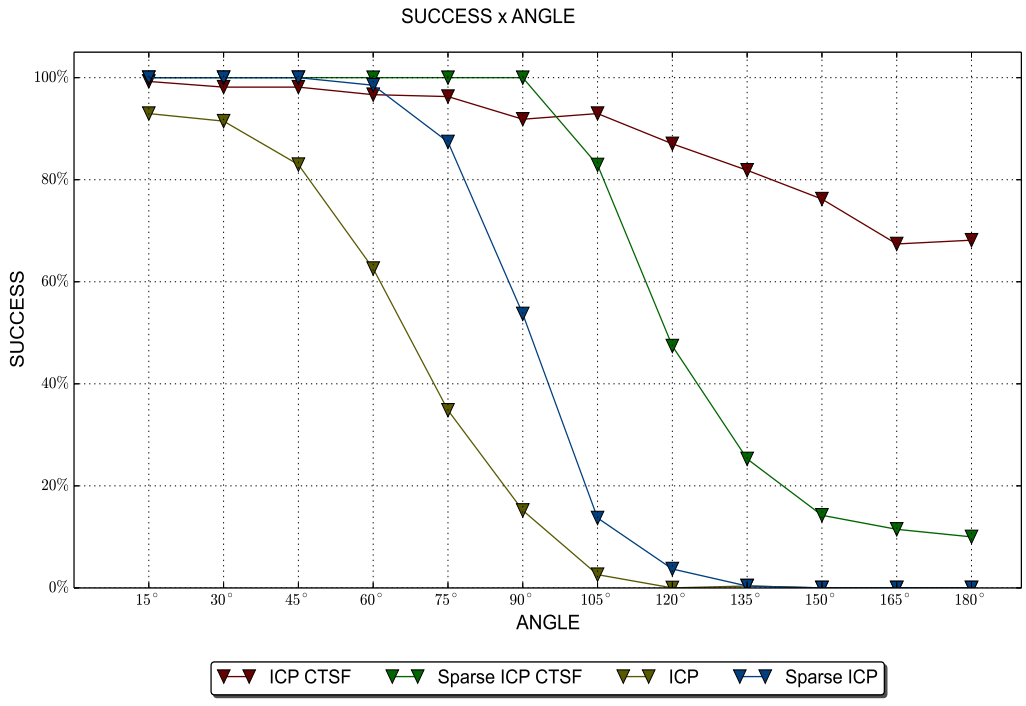


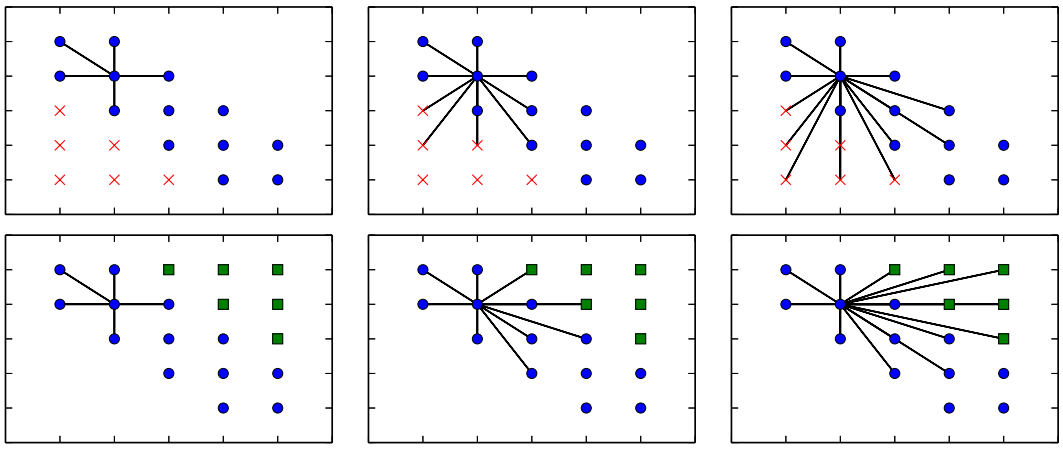
Figure 4.13: Overall success by method on the Genus.

4.4 RESULTS WITH PARTIAL OVERLAPPING

In this type of event, we evaluate 30 events for each parameter configuration of angle and overlapping/non-overlapping amount. The methods chosen were the Sparse ICP method, the original ICP, and its trimmed version, using match functions based on the Euclidean distance and on the CTSF. As stated before, it is known that the original ICP formulation is not proper for partially overlapped point clouds, therefore its performance tends to be poor.

In this case, we chose not to do the preliminary step and use the values of $\alpha_{ellip} = 45°$ and $\phi_{max} = 45°$, due to the high computational time that would be spent.

Specifically on partially overlapped point clouds, only correspondences between points over the common region are considered correct. The effectiveness of the CTSF is possibly lower in this case, because the neighborhood of the points on the overlapping region can include points outside of this region, generating different tensors for each point cloud. Figure 4.14 illustrates this situation.



(a) 5 neighbors (b) 10 neighbors (c) 15 neighbors

Figure 4.14: Example of nearest neighbors with partial overlapping point clouds. In this case, the blue points are inside the overlapping region, while red and green points are unique regions of each point cloud. Note that as the number of neighbor gets larger the neighborhoods of the two point clouds consider points outside the overlapping region, which yield different tensors.

It is fair to conclude that as the value of k increases, the difference between the regions represented by the tensors would increase, since points outside of the overlapping region have a bigger chance of being used in the computation of the tensor. Conversely, smaller

values of k have a tendency to provide similar tensors between correspondences on the overlapping region, since the number of different neighbors outside the overlapping region tends to be smaller and limited to a boundary of the region. Theoretically, those tensors can be identified more easily by the method. If the size of the neighborhood is too small, however, the information cast on the tensors can be insufficient to represent the local geometry. We test four values of k: 15%, 10%, 5% and 1% of the points.

The Trimmed ICP was included on the evaluation of this type of event since it is a classical strategy to deal with partial overlapping point sets. For this method, we fixed the amount of discarded points in 10% of the number of points of the mesh, although ideally the amount should be proportional to the overlapping percentage. In practical situations, the amount of overlapping between the point sets usually is unknown. Therefore, this value is set as an intermediate value.

The Tables 4.12, 4.13, 4.10, 4.15 show the percentual convergence rate per overlapping and non-overlapping amount, and the graphics show the general convergence per angle on the four meshes.

α	**12.5%**			
β	**75.0%**	**50.0%**	**25.0%**	**12.5%**
Trimmed ICP CTSF k = 15%	91.39%	95.00%	16.39%	0.00%
Trimmed ICP CTSF k = 10%	95.00%	95.83%	16.39%	0.00%
Trimmed ICP CTSF k = 5%	93.89%	89.44%	19.72%	0.28%
Trimmed ICP CTSF k = 1%	75.83%	78.06%	21.94%	0.28%
Trimmed ICP	18.61%	16.67%	1.67%	0.00%
ICP CTSF k = 15%	99.44%	38.61%	3.61%	0.00%
ICP CTSF k = 10%	99.72%	39.17%	4.44%	0.00%
ICP CTSF k = 5%	98.89%	38.61%	4.44%	0.00%
ICP CTSF k = 1%	91.39%	36.94%	4.72%	0.00%
ICP	25.28%	7.50%	0.56%	0.00%
Sparse ICP CTSF k = 15%	93.89%	94.72%	72.22%	26.11%
Sparse ICP CTSF k = 10%	95.56%	96.94%	73.33%	27.50%
Sparse ICP CTSF k = 5%	95.56%	93.33%	75.56%	33.89%
Sparse ICP CTSF k = 1%	93.33%	93.89%	71.31%	32.22%
Sparse ICP	41.67%	29.44%	13.65%	6.11%
α	**25.0%**			
β	**50.0%**	**37.5%**	**25.0%**	**12.5%**
Trimmed ICP CTSF k = 15%	45.83%	4.72%	0.28%	0.00%
Trimmed ICP CTSF k = 10%	50.28%	5.28%	0.28%	0.00%
Trimmed ICP CTSF k = 5%	50.00%	5.56%	0.00%	0.00%
Trimmed ICP CTSF k = 1%	44.44%	8.06%	0.00%	0.00%
Trimmed ICP	6.39%	1.39%	0.00%	0.00%
ICP CTSF k = 15%	8.33%	0.28%	0.00%	0.00%
ICP CTSF k = 10%	9.17%	0.28%	0.00%	0.00%
ICP CTSF k = 5%	8.61%	0.00%	0.00%	0.00%
ICP CTSF k = 1%	8.33%	0.56%	0.00%	0.00%
ICP	1.11%	0.00%	0.00%	0.00%
Sparse ICP CTSF k = 15%	84.12%	73.06%	50.70%	0.00%
Sparse ICP CTSF k = 10%	87.19%	80.78%	56.55%	0.00%
Sparse ICP CTSF k = 5%	89.69%	78.33%	55.00%	0.00%
Sparse ICP CTSF k = 1%	89.94%	75.21%	58.89%	0.00%
Sparse ICP	29.05%	17.83%	9.44%	0.00%

