Mining Large Point Clouds for Feature Matching of LIDAR Datasets using Self-Similarity

Daniel Pulido[1], **Anthony Stefanidis**[2]
[1]Leidos Inc., Fairfax, VA, USA
[2]Department of Computational and Data Sciences, George Mason University, Fairfax, VA, USA

Abstract – *LIDAR captured point cloud datasets are another type of big data. Mining such large point clouds for feature selection to perform point cloud matching is a problem of fundamental importance for the geospatial sciences. Standard approaches to point cloud matching face a number of challenges including datasets that exhibit spatial and scale variations. This paper proposes a novel 3D feature descriptor, referred to as "Self-Similar Spin Images", that provides a robust method to perform spatial matching of point clouds by combining the robustness of local self-similarity with the descriptive power of spin images. This descriptor is used to detect the scale difference between point clouds by introducing the "Self-Similar Keyscale" metric. The proposed method was tested with model and LIDAR datasets*.

Keywords: lidar; matching; self-similarity; spin-images; scale

1 Introduction

The rapid growth of Light Detection And Ranging (LIDAR) technologies that collect, process, and disseminate 3D point clouds have allowed for increasingly accurate spatial modeling and analysis of the real world. With millions of points captured per square kilometer, a typical tile of LIDAR data is often of a size of few GBs, and the processing of such datasets represents challenges commonly associated with big data.

LIDAR sensors can generate massive 3D point clouds that provide highly detailed spatial information of an area of interest. Fusing such point clouds allows the generation of broad coverage products that mitigate the particular disadvantages of individual point clouds (e.g. uneven point distribution, small area coverage) and combine their advantages (high accuracy potential). However, this fusion of point clouds collected from a variety of sources can prove challenging.

Pairwise point cloud fusion requires pre-processing to align the two data into a common frame. This common frame may vary from problem to problem, and it can be a common coordinate system, spectral space, or some other abstract reference frame from which the data's features can be compared. A common drawback of such alignment methods is that they often assume knowledge of scale differences in the data. In order to address this challenge, in this paper we present a novel approach to mine features in large point clouds and then use such features to identify scale differences among two point clouds that are to be fused.

1.1 Self-Similarity

Our approach is based on the concept of self-similarity as it can be used to detect prominent features in a point cloud dataset. While self-similarity has featured prominently in fractal theory, and has been applied to computing spatial dimensionality and radar sensor designs, its application to image and point cloud matching is relatively recent. The use of self-similarity has been exploited as a feature descriptor for imagery and video registration [1], detecting deformable shapes [2], and interest point detection [3]. The authors in [4] exploit a self-similarity based feature descriptor to match multi-modal images. In [1], the authors used self-similarity to match objects in an image to hand-drawn templates. Self-similarity was exploited in [5] to match a 2D image to a LIDAR point cloud by using pixel intensities and spatial gradients. Local self-similarity has recently been applied as a local feature descriptor for 3D point cloud matching by Huang [5]. Their approach exploits the local self-similarity of geometric properties of point clouds, principally the local surface normal. In this paper we advance this approach to incorporate Spin Images, a far more descriptive point cloud feature descriptor than the simple local normal metric used in [5].

1.2 Scale Detection

A second component of our approach relates to the assessment of scale variations between two datasets that are to be fused, to ensure that they are indeed comparable ([6], [7], [8]). In [9] a 3D extension of the well-known SIFT descriptor was proposed known as LD-SIFT. Similarly, in [10] a rotational projection statistics (RoPS) multi-scale representation of features is used to perform object recognition in a scale invariant manner. An entirely different approach is to directly detect the scales between two point clouds. In [11] the authors compute a characteristic length for a point cloud, referred to as its "keyscale", that is an optimal scale that best captures point cloud feature descriptors. In our work we extend this keyscale-based process by incorporating self-similar Spin

ISBN: 1-60132-453-7, CSREA Press ©

Images, a more descriptive feature descriptor than the standard Spin Image used in [11].

1.3 Study Description

This study will address the problem of matching two point clouds from potentially different sources. Specifically, we will consider two problems: scale matching and feature matching. Scale matching consists of computing feature metrics of each point cloud and analyzing their distributions to determine scale differences. Feature matching consists of defining local descriptors that are invariant to common dataset distortions (e.g., rotation and translation).

