Improved feature descriptors for 3-D surface matching

Luke J. Skelly[a] and Stan Sclaroff[b]

[a]MIT Lincoln Laboratory, 244 Wood Street, Lexington, Massachusetts, USA;
[b]Dept. of Computer Science, Boston University, Boston, Massachusetts, USA

ABSTRACT

Our interest is in data registration, object recognition and object tracking using 3D point clouds. There are three steps to our feature matching system: detection, description and matching. Our focus will be on the feature description step. We describe new rotation invariant 3D feature descriptors that utilize techniques from the successful 2D SIFT descriptors. We experiment with a variety of synthetic and real data to show how well our newly developed descriptors perform relative to a commonly used 3D descriptor, spin images. Our results show that our descriptors are more distinct than spin images while remaining rotation and translation invariant. The improvement in performance in comparison to spin images is most evident when an object has features that are mirror images of each other, due to symmetry.

Keywords: range image, object recognition, 3D surface matching, feature detection, rotation invariant, surface registration, spin-images, SIFT

1. INTRODUCTION

Our interest is in data registration, object recognition and object tracking using 3D point clouds. To achieve the aforementioned tasks we solve the correspondence problem by matching features that two differing images have in common. There are three steps to the feature matching process: detection, description and matching. Although all three steps are important, our focus is on feature description. The descriptions of these features must be distinctive enough to filter out wrong matches while remaining robust to changes that occur between images, including rotation, translation, and noise.

We experiment with a variety of synthetic and real data to show how well our newly developed descriptor performs relative to a commonly used descriptor, spin images.[1] Results show that our descriptor is more distinctive than spin images while remaining rotation and translation invariant. The improvement in performance is most evident when an object has features that are mirror images of each other due to symmetry.

2. RELATED WORK

Spin images[1] are perhaps the most widely used rotation invariant 3D feature descriptors. The method of computing a spin image is to relate the neighboring points of a feature to the normal vector of the feature. The three dimensional information is then recorded into a 2D histogram. The coordinates of this histogram are distance along the feature's normal vector and radial distance from the normal vector, with the feature location as the origin. The problem with spin images is that information is lost in the projection to two dimensions. This loss of information leads to an inability to discriminate between features that otherwise should be considered different. An example would be two features that are mirror images of each other.

The pairwise geometric histogram[2] is very similar to the spin image in that it employs a 2D histogram. However, the coordinate system of the histogram is slightly different. The abscissa is distance along the feature's normal vector, just as in the spin image, but the ordinate is the absolute angle difference between feature's normal vector and the normal vector of the neighborhood points being accumulated. Using only the absolute angle difference causes an inability to differentiate between convex and concave features.

Another interesting 3D descriptor is tensor based.[3] A 3D tensor is created by recording a 3D histogram of intersecting surface areas. The goal is to improve on spin images by not reducing the dimensionality of the surface being described. Rotation invariance along all three dimensions is incomplete due to coordinate system instabilities.

Extended Gaussian Images (EGI) have also been used to describe surfaces.[4] EGIs are not rotation invariant, so a search in three dimensions is required to find the best match for each possible pairing of features; this is computationally expensive.

We seek to improve on the spin image descriptor. Our inspiration is the widely used 2D descriptor, the Scale Invariant Feature Transform (SIFT).[5] The SIFT descriptor is a 3D histogram of image intensity gradients made to be rotation invariant by recording each gradient relative to the most dominant gradient. The SIFT descriptor has been shown to be invariant to in-plane rotation and robust to out-of-plane rotation. Our descriptor makes use of the spin image basis, normal vectors of each feature, but we intentionally avoid the 2D projection by using the rotation invariance technique from SIFT. Our use of this technique has led us to name our method Rotation Invariant Feature Transform or RIFT.

3. OVERVIEW OF APPROACH

First, we create an oriented point cloud, which is composed of a set of points with an orientation.[1] The orientation of a point is determined by its unit normal vector. We estimate the normal vectors by fitting a plane to the local neighborhood. Feature points are then selected from the oriented point cloud using a feature detector.[6]

The neighboring points of each selected feature point are first transformed into a rotation invariant coordinate system, which we refer to as RIFT. To achieve rotation invariance, the position and orientation of each neighboring point is measured relative to a neighborhood determined coordinate system, rather than the true or world coordinate system. This local neighborhood coordinate system is determined by the local surface gradients, using a technique from SIFT.[5]

Once the feature neighborhood is made translation and rotation invariant using RIFT, we then encapsulate it into a feature descriptor. We have developed and evaluated a number of different variants of our RIFT descriptor formulation.[6] In this paper, we describe the RIFT descriptor formulation that offers the most consistently superior performance: RIFT2-M.