Table 4.12: Success per overlapping and nonoverlapping level - Bunny. Darker cells indicate higher convergence rates.

α	**12.5%**			
β	**75.0%**	**50.0%**	**25.0%**	**12.5%**
Trimmed ICP CTSF k = 15%	76.39%	80.00%	7.50%	0.00%
Trimmed ICP CTSF k = 10%	93.06%	85.00%	7.78%	0.00%
Trimmed ICP CTSF k = 5%	95.28%	88.33%	11.39%	0.00%
Trimmed ICP CTSF k = 1%	99.17%	93.33%	15.83%	0.00%
Trimmed ICP	31.11%	20.28%	1.39%	0.00%
ICP CTSF k = 15%	94.44%	10.83%	0.00%	0.00%
ICP CTSF k = 10%	95.56%	10.56%	0.28%	0.00%
ICP CTSF k = 5%	96.94%	10.83%	0.28%	0.00%
ICP CTSF k = 1%	96.94%	11.39%	0.56%	0.00%
ICP	36.39%	1.67%	0.00%	0.00%
Sparse ICP CTSF k = 15%	78.21%	82.45%	48.75%	1.39%
Sparse ICP CTSF k = 10%	89.89%	88.02%	53.20%	2.50%
Sparse ICP CTSF k = 5%	92.66%	91.34%	57.38%	6.94%
Sparse ICP CTSF k = 1%	94.25%	94.40%	55.99%	9.44%
Sparse ICP	46.57%	34.73%	10.86%	1.67%
α	**25.0%**			
β	**50.0%**	**37.5%**	**25.0%**	**12.5%**
Trimmed ICP CTSF k = 15%	6.67%	0.28%	0.00%	0.00%
Trimmed ICP CTSF k = 10%	6.39%	0.00%	0.00%	0.00%
Trimmed ICP CTSF k = 5%	8.33%	0.28%	0.00%	0.00%
Trimmed ICP CTSF k = 1%	9.72%	0.28%	0.00%	0.00%
Trimmed ICP	1.67%	0.00%	0.00%	0.00%
ICP CTSF k = 15%	0.00%	0.00%	0.00%	0.00%
ICP CTSF k = 10%	0.00%	0.00%	0.00%	0.00%
ICP CTSF k = 5%	0.00%	0.00%	0.00%	0.00%
ICP CTSF k = 1%	0.00%	0.00%	0.00%	0.00%
ICP	0.00%	0.00%	0.00%	0.00%
Sparse ICP CTSF k = 15%	73.45%	53.61%	21.39%	0.00%
Sparse ICP CTSF k = 10%	77.12%	56.67%	27.78%	0.00%
Sparse ICP CTSF k = 5%	89.17%	64.72%	32.22%	0.00%
Sparse ICP CTSF k = 1%	91.27%	71.94%	39.17%	0.00%
Sparse ICP	33.61%	20.28%	7.22%	0.00%

Table 4.13: Success per overlapping and nonoverlapping level - Happy Buddha. Darker cells indicate higher convergence rates.

α	**12.5%**			
β	**75.0%**	**50.0%**	**25.0%**	**12.5%**
Trimmed ICP CTSF k = 15%	81.11%	58.06%	2.22%	0.00%
Trimmed ICP CTSF k = 10%	87.78%	57.78%	2.22%	0.00%
Trimmed ICP CTSF k = 5%	90.28%	63.61%	2.50%	0.00%
Trimmed ICP CTSF k = 1%	92.78%	75.28%	3.89%	0.00%
Trimmed ICP	14.17%	8.33%	0.56%	0.00%
ICP CTSF k = 15%	5.28%	1.67%	0.00%	0.00%
ICP CTSF k = 10%	5.00%	2.22%	0.28%	0.00%
ICP CTSF k = 5%	9.44%	2.22%	0.00%	0.00%
ICP CTSF k = 1%	12.78%	1.94%	0.28%	0.00%
ICP	1.39%	0.00%	0.00%	0.00%
Sparse ICP CTSF k = 15%	98.33%	74.72%	25.83%	2.50%
Sparse ICP CTSF k = 10%	99.17%	72.50%	23.89%	3.06%
Sparse ICP CTSF k = 5%	99.44%	75.56%	31.67%	2.22%
Sparse ICP CTSF k = 1%	99.72%	79.17%	28.61%	3.06%
Sparse ICP	26.67%	23.33%	10.28%	0.83%
α	**25.0%**			
β	**50.0%**	**37.5%**	**25.0%**	**12.5%**
Trimmed ICP CTSF k = 15%	4.44%	0.00%	0.00%	0.00%
Trimmed ICP CTSF k = 10%	4.44%	0.00%	0.00%	0.00%
Trimmed ICP CTSF k = 5%	3.89%	0.00%	0.00%	0.00%
Trimmed ICP CTSF k = 1%	3.61%	0.00%	0.00%	0.00%
Trimmed ICP	0.28%	0.00%	0.00%	0.00%
ICP CTSF k = 15%	0.00%	0.00%	0.00%	0.00%
ICP CTSF k = 10%	0.00%	0.00%	0.00%	0.00%
ICP CTSF k = 5%	0.00%	0.00%	0.00%	0.00%
ICP CTSF k = 1%	0.00%	0.00%	0.00%	0.00%
ICP	0.00%	0.00%	0.00%	0.00%
Sparse ICP CTSF k = 15%	62.78%	42.50%	19.72%	0.00%
Sparse ICP CTSF k = 10%	73.61%	43.61%	21.94%	0.00%
Sparse ICP CTSF k = 5%	72.50%	49.44%	24.15%	0.00%
Sparse ICP CTSF k = 1%	83.33%	76.39%	36.54%	0.00%
Sparse ICP	20.56%	20.00%	10.83%	0.00%

Table 4.14: Success per overlapping and nonoverlapping level - Octopus. Darker cells indicate higher convergence rates.