In Section 2 we present the application of local self-similarity to matching point cloud features and the detection of pairwise scale differences. In Section 3 we present experimental results of our proposed approach, and its comparison to a baseline method with model and LIDAR data sets. Finally, Section 4 draws conclusions based on the experimental results.

2 Local Self-Similarity Matching

2.1 Introduction

The notion of local self-similarity is based on the idea of identifying local geometric patterns across data sources regardless of modality. Finding similar features across data sets is based on the notion that "*local internal layouts of self-similarities are shared by these images, even though the patterns generating those self-similarities are not shared by those images*" [1]. That is, by examining how the structure near a feature is self-similar, that self-similarity will also appear in similar features in other data sources. To demonstrate the technique, the authors in [1] were able to match human figures in an image to a hand-drawn stick figures of a human pose.

Specifically, local self-similarity is computed as a correlation surface using a particular local descriptor (e.g., pixel intensity) for a given neighborhood of pixels. This correlation surface is then transformed into a binned log-polar representation to account for local spatial affine deformations. This descriptor is constructed in a manner to ensure it provides a compact representation and mitigates against spatial distortions as well as local non-rigid deformations.

The local self-similarity approach was extended from 2D images to 3D point clouds in [5] by selecting a simple geometric descriptor, the surface normal, to generate the needed correlation surfaces for a given neighborhood of points. The authors also incorporated multiple descriptors in building the self-similarity correlation surface such as curvature and intensity.

In this study, the application of local self-similarity to 3D point features is used to match their respective point clouds in both scale and space. The proposed approach extracts "Self-Similar Spin Image" (SSSI) descriptors, as defined in Section 2.2 at feature locations for both point clouds. The extracted SSSIs are then analyzed using a PCA analysis to compute the "Self-Similar Keyscale" (SSK) metric, as described in Section 2.3, of each point cloud in order to determine their relative scale differences. Once the two point clouds are scale matched, SSSIs that are deemed to be matches define a transformation between the two point clouds which can be used to register them.

Finally, for massive point clouds it can be computationally infeasible to attempt to match every point. Therefore, point clouds must typically be preprocessed using a feature detector to identify points that represent unique features in the scene. The feature detectors in the scientific literature include Harris points [12], Heat Kernels [13], and Mesh SIFT [14], among others. However, the feature detector selected for this study is the "Maximum of Principle Curvature" (MoPC) method as it provides a robust and stable locator of interest regions invariant to rotations and local affine distortions [5]. Figure 1 provides an example of feature points extracted using MoPC.

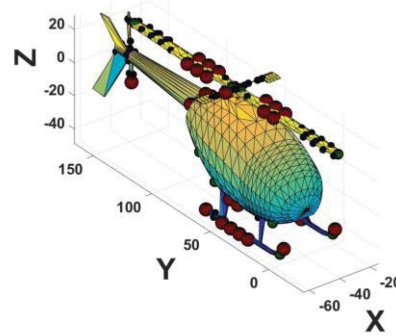

Figure 1: MoPC feature extraction. The marker sizes are proportional to the scale at which the feature was detected.

2.2 Self-Similar Spin Images

The self-similarity framework introduced in [1] computes an attribute (e.g., local normal) for a data point (e.g., 2D pixel or 3D point) and its neighbors. This attribute is then compared against those neighboring attributes using some comparator metric (e.g., correlation) and then normalized. This collection of comparison values defines the self-similarity descriptor.

In this study, it is shown that by using a more powerful descriptor within the self-similarity framework we achieve a more robust method of feature matching. In particular, using a well-established robust descriptor such as Spin Images, instead of just the local normal, can provide a more descriptive correlation surface. The basic approach is to define a neighborhood near each detected feature point, then compute and compare the Spin Image of the feature point against the Spin Image of its neighbors. The comparison is performed by computing a correlation coefficient and then using their values to build a spherical correlation surface.