Once this feature description process has been completed for two images, we then find correspondence by matching feature descriptors from each set. Since all the feature descriptors we refer to are translation and rotation invariant, the matching process requires no knowledge of this information. We utilize a greedy feature matching process similar to the one used by SIFT.

Once we have a set of correspondences, represented by our feature pairs, we can compute a transformation to align the features of the two sets. This transformation can be used to align the entire dataset for data registration, object recognition or object tracking purposes.

4. FORMULATION

The focus of our work is on feature description and due to space constraints we have chosen to only describe this step in detail. Implementation details of normal vector estimation, feature detection and matching steps are available.[6]

Algorithm 1 outlines the basic steps of RIFT and feature description for a single feature neighborhood. The input is an oriented point cloud and feature index F. First, we remove translation dependence by using the feature location as the origin of our coordinate system. We then determine which points are part of the feature neighborhood using a spherical constraint with radius in units of mesh resolutions[1],[6] which approximates the density of the point cloud. We then compute the rotation matrices that will remove the rotation dependence using Algorithm 3. Each resulting rotation matrix represents a potentially rotation invariant coordinate transform. We refer to the translation and rotation of the feature neighborhood as RIFT. Finally, for each rotation matrix, we create a feature descriptor using the RIFT2-M descriptor storage process described by Algorithm 4.

Algorithm 1: RIFT transformation and feature description

```
$\mathbf{p}$ : Oriented 3D Point Cloud, with locations ($\mathbf{p}$), normal vectors ($\mathbf{n}$) and feature index $F$
$\mathbf{D}$: RIFT feature descriptors, $D$
$L \leftarrow$ all locations that are within neighborhood of $\mathbf{p_F}$;
foreach element $i$ of $L$ do
    Compute $\mathbf{p}'_i \leftarrow \mathbf{p}_i - \mathbf{p}_F$;
end
$\mathbf{R} \leftarrow$ FindRiftRotations($p'$,$n$);
foreach element $j$ of $\mathbf{R}$ do
    foreach element $i$ of $L$ do
        Compute $\alpha_i$ and $\beta_i$ via Eq. 7;
        $\mathbf{n}_i^{\alpha} \leftarrow \mathbf{n}_i\mathbf{R}_j$;
    end
    $D_j \leftarrow$ RIFT2M($\alpha$,$\beta$,$\mathbf{n}^{\alpha}$)
end
```

4.1 The Rotation Invariant Feature Transform (RIFT)

Our work utilizes techniques from SIFT,[5] which is a complete feature matching process for 2D images that includes translation, rotation and scale invariance. The feature detection process and the scale invariance is of little interest to us. However, the rotation invariance is of particular interest. For each feature a 1D histogram of gradient angles is created. The peaks of this histogram are detected using an algorithm very similar to Algorithm 2. For each peak detected in this histogram a descriptor is created. Each descriptor is made relative to these peaks in order to achieve rotation invariance.

For SIFT, the rotation invariance is accomplished by finding all dominant intensity gradient angles along the 2D image plane. Our goal is to find a new 2D image plane. The most likely candidate is the surface tangent plane; described and approximated by the feature point location and its normal vector. If we project the data points to this plane, we have our 2D coordinate system that can be treated the same way as a 2D image. Note that we do not need to throw away the information lost in this projection; it is simply used as an analogy to the 2D image plane used in SIFT.

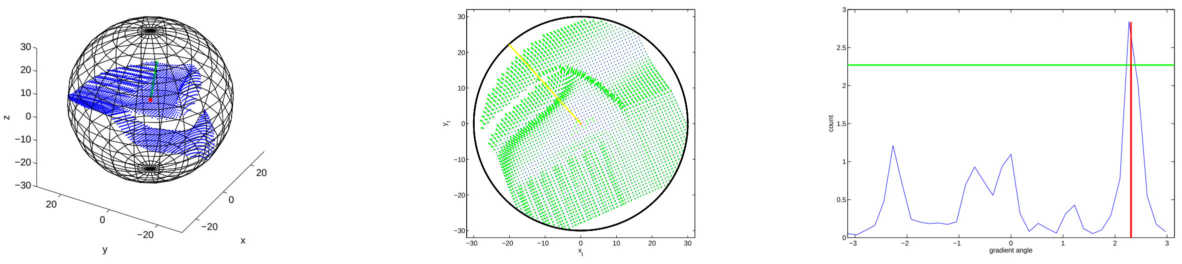
To compute the 2D gradients a temporary coordinate system is needed. Given the feature's normal vector, the other two axes of the orthogonal coordinate system can be obtained via cross products with an arbitrary vector that lies in the tangent plane, or a simpler approach is to choose an arbitrary vector that is not collinear with the normal vector. We find the basis vectors $\hat{\mathbf{x_t}}$ and $\hat{\mathbf{y_t}}$ of the tangent plane coordinate system by computing