α	**12.5%**			
β	**75.0%**	**50.0%**	**25.0%**	**12.5%**
Trimmed ICP CTSF k = 15%	77.22%	77.78%	15.56%	0.28%
Trimmed ICP CTSF k = 10%	82.50%	81.67%	18.33%	0.28%
Trimmed ICP CTSF k = 5%	83.06%	84.17%	22.50%	0.28%
Trimmed ICP CTSF k = 1%	83.89%	83.61%	25.56%	0.28%
Trimmed ICP	27.22%	19.17%	3.06%	0.00%
ICP CTSF k = 15%	42.22%	17.50%	1.11%	0.00%
ICP CTSF k = 10%	46.11%	19.44%	0.83%	0.00%
ICP CTSF k = 5%	51.39%	19.17%	1.11%	0.00%
ICP CTSF k = 1%	57.50%	18.89%	2.22%	0.00%
ICP	12.78%	4.17%	0.28%	0.00%
Sparse ICP CTSF k = 15%	75.83%	76.94%	52.50%	2.50%
Sparse ICP CTSF k = 10%	85.56%	83.61%	56.11%	3.06%
Sparse ICP CTSF k = 5%	90.28%	88.06%	61.11%	7.78%
Sparse ICP CTSF k = 1%	91.67%	83.89%	61.67%	13.33%
Sparse ICP	47.50%	38.06%	19.17%	3.89%
α	**25.0%**			
β	**50.0%**	**37.5%**	**25.0%**	**12.5%**
Trimmed ICP CTSF k = 15%	12.50%	0.28%	0.00%	0.00%
Trimmed ICP CTSF k = 10%	15.00%	0.00%	0.00%	0.00%
Trimmed ICP CTSF k = 5%	15.28%	0.00%	0.00%	0.00%
Trimmed ICP CTSF k = 1%	17.22%	0.00%	0.00%	0.00%
Trimmed ICP	5.28%	0.00%	0.00%	0.00%
ICP CTSF k = 15%	0.00%	0.00%	0.00%	0.00%
ICP CTSF k = 10%	0.28%	0.00%	0.00%	0.00%
ICP CTSF k = 5%	0.00%	0.00%	0.00%	0.00%
ICP CTSF k = 1%	0.00%	0.00%	0.00%	0.00%
ICP	0.28%	0.00%	0.00%	0.00%
Sparse ICP CTSF k = 15%	71.11%	66.94%	26.11%	0.00%
Sparse ICP CTSF k = 10%	79.72%	75.28%	34.72%	0.00%
Sparse ICP CTSF k = 5%	86.11%	83.89%	43.06%	0.00%
Sparse ICP CTSF k = 1%	88.89%	82.78%	53.33%	0.00%
Sparse ICP	38.06%	28.89%	11.67%	0.00%

Table 4.15: Success per overlapping and nonoverlapping level - Genus. Darker cells indicate higher convergence rates.

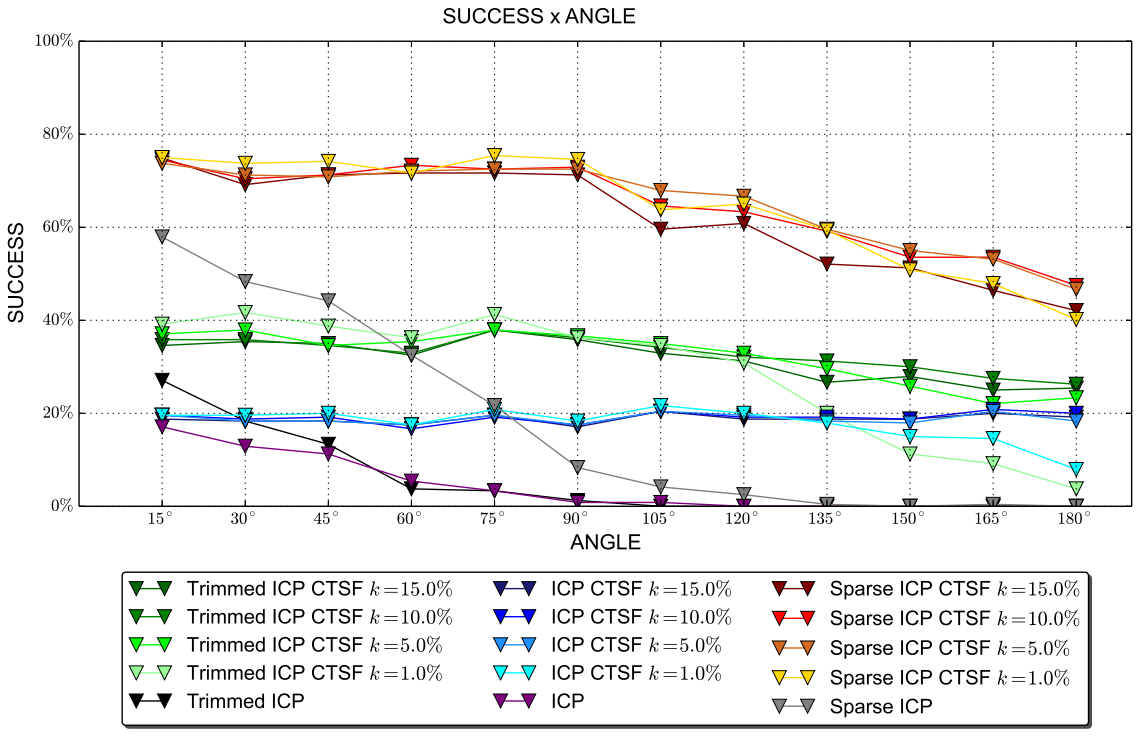


Figure 4.15: Convergence per angle on the Bunny, with all partial overlapping levels.

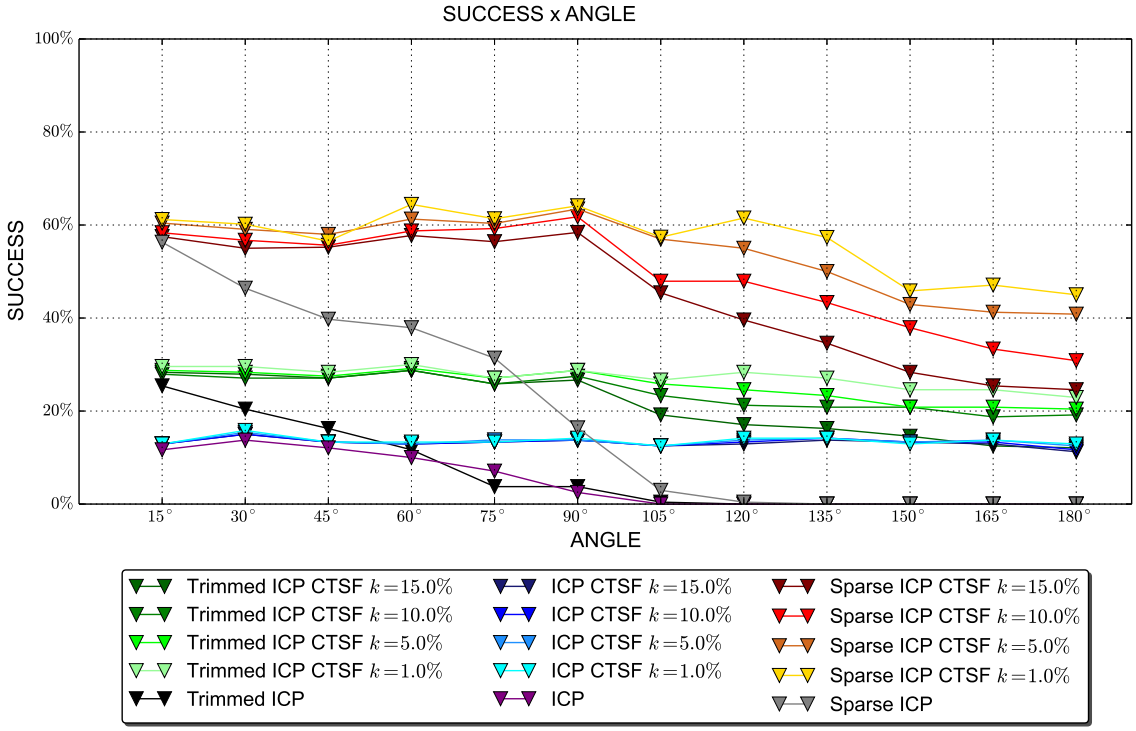


Figure 4.16: Convergence per angle on the Happy Budha, with all partial overlapping levels.