Therefore, a SSSI is computed follows:

a) Define *C=Point Cloud*, *F = Feature Points*, and *R = Neighborhood Size*.
b) Let $f \in F$ be an extracted feature point
c) Let $N = \{p \mid ||p-f|| \leq R\}$ be a spherical neighborhood of radius R about the feature point that includes all points within that radius (Figure 2)
d) Compute a Spin Image S_n for every point within this neighborhood $n \in N$.
e) Compare the Spin Image of the feature point S_f to every other point's Spin Image in the neighborhood S_n by using a similarity metric, the correlation coefficient, $M_{(fn)} = correlation(S_f, S_n)$.
f) Define a spherical coordinate system as defined in Figure 2 with the origin set at the feature point, the X axis is the local principal direction, the Z axis is the local normal, and Y axis is the cross product of Z and X to define a right-handed coordinate system.
g) The neighborhood is then spherically binned and each bin is given the value of the average similarity metric $M_{(fn)}$ within that bin.
h) The values in this correlation surface are then normalized to have a maximum of one.

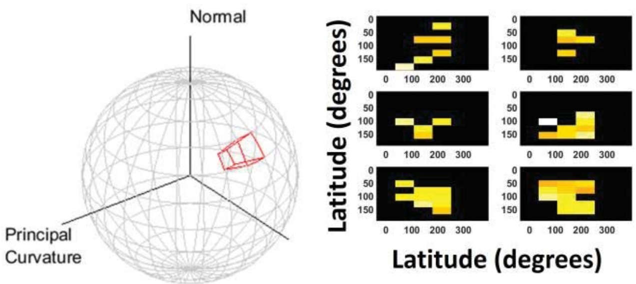

Figure 2: Descriptor local coordinate system and quantization (left). The highlighted region depicts a spherical bin. Example radial slices of the corresponding descriptor (right).

This spherical correlation surface is the desired feature descriptor that can be compared against other features in other point clouds to find a match. Figure 2 also presents six radial slices of the descriptor. For the testing used in this study the number of bins in the radial, longitude, and latitude directions are six, eight, and six, respectively.

2.3 Self-Similar Keyscale

In order for the above descriptor to be truly scale-invariant the size of the spherical neighborhood must be determined. In the approach described in [5] the size of this neighborhood was empirically chosen to be four times "the detected scale at which the principal curvature reaches its local maxima". However, for this method to be scale-invariant and minimize user-interaction a method must be devised to determine the scale at which the point cloud descriptors should be computed. To determine this characteristic size of the point cloud we extend the technique introduced by Tamaki (et al) [11].

The technique relies on computing a "keyscale", which is a characteristic scale by which if two point clouds are scaled by the ratio of their respective keyscales then the point clouds can be matched. A point cloud's keyscale is computed by performing a Principal Component Analysis (PCA) of its Spin Images over a range of scales and finding the scale that yields the minimum cumulative contribution rate. The motivation behind this technique is that for both very small scales and very large scales all Spin Images will tend to look the same, either all representing a plane or a point, respectively. Therefore, it is conjectured that there is an optimal scale in between.

For example, Figure 3 shows a sample point cloud and the same cloud scaled by a factor of 10. The PCA scores for the principal component of both point clouds are displayed in Figure 4. The keyscale for the original Ant point cloud is found to be 6.55 and for the scaled point cloud is 65.5 which yields a keyscale ratio of 10.

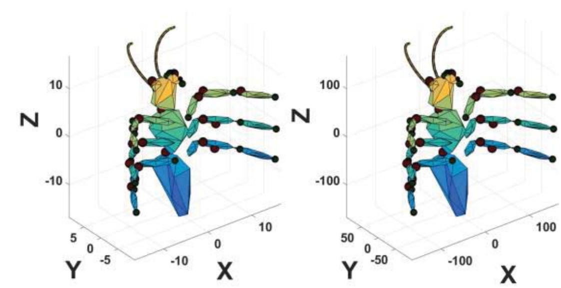

Figure 3: Sample point cloud (left) and its 10x scaled version (right) with their respective extracted features using MoPC.