$$\hat{\mathbf{x_t}} = \frac{\mathbf{n}_G \times \mathbf{n}_F}{\|\mathbf{n}_G \times \mathbf{n}_F\|} \text{ and } \qquad \hat{\mathbf{y_t}} = \frac{\mathbf{n}_F \times \hat{\mathbf{x_t}}}{\|\mathbf{n}_F \times \hat{\mathbf{x_t}}\|} \tag{1}$$

where $\mathbf{n}_G$ is the arbitrary vector, and $\mathbf{n}_F$ is the unit normal vector of the feature point. We will also refer to the

Algorithm 2: ComputeDominantAngles() finds the most dominant values in a 1D gradient angle histogram. Multiple values can be returned if they are not adjacent.

```
input  : Histogram, $H$
output: list of dominant angles, P
$m \leftarrow max(H)$;
foreach bin $i$ in $H|H_i > 0.8 \cdot m$ do
    $p \leftarrow \frac{H_{i-1}\cdot(i-1)+H_i\cdot i+H_{i+1}\cdot(i+1)}{H_{i-1}+H_i+H_{i+1}}$;
    Add $p$ to set $P$;
end
```



(a) Feature neighborhood (b) Gradients in temporary coordinates (c) Gradient Histogram

Figure 1. In (a) the points of a spherical neighborhood are shown in blue and the feature location in red. Image (b) shows the same neighborhood after a transformation to a temporary coordinate system, with gradient projections shown in green, and the most dominant angle detected is shown in yellow. The graph (c) shows the gradient histogram created from the gradients in (b). The peak location in the gradient histogram is shown in red and will be used as a reference angle for RIFT.

vector $\mathbf{n}_F$ as $\hat{\mathbf{z}}_t$. Then we create a rotation matrix

$$\mathbf{R_t} = \left[\hat{\mathbf{x_t}}, \hat{\mathbf{y_t}}, \hat{\mathbf{z}}_t\right], \tag{2}$$

where $\hat{\mathbf{x_t}}$, $\hat{\mathbf{y_t}}$ and $\hat{\mathbf{z}}_t$ are column vectors. The data in this new coordinate system can be found by multiplying the neighborhood point location relative to the feature point by our rotation matrix:

$$\mathbf{p}_i^t = (\mathbf{p}_i - \mathbf{p}_F)\mathbf{R_t}, \tag{3}$$

where $\mathbf{p}_F$ is the feature point location (our new origin). We also need to rotate the normal vectors associated with all the neighborhood points

$$\mathbf{n}_i^t = \mathbf{n}_i\mathbf{R}, \tag{4}$$

where $\mathbf{n}_i$ is the normal vector for the neighborhood point $\mathbf{i}$. We can now compute the feature neighborhood gradients, θ_i, and their corresponding magnitudes, mag_i

$$x_i = \hat{\mathbf{x}}_i \cdot \mathbf{d}_i, \qquad y_i = \hat{\mathbf{y}} \cdot \mathbf{d}_i, \qquad \theta_i = \arctan\frac{y_i}{x_i}, \qquad \text{and} \qquad mag_i = \sqrt{x_i^2 + y_i^2}. \tag{5}$$

Then we create a histogram of 36 bins covering all 360 degrees of rotation. The gradient magnitudes along with a Gaussian window are used as weightings to the histogram. We used a spherical Gaussian window, but a circular one along the tangent plane is also a possibility. The σ of the Gaussian window is 1.5 times the scale of the descriptor neighborhood size; this is same window size used for the SIFT descriptor. Linear interpolation is used to avoid boundary effects.[1] It should also be noted that the histogram is circular since we are dealing with angles that wrap around (360 degrees equals 0 degrees). The gradient histogram computed from the gradients shown in Figure 1(b) is shown in Figure 1(c). The highest peaks of the histogram are then extracted using Algorithm 2, which is based off of the algorithm from SIFT.[5] Each peak is used to create a unique descriptor. Most features will have only one peak and therefore only one descriptor, but others may have a few peaks and therefore multiple descriptors. For peak extraction, first the largest bin is found. All bins that are at least 80% power of this bin are also considered as peak candidates. This same 80% threshold was used in the SIFT descriptor formulation. The energy of a peak will often have energy split up into at least two bins. To avoid treating this as two different peaks, the bins that are peak candidates are grouped together if consecutive. Again, care must be taken to include the circular nature of the histogram. Then, the first moment of each group, and its closest bin from each side, is calculated in order to get a more precise measurement of the peak locations. Lowe uses a parabolic fit instead of the first moment.[5]