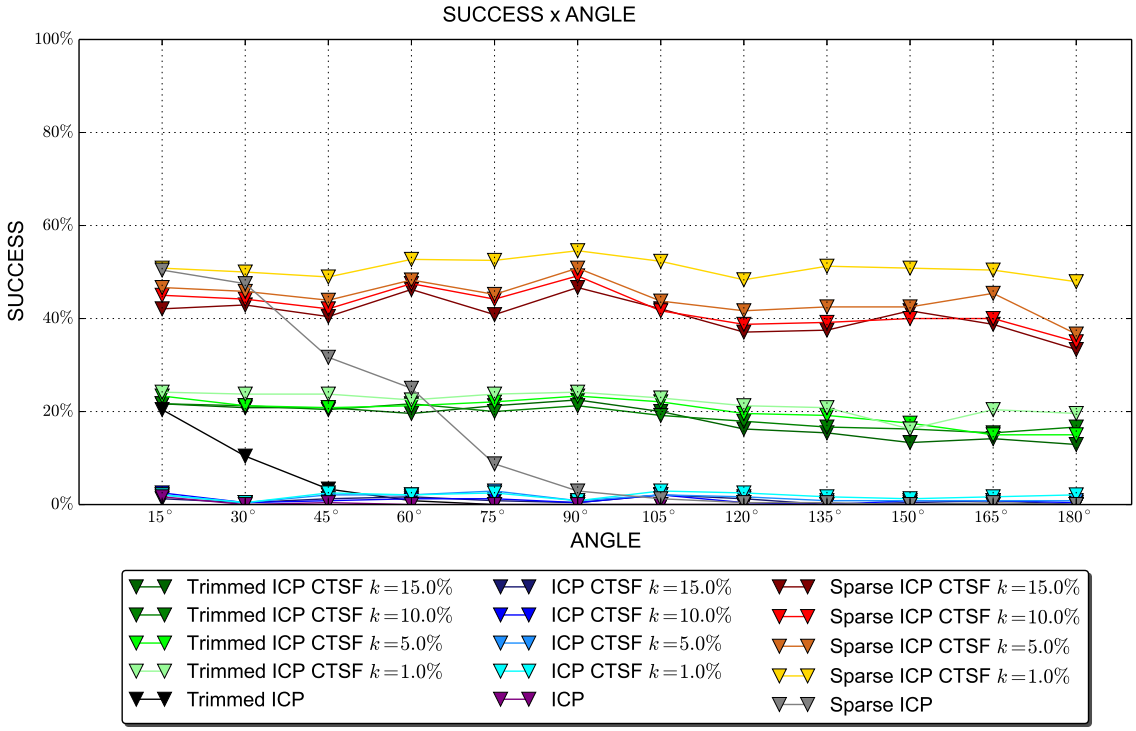


Figure 4.17: Convergence per angle on the Octopus, with all partial overlapping levels.

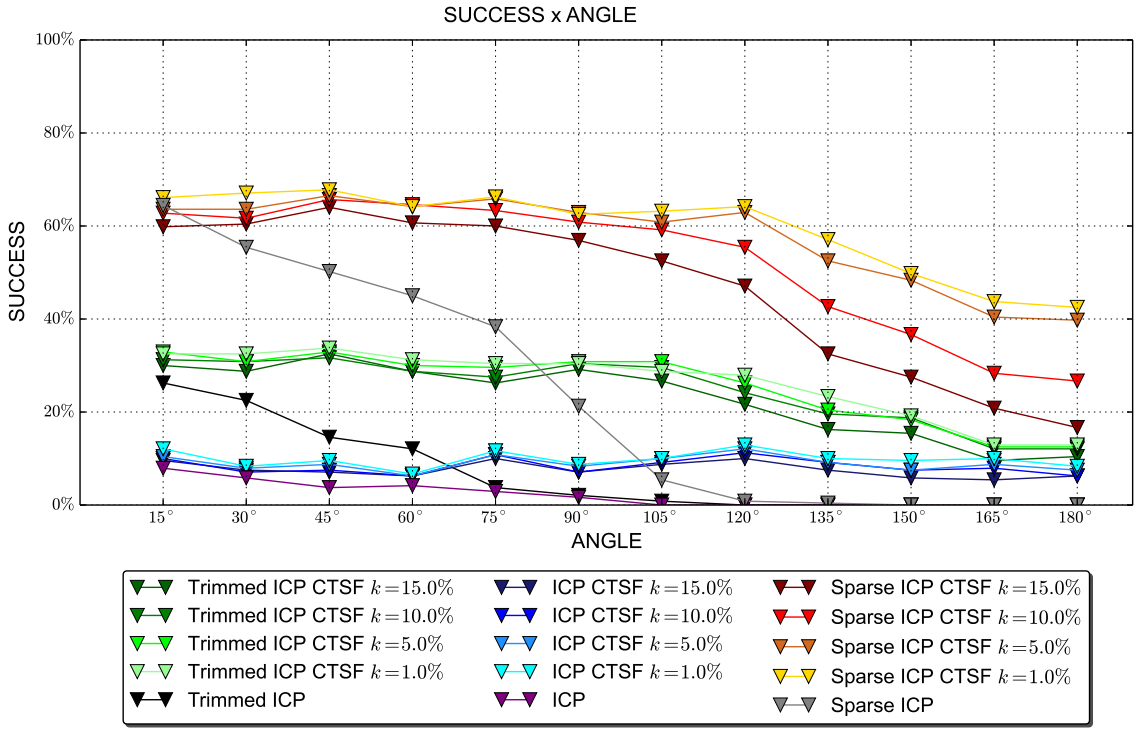


Figure 4.18: Convergence per angle on the Genus, with all partial overlapping levels.

In this experiment, it is clear that the CTSF enhances the convergence of the ICP also in partial overlapping situations. In the large majority of situations, the CTSF-based methods obtained better results than the original methods. However, the success rates were nowhere near the ones obtained on full overlap events, and there is a substantial decay as the proportion between the number points over the overlapping region and over the non-overlapping region gets smaller.

The Sparse ICP with the CTSF presented the best overall performance with partial overlapping. All the CTSF-based methods presented a slight variation with relation to the rotation angle, which indicates that the CTSF is effective for high angle displacements even in partial overlapping situations, yielding approximately the same convergence probability for all angle values. However, most of the parameter combinations for partial overlapping had proven themselves as hard scenarios, with very low convergence rates. The Sparse ICP had an advantage in comparison to the Trimmed ICP in this aspect, reaching much better rates particularly in cases where $\alpha = 25\%$, for which the Trimmed ICP was hardly successful. The original ICP method combined with the CTSF achieved good results for $\alpha = 12.5\%$ and $\beta = 75\%$ on the Bunny and Happy Budha, but in general performed poorly.

Analyzing the variation of the parameter k, smaller values of k presented slightly better results, as expected, with the exception of the Bunny. This tendency is more easily observed on cases with a low overlapping amount. The reduction of the number of points on the overlapping region and the increasement of the points on the non-overlapping region significantly impacted both Sparse ICP-CTSF and Trimmed ICP-CTSF methods. On Tables 4.12, 4.13, 4.10 and 4.15, we observe that the convergence rates were particularly lower when $\beta = 12.5\%$, and null in cases where $\beta = 12.5\%$ and $\alpha = 25\%$. Besides the natural difficulty of events with small overlapping regions, the CTSF also loses part of its accuracy in these cases, because the neighborhood used for the tensor estimation is hardly similar on points on both meshes.

4.5 ADDITIONAL RESULTS

In this section, we show qualitative results of alignment situations addressed by our dataset, and discuss the effect of the parameter b (weight step) of the iterative coarse-to-fine scheme used on the ICP-CTSF.

4.5.1 ITERATIVE COARSE-TO-FINE SCHEME

To show the importance of the coarse-to-fine scheme, we vary the parameter b of the weight combination between the CTSF and the matching step on the Genus, which is the point cloud that obtained the worst results on full overlapping events. The same events were executed for $b = 0$, 0.001, 0.01, 0.3, 0.5, 0.75 and 0.9, and Tables 4.16 and 4.17 shows the results for the same events considered on the first step of the evaluation. When $b = 0$, we perform a minimization totally based on the CTSF, followed by the original ICP. The other parameters used are the ones that reached better results: $k = 100\%$, $\alpha_{ellip} = 60°$ and $\phi_{max} = 60°$.