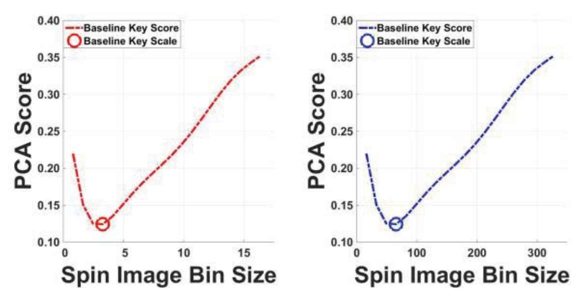

Figure 4: The keyscales of the sample point cloud (left) and its 10x scaled version (right) computed from its PCA score.

In this research we extend this approach to compute the keyscale of a point cloud by performing this multi-scale PCA analysis on the SSSIs. By applying this scale matching technique to a local descriptor that captures the self-similarity of local geometric patterns instead of simply the distribution of neighboring points it is shown that this will provide a more robust scale detector. The basic approach is to compute the SSSI of every feature point for a range of scales and then a perform PCA analysis. The extrema of the principal component yields the keyscale. To compute the SSK:

a) Let R_{min} and R_{max} be minimum and maximum spherical neighborhood sizes, respectively.

b) Let T be the set of neighborhood sizes to be tested in the range $[R_{min}, R_{max}]$ such that the ratio of sequential scales are equal.
c) Let $R \in T$ and compute the spin image S_f for every feature point $f \in F$ with a neighborhood size R.
d) Vectorize each S_f and compute the PCA decomposition of all S_f.
e) Compute the cumulative contribution rates of the PCA bands
f) Repeat for all values of R in T
g) Compute the value of R that maximizes the cumulative contribution rate of the first principal component.

The values for R_{min} and R_{max} must be selected. However, they can be readily chosen based on a characteristic size of the point cloud such that the keyscale algorithm is insensitive to their values. Specifically, we use, ε, the average nearest neighbor distance of all points in the point cloud, as the characteristic size and we set $R_{min} = \varepsilon$ and $R_{max} = 20\varepsilon$.

Figure 5 shows an example of the PCA scores and the highlighted keyscale found at the curve maximum. It should be noted that unlike the standard keyscale approach introduced in [11] the keyscale is not the minimum of the PCA rate curve but for SSK it is the maximum.

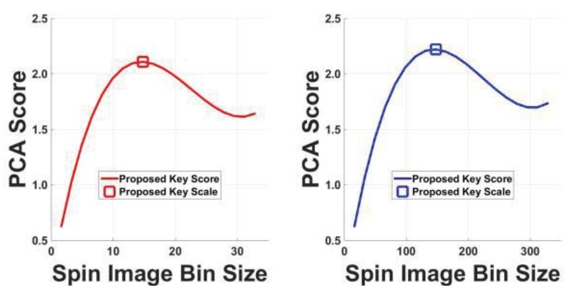

Figure 5: The keyscales of the sample point cloud (left) and its 10x scaled version (right) computed from its PCA score.

The motivation behind this approach is similar to that of the keyscale approach in [11], whereas values of R that are either too small or too large will yield similar flat spin images and so an optimal value in between is expected.

3 Results

3.1 Overview

To evaluate the performance of the proposed algorithms two categories of experiments are performed: Scale Matching and Feature Matching. The purpose of the Scale Matching experiments in Section 3.3 is to evaluate the ability of detecting the scale differences between a reference and target point cloud. Specifically, the reference and target point clouds differ by a Rotation/Scale/Translation (RST) transformation that contains a scale component that is to be computed using the SSK approach defined in Section 2.3. The proposed SSK approach will be compared against the baseline Keyscale approach described in [11]. The purpose of the Feature Matching experiments in Section 3.4 is to evaluate the ability to spatially match the features of a reference and target point cloud. As in the Scale Matching experiments, the reference and target point clouds differ by a RST transformation that must be computed using the SSSI approach defined in Section 2.2. The proposed SSSI approach will be compared against the baseline Self-Similar approach described in [5].

3.2 Test Data

The feature and scale matching methods developed in Section 2.2 and Section 2.3, respectively, were tested against model and LIDAR datasets presented in Figure 6. These datasets were acquired from online academic and government sources (see the Appendix).