Table 4.16 shows that the weight variation in fact has influence on the results of the method, and better results are obtained with bigger values of b. The result with the bigger value of b ($b = 0.9$) is enhanced by approximately 5% when compared to the original result, with $b = 0.1$. The convergence rate is also reduced with smaller weights, and for $b = 0$ the rate is lower in all cases. In cases with outliers, particularly, the convergence rate was nearly null. In some events, the coarse alignment is executed, but the fine adjustment

Weight step (b)	Clean	Outliers	Noise	Noise + outliers	Overall
0.9	100.00%	94.72%	90.83%	81.81%	88.70%
0.75	100.00%	93.06%	91.39%	80.00%	87.65%
0.5	100.00%	93.06%	90.00%	76.94%	85.99%
0.3	100.00%	91.94%	88.33%	75.56%	84.75%
0.1	100.00%	91.67%	86.11%	73.06%	83.09%
0.01	100.00%	90.00%	83.61%	71.39%	81.42%
0.001	100.00%	88.89%	81.67%	68.47%	79.44%
0.0	100.00%	0.56%	59.72%	35.14%	40.12%

Table 4.16: Success per weight step - Genus

Weight step (b)	Clean	Outliers	Noise	Noise + outliers	Overall
0.9	227.70	598.91	621.81	659.35	589.61
0.75	89.09	248.33	255.37	291.58	251.42
0.5	41.68	117.40	127.38	155.52	128.15
0.3	28.04	77.03	88.99	111.78	89.69
0.1	18.63	50.22	63.26	81.20	63.38
0.01	13.31	35.20	50.06	61.77	47.88
0.001	11.49	32.14	42.84	57.83	43.64
0.0	6.99	7.00	6.36	5.99	6.41

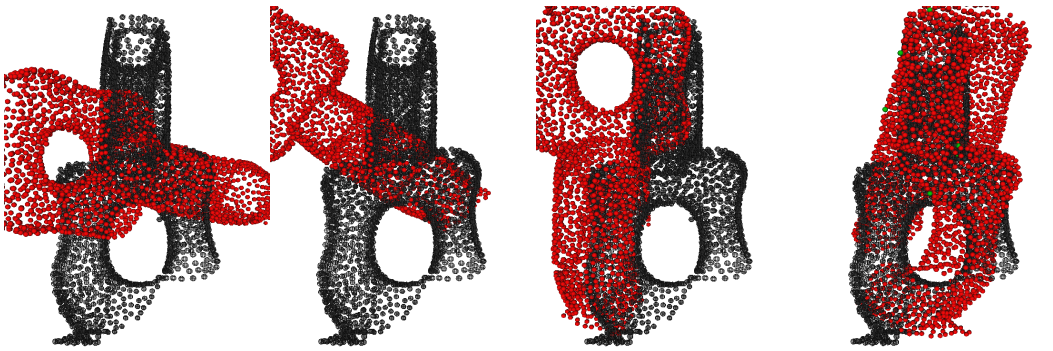
Table 4.17: Average number of iterations per weight step - Genus

between the clouds was insufficient to consider the event successful.

However, the number of iterations grows exponentially when the weight step gets closer to 1, as confirmed by Table 4.17. Therefore, this result comproves the tradeoff between chances of convergence and number of iterations associated with this parameter.

4.5.2 QUALITATIVE RESULTS

Figures 4.21, 4.23 and 4.19 show examples of successful alignment situations respectively with partial overlapping and additive noise. In all figures, the two point clouds are red and black. When the point have a correct correspondence, its color is changed to green. Figures 4.22, 4.24 and 4.20 show the variation of the RMS error, in red, and the ground-truth RMS error (GT-RMS), in green, throughout the iterations. Points marked in the x-axis indicate the iterations in which the weight is changed.

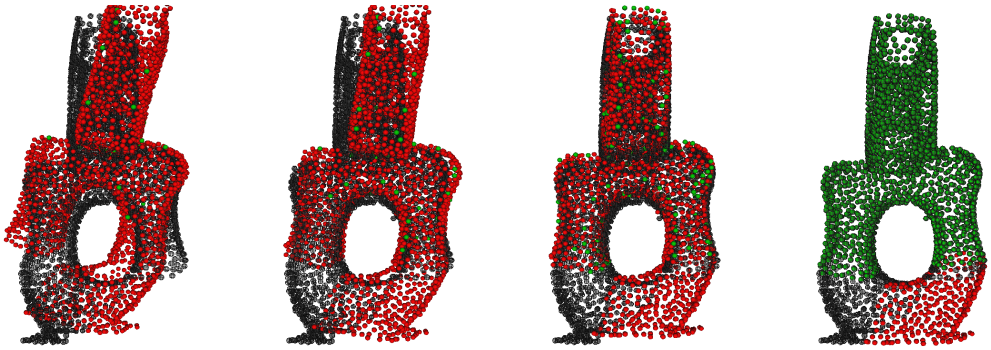


(a) Iteration 1
RMS = 0,520668
GT-RMS = 0,449581

(b) Iteration 2
RMS = 0,520668
GT-RMS = 0,449581

(c) Iteration 7
RMS = 0,330357
GT-RMS = 0,269590

(d) Iteration 20
RMS = 0,113371
GT-RMS = 0,165358



(e) Iteration 40
RMS = 0,060175
GT-RMS = 0,098581

(f) Iteration 45
RMS = 0,056042
GT-RMS = 0,077697

(g) Iteration 68
RMS = 0,036350
GT-RMS = 0,024591

(h) Iteration 92
RMS = 0,006032
GT-RMS = 0,000850

Figure 4.19: Convergence for the Genus, partial overlapping with $\alpha = 75\%$, $\beta = 12.5\%$, rotation of 105°. Method: Trimmed ICP+CTSF, with $k = 1\%$, $\alpha_{ellip} = 60°$, $\phi_{max} = 45°$. Green points indicate correct correspondences.

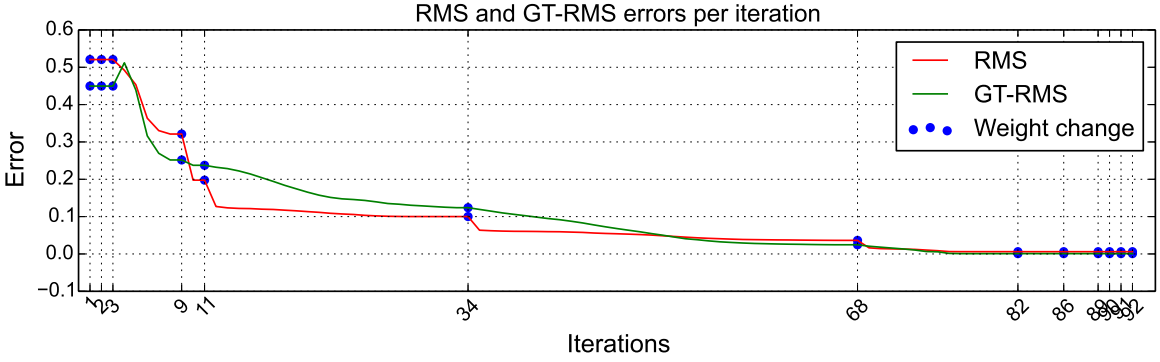
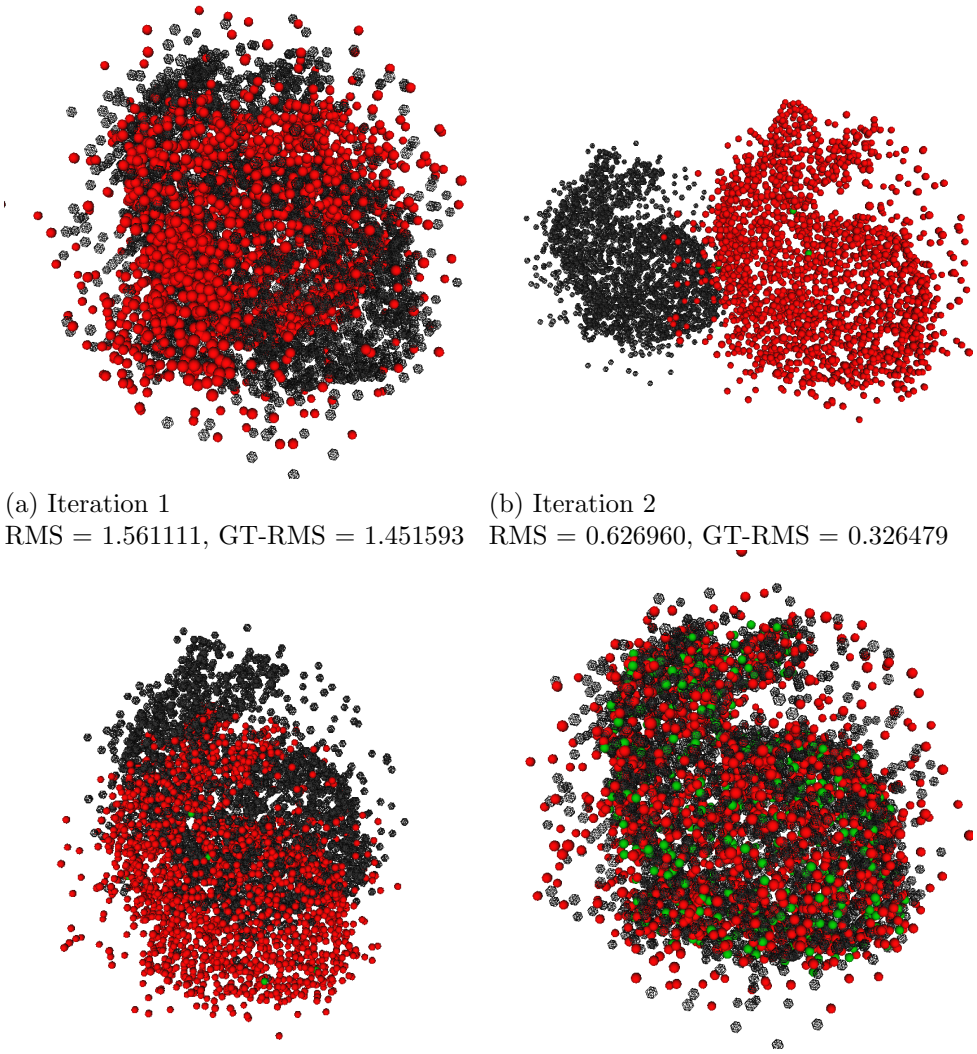


Figure 4.20: RMS and GT-RMS errors per iteration on the Genus. Between iterations 10 and 50, the GT-RMS is higher than the RMS. This can indicate that the method was following a wrong local minimum but recovered itself, yielding a successful result.



(a) Iteration 1
RMS = 1.561111, GT-RMS = 1.451593

(b) Iteration 2
RMS = 0.626960, GT-RMS = 0.326479

(c) Iteration 3
RMS = 0.528149, GT-RMS = 0.088650

(d) Iteration 92
RMS = 0.041573, GT-RMS = 0.070954

Figure 4.21: Convergence for the Bunny on a case with outliers and noise, original ICP with CTSF, angle of 180°, with 20% of outliers and $\delta = 0.05$, $k = 100\%$, $\alpha_{ellip} = 60°$ and $\phi_{max} = 45°$. Note that in (c) the point clouds are already have the same orientation. Green points indicate correct correspondences.

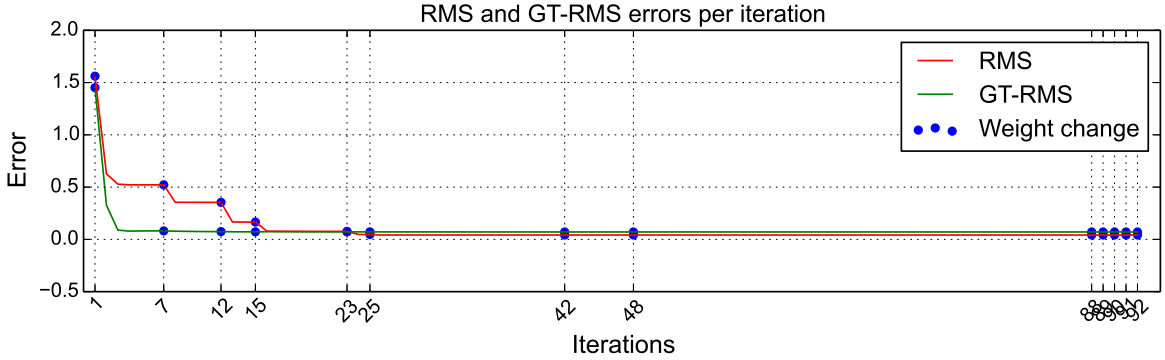
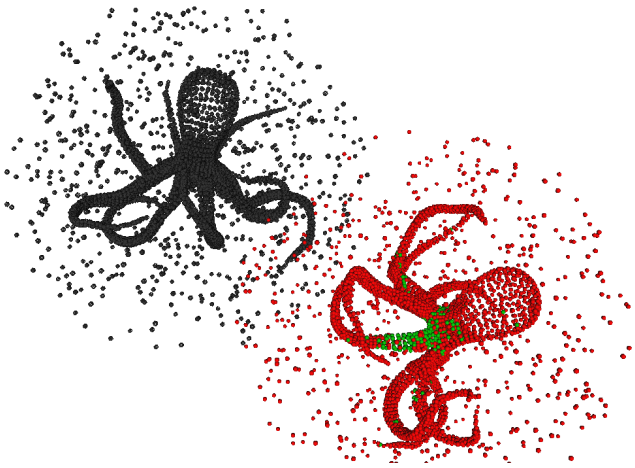


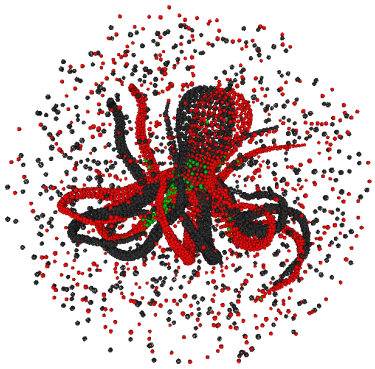
Figure 4.22: RMS and GT-RMS errors per iteration on the Bunny example. In this case, the alignment is reached in the first iterations, and only slightly enhanced with smaller weights.

(a) Iteration 1
RMS = 1.093765
GT-RMS = 1.028373

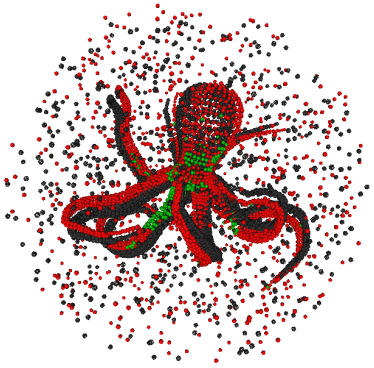


(b) Iteration 2
RMS = 0.921383
GT-RMS = 0.850880

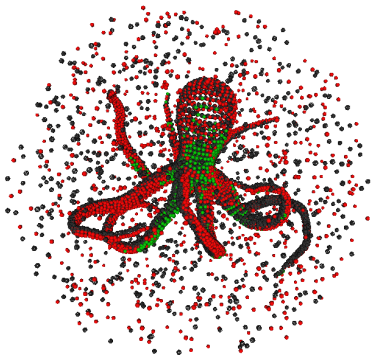
(c) Iteration 5
RMS = 0.397163
GT-RMS = 0.080367



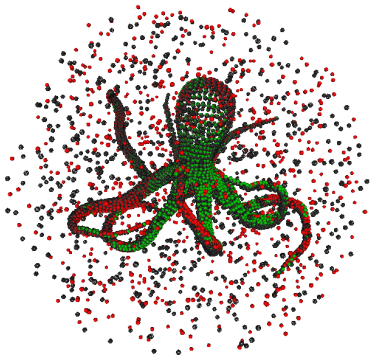
(d) Iteration 13
RMS = 0.393860
GT-RMS = 0.068015



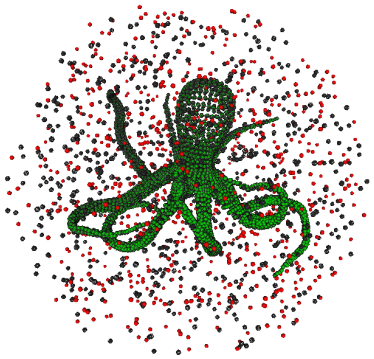
(e) Iteration 18
RMS = 0.245980
GT-RMS = 0.029073



(f) Iteration 26
RMS = 0.113790
GT-RMS = 0.011223



(g) Iteration 32
RMS = 0.047715
GT-RMS = 0.003969



(h) Iteration 55
RMS = 0.027734
GT-RMS = 0.01144

Figure 4.23: Sequence of convergence for the Octopus on a case with outliers, original ICP with CTSF, angle of 105°, with 20% of outliers, $k = 100\%$, $\alpha_{ellip} = 60°$ and $\phi_{max} = 45°$. Green points indicate correct correspondences.