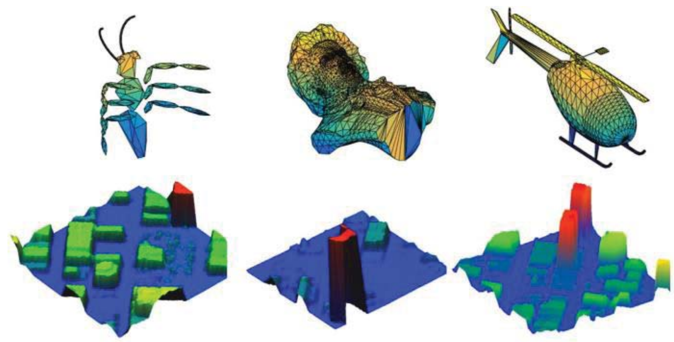

Figure 6: Test Model Point Clouds (Ant, Beethoven, Chopper) and LIDAR (SMALL, LARGE, MIX).

The test data provides a set of point clouds with a varying range of complexity. The model point clouds provide simpler simulated datasets with well-defined feature points. The LIDAR point clouds provide a more realistic data set with the types of typical features found in urban scenes. In addition, to test the proposed and baseline methods for robustness against a wide range of terrains, a set of twenty five LIDAR point clouds were also tested. This robustness test set are presented in the Appendix. The model point clouds contain between 1 and 10 thousand points while the LIDAR point clouds contain approximately 3.5 million points.

Table 1: List of transformations applied to test point clouds.

Label	Rotation	Scale	Translation
Y180	180 deg. about Y	None	None
S10	None	10	None
RST	-45 deg. about Y	2	Max dimension

For the experiments, the test point clouds are transformed using a variety of transformations to test the limits of the proposed algorithms. The transformations and the labels used to identify them are described in Table 1.

3.3 Scale Matching

For the scale matching experiments the test point clouds were initially tested using the S10 transformation. The scale matching step described in Section 2.3 requires computing a PC analysis of the local descriptors (SSSIs for the proposed method and Self-Similar Normals for the baseline) for all extracted features over multiple scales. This yields a point cloud's keyscale which is defined by the authors in [11] as "*the scale that gives the minimum of cumulative contribution rate of PCA at a specific dimension of eigenspace*".

However, initial experiments using the model point clouds (Figure 7 and Figure 8) showed that using the minimum to define the keyscale proved valid only for the baseline algorithm. For the proposed algorithm using SSSIs the maximum needed to be computed.

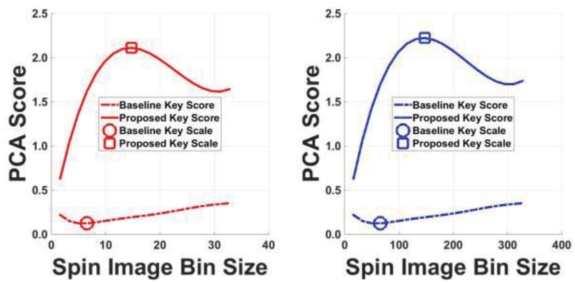

Figure 7: The Ant PCA scores (left) show the baseline keyscale at 6.55 and the proposed keyscale at 14.74 while the respective keyscales for Ant (S10) PC scores (right) were 65.5 and 147.4.

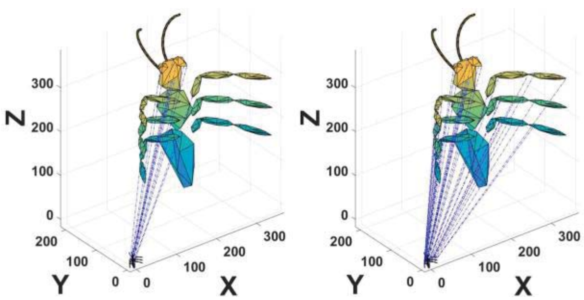

Figure 8: The scale difference between the point cloud (left) and its scaled version (right) was detected by both methods.

Therefore, a more general definition for a point cloud's keyscale would be "*the scale that gives the extremum of cumulative contribution rate of PCA at a specific dimension of eigenspace*".

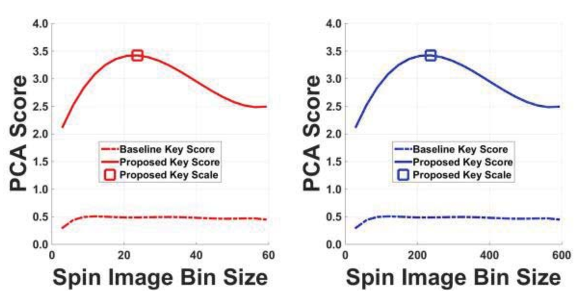

Figure 9: The ratio of the keyscale for the proposed method (23.66 to 236.6) curves yields the correct scale of 10. The baseline method did not yield a keyscale.

Initial experiments for the LIDAR point clouds showed similar results for the proposed algorithm as shown in Figure 9 and Figure 10. However, for the baseline algorithm the PC analysis curves did not have any local minimums regardless of scales selected and, therefore, a keyscale could not be found.

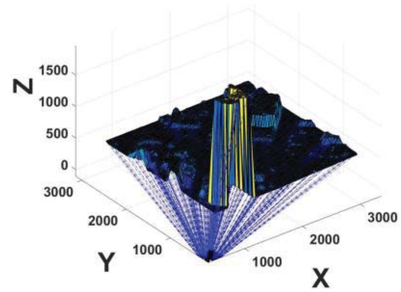

Figure 10: The scale difference between the point cloud and its scaled version was detected only by the proposed method.

Table 2 present the results of performing scale matching on the test dataset. For the simpler model point clouds both the baseline and proposed methods performed well, with the proposed method providing a slightly better estimate of the scale mismatch. For the realistic LIDAR point clouds the baseline method was unable to estimate the scale for any of the point clouds since there was no local minimum in their keyscale curves regardless of the scale ranges tested.

Table 2: Results of performing the scale matching tests.

Point Cloud	Baseline	Proposed	Error	Improve-ment (%)
Ant	10	10	0	0
Beethoven	9.41	10	0.59	5.9
Chopper	10	10	0	0
SMALL	N/A	10	N/A	N/A
LARGE	N/A	10	N/A	N/A
MIX	N/A	10	N/A	N/A

While the initial tests for the baseline method proved it unable to detect the scale difference between the reference and target LIDAR point clouds, tests with the robustness test set showed it able to detect scale difference of some but not all datasets.

Table 3: Results of scale matching on the robustness dataset.

Case	Algorithm	Truth	Avg	Std	Error (%)
S10	Proposed	10	9.98	0.11	0.21
S10	Baseline	10	10.06	0.25	0.64
Y180	Proposed	1	1.01	0.03	0.51
Y180	Baseline	1	0.98	0.07	1.64
RST	Proposed	2	1.96	0.24	1.86
RST	Baseline	2	1.71	0.41	14.41

Finally, in order to evaluate and compare the performance of the proposed and baseline methods for a wide range of terrains and misalignment transformations they were tested against the robustness test set. The results presented in Table 3 show both the baseline and proposed did a good job of estimating the scale for the simplest transformations S10 and Y180 with errors ranging from 0.21% to 1.64%. However, in both cases the proposed method still reduced the error over that of the baseline by 67.42% for S10 and 68.83% for Y180. For the more complex RST transformation the proposed method, with an error of 1.86%, outperformed the baseline method, with an error of 14.41%, again reducing the error by 87.10%.

3.4 Feature Matching

For the feature matching experiments the test point clouds were tested using the RST transformation. The feature matching step described in Section 2 requires detecting feature points (using the MoPC method), computing a local descriptor for all extracted features (SSSIs for the proposed method and Self-Similar Normal for the baseline), and matching them (using the Nearest Neighbor Distance Ratio method). The baseline and proposed methods were evaluated by comparing their accuracy in estimating the transformation between the reference and target point cloud.

Figure 11 shows the extracted feature points for one of the LIDAR point clouds and its transformed point cloud that were matched.

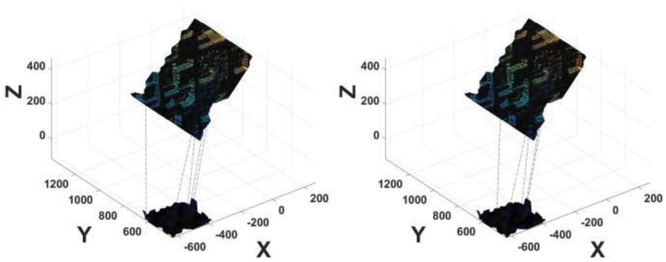

Figure 11: The point cloud and its RST version were matched using the the baseline (left) and proposed method (right).