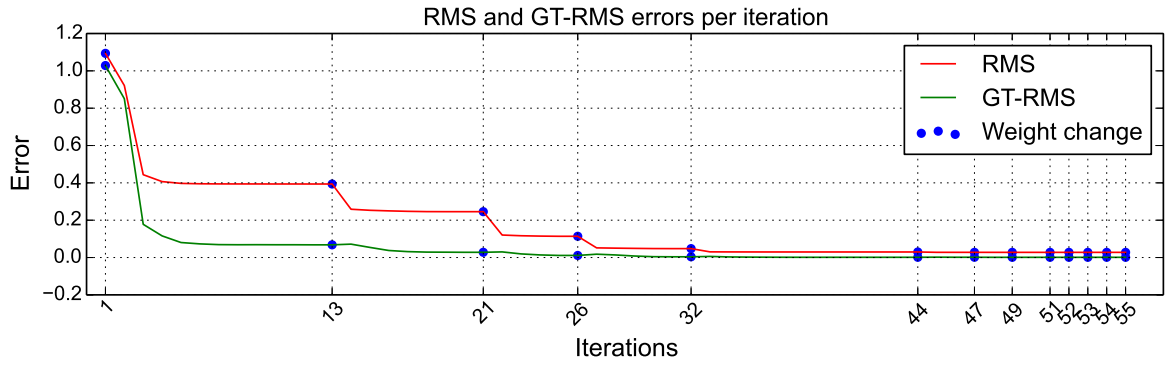


Figure 4.24: RMS and GT-RMS errors per iteration on the Octopus example. Note that the effect of the weight changes in the errors are well defined and the method continues to follow the same correct optima since its first iterations.

5 CONCLUSION

This work presents a new method for rigid registration, based on tensor eigenvalues. This is the first rigid registration approach that treats covariance matrices as second-order tensors. The tensors are estimated by a two-step voting process using 3D tensor structuring elements, aiming to infer how likely their neighborhood form a surface. In order to compare tensors, we define the CTSF, a similarity factor based on tensor invariant features. This factor is used on the ICP matching step in order to enhance the quality of correspondences. A heuristic weighting strategy between the Euclidean distance and the CTSF is proposed to guide the solution from a coarse alignment, based on tensor dissimilarity, to fine, based on Euclidean distance. This enhances the convergence probability, specially on wider initial angle situations. Since only the matching step is modified, our approach can be used alongside many minimization methods. The major drawback is the preprocessing time needed for the tensor estimation, specially on large point sets.

The analysis made on our dataset is quantitative and considers only the success or failure of each event. On that sense, our experiments reveal that the convergence rate of the CTSF methods is enhanced in situations of wider angles, additive noise and outliers when compared to their respective non-CTSF versions. On partial overlapping situations, the CTSF was capable of enhancing the convergence of the Trimmed ICP and the Sparse ICP in general. The wide angle registration of partial overlapping surfaces makes easier the task of 3D object reconstruction, thus the importance of such results.

The main parameter added to the ICP is the size k of the neighborhood of each point, which defines the extent of the structuring elements in the tensor estimation. It is important to note that no input point is discarded at any time during the process. Our method takes into account the information of all input points regardless they are inliers or outliers.

As future works, the CTSF can be used as a dissimilarity factor between any second-order tensors and applied in tasks other than rigid registration. Since our heuristic weights Euclidean distance and dissimilarity between tensors, any covariance matrix that represents locally the surface can be used instead of ours. Other steps and variants of the ICP can also be adapted to use the CTSF to enhance the alignment, such as the minimization

and the selection of points. Our tensor estimation step can be optimized in terms of speed, by computing the nearest neighbors list faster. It is also possible to determine a value of k that adapts better for each region of the mesh, which could reach better results on partial overlapping cases.

REFERENCES

AIGER, D.; MITRA, N. J.; COHEN-OR, D. 4-points congruent sets for robust pairwise surface registration. In: ACM. **ACM Transactions on Graphics (TOG)**, 2008. v. 27, n. 3, p. 85.

ALBARELLI, A.; RODOLÀ, E.; TORSELLO, A. Fast and accurate surface alignment through an isometry-enforcing game. **Pattern Recognition**, Elsevier, v. 48, n. 7, p. 2209–2226, 2015.

BESL, P.; MCKAY, N. D. A method for registration of 3-d shapes. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, v. 14, n. 2, p. 239–256, 1992.

BOUAZIZ, S.; TAGLIASACCHI, A.; PAULY, M. Sparse iterative closest point. **Computer Graphics Forum**, Blackwell Publishing Ltd, v. 32, n. 5, p. 113–123, 2013. ISSN 1467-8659.

BOYER, E.; BRONSTEIN, A. M.; BRONSTEIN, M. M.; BUSTOS, B.; DAROM, T.; HORAUD, R.; HOTZ, I.; KELLER, Y.; KEUSTERMANS, J.; KOVNATSKY, A. et al. Shrec 2011: robust feature detection and description benchmark. In: **3DOR2011-Eurographics Workshop on 3D Object Retrieval**, 2011. p. 71–78.

BRONSTEIN, A. M.; BRONSTEIN, M. M.; BUSTOS, B.; CASTELLANI, U.; CRISTANI, M.; FALCIDIENO, B.; GUIBAS, L. J.; KOKKINOS, I.; MURINO, V.; OVSJANIKOV, M. et al. Shrec'10 track: Feature detection and description. In: **3DOR-Eurographics Workshop on 3D Object Retrieval**, 2010. p. 79–89.

CHEN, C.-S.; HUNG, Y.-P.; CHENG, J.-B. Ransac-based darces: A new approach to fast automatic registration of partially overlapping range images. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, v. 21, n. 11, p. 1229–1234, 1999.

CHETVERIKOV, D.; SVIRKO, D.; STEPANOV, D.; KRSEK, P. The trimmed iterative closest point algorithm. In: IEEE. **Pattern Recognition, 2002. Proceedings. 16th International Conference on**, 2002. v. 3, p. 545–548.

CHUNG, D. H.; YUN, I. D.; LEE, S. U. Registration of multiple-range views using the reverse-calibration technique. **Pattern Recognition**, Elsevier, v. 31, n. 4, p. 457–464, 1998.

CIRUJEDA, P.; CID, Y. D.; MATEO, X.; BINEFA, X. A 3d scene registration method via covariance descriptors and an evolutionary stable strategy game theory solver. **International Journal of Computer Vision**, Springer, p. 1–24, 2015.

CORSINI, M.; CIGNONI, P.; SCOPIGNO, R. Efficient and flexible sampling with blue noise properties of triangular meshes. **Visualization and Computer Graphics, IEEE Transactions on**, IEEE, v. 18, n. 6, p. 914–924, 2012.

DÍEZ, Y.; ROURE, F.; LLADÓ, X.; SALVI, J. A qualitative review on 3d coarse registration methods. **ACM Computing Surveys (CSUR)**, ACM, v. 47, n. 3, p. 45, 2015.