Table 4: Estimated transformation parameters using both methods for the SMALL LIDAR point cloud.

Parameter	Truth	Baseline	Error (%)	Proposed	Error (%)
Angle	-45	-44.83	-0.37	-44.81	-0.42
X-axis	0	-0.004	-	-0.01	-
Y-axis	1	0.99	-0.02	0.99	-0.03
Z-axis	0	0.02	-	0.02	-
Scale	2.0	1.78	-11.1	2.0	0
X-offset	-249	-322.87	29.67	-255.88	2.76
Y-offset	-249	-224.42	-9.87	-259.35	4.15
Z-offset	-249	-340.34	36.68	-255.75	2.71
Error	-	-	27.94	-	3.29

Table 4 provides the transformation parameters estimated from the matched feature points as well as the true parameter values. The results show that the proposed method was able to provide a better estimate (total error of 3.29%) for the rotation, scale, and translation parameters than the baseline method (total error of 27.94%).

Table 5 presents the total registration error of performing feature matching on the test dataset. For both the simpler model and LIDAR point clouds the proposed methods outperformed the baseline method providing a significantly better estimate of the transformation mismatch with a median improvement of 88%.

Table 5: Total registration error of feature matching.

Point Cloud	Baseline	Proposed	Improvement (%)
Ant	5.11	4.97	-2.79
Beethoven	15.53	0.00	-100.00
Chopper	80.33	27.90	-65.27
SMALL	27.94	3.29	-88.23
LARGE	54.87	6.67	-87.84
MIX	28.88	0.61	-97.89

Finally, in order to evaluate and compare the performance of the proposed and baseline methods for a wide range of terrains and misalignment transformations they were tested against the robustness test set. The results presented in Table 6 show both the baseline and proposed methods did a very good job of estimating the misalignment transformation for the simplest transformations, S10 and Y180, with trivial errors ranging from 2.6E-11% to 1.7E-10%. For the more complex RST transformation the proposed method, with an error of 18.92%, outperformed the baseline method, with an error of 33.23%, reducing the error by 43.08%.

Table 6: Registration errors (%) matching the robustness sets.

Case	Algorithm	Error	Stdev	Improvement
S10	Proposed	1.7E-10	5.5E-11	-4.39
S10	Baseline	1.6E-10	5.4E-11	-
Y180	Proposed	2.6E-11	2.8E-11	9.74
Y180	Baseline	2.9E-11	3.1E-11	-
RST	Proposed	18.92	17.59	43.08
RST	Baseline	33.23	29.94	-

In conclusion, the Scale Matching experiments showed the proposed method outperformed the baseline method by reducing the error in detecting the scale difference by a factor of 67% to 87%. The Feature Matching experiments showed the proposed method outperformed the baseline method by reducing the alignment error by up to 43%.

4 Conclusions

Given the need to mine large 3D point clouds in a variety of fields, this study developed a promising method to align two point clouds with spatial and scale misalignments that only requires processing a subset of the data. To overcome many of the common problems in matching point clouds, this study proposed using a self-similarity based feature descriptor as the basis of its matching process because it can identify local patterns regardless of how they are generated. Combining this property with a powerful local metric like spin images leads to a robust new feature descriptor: Self-Similar Spin Images.

The distribution of these new descriptors was then analyzed in order to match scale differences between point clouds by defining a Self-Similar Keyscale for each cloud. For both feature and scale matching the proposed methods were able to accurately match the features and scales of the test cases over a variety of point cloud types. In particular, the proposed methods were able to outperform their baseline counterparts by exploiting a more powerful spin image descriptor within the self-similarity framework.

Future work on the SSSI method can incorporate additional metrics into the similarity descriptor in addition to spin images. Potential candidates for additional metrics are the local normal, the local curvature, and photometric intensity. Once new metrics have been added and a new descriptor defined then this will also yield a new SSK.

5 Appendix

The model point clouds are available online at: http://people.sc.fsu.edu/~jburkardt/data/ply/ply.html.