DONG, J.; PENG, Y.; YING, S.; HU, Z. Lietricp: An improvement of trimmed iterative closest point algorithm. **Neurocomputing**, Elsevier, v. 140, p. 67–76, 2014. Disponível em: <http://dx.doi.org/10.1016/j.neucom.2014.03.035>.

DONOSER, M.; BISCHOF, H. 3d segmentation by maximally stable volumes (msvs). In: IEEE. **Pattern Recognition, 2006. ICPR 2006. 18th International Conference on**, 2006. v. 1, p. 63–66.

EGGERT, D. W.; LORUSSO, A.; FISHER, R. B. Estimating 3-d rigid body transformations: a comparison of four major algorithms. **Machine Vision and Applications**, Springer, v. 9, n. 5-6, p. 272–290, 1997.

FITZGIBBON, A. W. Robust registration of 2d and 3d point sets. **Image and Vision Computing**, Elsevier, v. 21, n. 13, p. 1145–1153, 2003.

GELFAND, N.; IKEMOTO, L.; RUSINKIEWICZ, S.; LEVOY, M. Geometrically stable sampling for the icp algorithm. In: IEEE. **3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on**, 2003. p. 260–267.

GELFAND, N.; MITRA, N. J.; GUIBAS, L. J.; POTTMANN, H. Robust global registration. In: **Symposium on geometry processing**, 2005. v. 2, n. 3, p. 5.

GODIN, G.; RIOUX, M.; BARIBEAU, R. Three-dimensional registration using range and intensity information. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. **Photonics for Industrial Applications**, 1994. p. 279–290.

HERMANS, J.; SMEETS, D.; VANDERMEULEN, D.; SUETENS, P. Robust point set registration using em-icp with information-theoretically optimal outlier handling. In: IEEE. **Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on**, 2011. p. 2465–2472.

HORN, B. K. Closed-form solution of absolute orientation using unit quaternions. **JOSA A**, Optical Society of America, v. 4, n. 4, p. 629–642, 1987.

KINDLMANN, G. Superquadric tensor glyphs. In: EUROGRAPHICS ASSOCIATION. **Proceedings of the Sixth Joint Eurographics-IEEE TCVG conference on Visualization**, 2004. p. 147–154.

MAIER-HEIN, L.; FRANZ, A. M.; SANTOS, T. R. dos; SCHMIDT, M.; FANGERAU, M.; MEINZER, H.; FITZPATRICK, J. M. Convergent iterative closest-point algorithm to accomodate anisotropic and inhomogenous localization error. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, v. 34, n. 8, p. 1520–1532, 2012.

MASUDA, T.; SAKAUE, K.; YOKOYA, N. Registration and integration of multiple range images for 3-d model construction. In: IEEE. **Pattern Recognition, 1996., Proceedings of the 13th International Conference on**, 1996. v. 1, p. 879–883.

MEDIONI, G.; TANG, C.-K.; LEE, M.-S. Tensor voting: Theory and applications. **Proceedings of RFIA, Paris, France**, v. 3, 2000.

MELLADO, N.; AIGER, D.; MITRA, N. J. Super 4pcs fast global pointcloud registration via smart indexing. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**, 2014. v. 33, n. 5, p. 205–215.

MEN, H.; GEBRE, B.; POCHIRAJU, K. Color point cloud registration with 4d icp algorithm. In: IEEE. **Robotics and Automation (ICRA), 2011 IEEE International Conference on**, 2011. p. 1511–1516.

MORDOHAI, P.; MEDIONI, G. Tensor voting: a perceptual organization approach to computer vision and machine learning. **Synthesis Lectures on Image, Video, and Multimedia Processing**, Morgan & Claypool Publishers, v. 2, n. 1, p. 1–136, 2006.

POTTMANN, H.; WALLNER, J.; HUANG, Q.-X.; YANG, Y.-L. Integral invariants for robust geometry processing. **Computer Aided Geometric Design**, Elsevier, v. 26, n. 1, p. 37–60, 2009.

REYES, L.; MEDIONI, G.; BAYRO, E. Registration of 3d points using geometric algebra and tensor voting. **International Journal of Computer Vision**, Springer, v. 75, n. 3, p. 351–369, 2007.

RUSINKIEWICZ, S.; LEVOY, M. Efficient variants of the icp algorithm. In: IEEE. **3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on**, 2001. p. 145–152.

SALVI, J.; MATABOSCH, C.; FOFI, D.; FOREST, J. A review of recent range image registration methods with accuracy evaluation. **Image and Vision Computing**, Elsevier, v. 25, n. 5, p. 578–596, 2007.

SANTAMARÍA, J.; CORDÓN, O.; DAMAS, S. A comparative study of state-of-the-art evolutionary image registration methods for 3d modeling. **Computer Vision and Image Understanding**, Elsevier, v. 115, n. 9, p. 1340–1354, 2011.

SEGAL, A.; HAEHNEL, D.; THRUN, S. Generalized-icp. In: **Robotics: Science and Systems**, 2009. v. 2, n. 4.

SERVOS, J.; WASLANDER, S. L. Multi channel generalized-icp. In: IEEE. **Robotics and Automation (ICRA), 2014 IEEE International Conference on**, 2014. p. 3644–3649.

SHARP, G. C.; LEE, S. W.; WEHE, D. E. Invariant features and the registration of rigid bodies. In: IEEE. **Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on**, 1999. v. 2, p. 932–937.

SHARP, G. C.; LEE, S. W.; WEHE, D. K. Icp registration using invariant features. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, v. 24, n. 1, p. 90–102, 2002.

SIPIRAN, I.; BUSTOS, B. Harris 3d: a robust extension of the harris operator for interest point detection on 3d meshes. **The Visual Computer**, Springer, v. 27, n. 11, p. 963–976, 2011.

SUN, J.; OVSJANIKOV, M.; GUIBAS, L. A concise and provably informative multi-scale signature based on heat diffusion. In: WILEY ONLINE LIBRARY. **Computer graphics forum**, 2009. v. 28, n. 5, p. 1383–1392.

TAM, G.; CHENG, Z.-Q.; LAI, Y.-K.; LANGBEIN, F.; LIU, Y.; MARSHALL, D.; MARTIN, R.; SUN, X.-F.; ROSIN, P. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. **Visualization and Computer Graphics, IEEE Transactions on**, v. 19, n. 7, p. 1199–1217, 2013. ISSN 1077-2626.

TAREL, J.-P.; CIVI, H.; COOPER, D. B. Pose estimation of free-form 3d objects without point matching using algebraic surface models. In: CITESEER. **Proceedings of IEEE Workshop on Model-Based 3D Image Analysis**, 1998. p. 13–21.

TOMBARI, F.; SALTI, S.; STEFANO, L. D. Unique signatures of histograms for local surface description. In: **Computer Vision–ECCV 2010**, 2010. p. 356–369.

TRUCCO, E.; FUSIELLO, A.; ROBERTO, V. Robust motion and correspondence of noisy 3-d point sets with missing data. **Pattern recognition letters**, Elsevier, v. 20, n. 9, p. 889–898, 1999.

VIEIRA, M. B.; MARTINS, P.; ARAUJO, A.; CORD, M.; PHILIPP-FOLIGUET, S. Smooth surface reconstruction using tensor fields as structuring elements. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**, 2004. v. 23, n. 4, p. 813–823.

WESTIN, C.-F.; PELED, S.; GUDBJARTSSON, H.; KIKINIS, R.; JOLESZ, F. A. Geometrical diffusion measures for mri from tensor basis analysis. **Proceedings of the 5th Annual Meeting of ISMRM**, p. 1742, 1997.

YANG, J.; LI, H.; JIA, Y. Go-icp: Solving 3d registration efficiently and globally optimally. In: IEEE. **Computer Vision (ICCV), 2013 IEEE International Conference on**, 2013. p. 1457–1464.

ZAHARESCU, A.; BOYER, E.; VARANASI, K.; HORAUD, R. Surface feature detection and description with applications to mesh matching. In: IEEE. **Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on**, 2009. p. 373–380.

ZHONG, Y. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In: IEEE. **Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on**, 2009. p. 689–696.