The LIDAR point clouds are available online at: https://earthexplorer.usgs.gov.

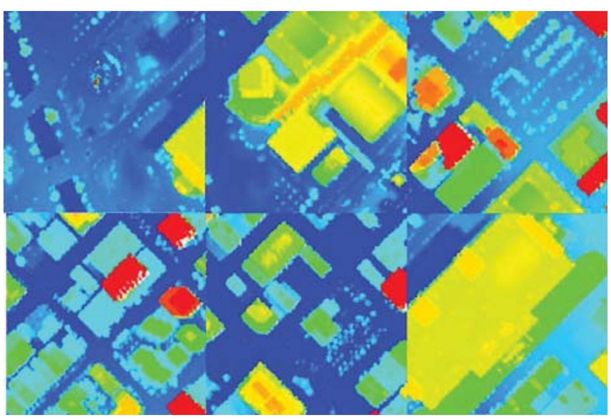

Figure 12: A sample of 6 of the 25 LIDAR point clouds used for robustness testing. The data sets were selected from an urban area in Denver that covered a variety of terrains.

6 Acknowledgement

The first author would like to acknowledge the funding provided by his employer Leidos Inc. that made this dissertation research possible.

7 References

[1] Shechtman, E.; Irani, M. Matching Local Self-Similarities across Images and Videos. IEEE Conference on Computer Vision and Pattern Recognition 2007.

[2] Chatfield, K.; Philbin, J.; Zisserman, A. Efficient Retrieval of Deformable Shape Classes using Local Self-Similarities. Workshop on Non-rigid Shape Analysis and Deformable Image Alignment, ICCV 2009.

[3] Maver, J. Self-Similarity and Points of Interest. IEEE Trans. Pattern Anal. Mach. Intell. 2010, 32, 1211-1226.

[4] Huang, J.; You, S.; Zhao, J. Multimodal image matching using self similarity. IEEE AIPR 20, 1-6.

[5] Huang, J.; You, S. Point cloud matching based on 3D self-similarity. IEEE CVPR Workshops 2012, 41-48.

[6] Stefanidis, A. Using scale space techniques to eliminate scale differences across images. Ohio State University 1993.

[7] Dufournaud, Y.; Schmid, C.; Horaud, R. Image matching with scale adjustment. Computer Vision and Image Understanding 2004, 2, 175-194.

[8] Khaleghi, B.; Baklouti, M.; Karray, F. SILT: Scale-invariant line transform. IEEE CIRA 2009, 78-83.

[9] Darom, T.; Keller, Y. Scale-Invariant Features for 3-D Mesh Models. IEEE Transactions on Image Processing 2012, 21, 2758-2769.

[10] Lu, M.; Guo, Y.; Zhang, J.; Ma, Y.; Lei, Y. Recognizing Objects in 3D Point Clouds with Multi-Scale Local Features. Sensors 2014, 14, 24156.

[11] Tamaki, T.; Tanigawa, S.; Ueno, Y.; Raytchev, B.; Kaneda, K. Scale Matching of 3D Point Clouds by Finding Keyscales with Spin Images. ICPR 2010, 3480-3483.

[12] Mikolajczyk, K.; Schmid, C. A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence 2005, 10, 1615-1630.

[13] Bronstein, A.; Bronstein, M.; Bustos, B.; Castellani, U.; Crisani, M.; Falcidieno, B.; Guibas, L.; Kokkinos, I.; Murino, V.; Sipiran, I.; Ovsjanikovy, M; Patane, G.; Spagnuolo, M.; Sun, J. SHREC 2010: robust feature detection and description benchmark. Eurographics 2010 Workshop on 3D Object Retrieval (3DOR'10) 20, 79-86.

[14] Boyer, E.; Bronstein, A.M.; Bronstein, M.M.; Bustos, B.; Darom, T.; Horaud, R.; Hotz, I.; Keller, Y.; Keustermans, J.; Kovnatsky, A.; Litman, R.; Reininghaus, J.; Sipiran, I.; Smeets, D.; Suetens, P.; Vandermeulen, D.; Zaharescu, A.; Zobel, V. SHREC 2011: robust feature detection and description benchmark. CoRR 2011.