Algorithm 3: FindRiftRotations() computes the rotation matrices for RIFT. A rotation matrix will be computed for each dominant gradient angle.

```
p': Oriented 3D Point Cloud, with locations $\mathbf{p}'$ and normal vectors $\mathbf{n}$
R: Rotation Matrices, $\mathbf{R}$
Initialize $H$ to 0, a 1D array of gradient angle bins;
Compute rotation matrix $\mathbf{R}_t$ via Eq. 2;
foreach element i of point cloud neighborhood do
    Compute $\mathbf{p}_i^t$ via Eq. 3;
    Compute $\mathbf{n}_i^t$ via Eq. 4;
    Compute $mag_i$ and $\theta_i$ via Eq. 5;
    Distribute $mag_i$ to neighborhood of $H(\theta_i)$ using linear interpolation;
end
$\theta \leftarrow$ ComputeDominantAngles($H$);
foreach elementi$\in \theta$ do
    Compute $\mathbf{R}_(\alpha)$ for $\theta_i$;
    $\mathbf{R}_j \leftarrow \mathbf{R}_t\mathbf{R}_(\alpha)$;
end
foreach element i of point cloud do
    Compute $\alpha_i$ and $\beta_i$ via Eq. 7;
    $\mathbf{n}^\alpha \leftarrow \mathbf{n}'\mathbf{R}^\alpha$;
end
```

Once we have the most dominant gradient angles, we need to rotate the data locations, and their normal vectors, so that the most dominant gradient angle is pointing at 0 degrees. To do this we rotate about the $\mathbf{z}_t$ axis, but with the opposite polarity of the dominant angle by using Eq. 7, where

$$R_\alpha = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

and θ is the dominant angle, $[x_t, y_t, z_t]$ is the vector being rotated, and $[\alpha, \beta, \gamma]$ is the resulting vector. Our new coordinate system (α, β, γ) is now rotation invariant.

$$[\alpha, \beta, \gamma] = [x_t, y_t, z_t]\mathbf{R}_\alpha \tag{7}$$

4.2 The RIFT2-M feature description format

Once we have applied RIFT to the feature, we then form our descriptor, which is a small data structure that encapsulates the important feature information. Algorithm 4 describes the RIFT2-M descriptor format, which stores the feature neighborhood in a 2D array of 2D vectors that represent the average projected normal vector for the neighborhood points. For each neighborhood point, a bin is selected by the point's location and its orientation is then distributed using bilinear interpolation. Only the α and β components are used, while the γ component is discarded. To maximize information content in our descriptor, the bin size should be set to the radius of the feature's spherical neighborhood divided by the image size, multiplied by $\sqrt{2}$. One reason to compute an average normal vector is to avoid problems with nonuniform surface sampling. This difference in surface sampling density occurs because of its dependence on the angle of incidence between the surface and the sensor orientation. As long as the sampling density is not considerably smaller than our bin size, each RIFT2-M bin should contain a 2D normal vector representing an average surface gradient projected onto the feature tangent plane. Although its accuracy may depend on the number of samples per bin, the magnitude (in 3D) will remain constant. In contrast, a histogram based descriptor, such as the spin image will be effected by nonuniform surface sampling.

Algorithm 4: RIFT2M() encapsulates a feature neighborhood into a 2D array of 2D vectors. The oriented point cloud used as input is already translation and rotation invariant from the RIFT transform. The 2D vectors in the output array represent the average normal vector distribution for the neighborhood.

```
input  : Oriented Point Cloud, with locations $(\alpha, \beta)$, and normal vectors $(\mathbf{n})$
output: RIFT2-M Descriptor, $d$
Initialize $d$, a 2D array of 2D vectors, to 0;
Initialize $c$, a 2D array of scalars, to 0;
foreach element $i$ of point cloud do
    Compute $mag_i$ and $\theta_i$ via Eq. 5;
    Distribute $\mathbf{n_i}$ to neighborhood of $d(\alpha_i, \beta_i)$ using bilinear interpolation;
    Distribute 1 to neighborhood of $c(\alpha_i, \beta_i)$ using bilinear interpolation;
end
foreach element $j$ of $c$ do
    $d(\alpha_j, \beta_j) \leftarrow d(\alpha_j, \beta_j)/c(\alpha_j, \beta_j)$
end
```

5. RESULTS

For performance testing we use synthetic data from the University of Stuttgart[7] and real data collected from the Jigsaw program.[8] We compare the matching performance of our RIFT descriptor to that of spin images.

To compare our newly developed descriptor to spin images, we use the same feature detector and match filter parameters for both the real and synthetic data. The spherical neighborhoods for the feature detection process are all 5 mesh resolutions, including the normal vector estimator. For a fair comparison, we have kept the feature neighborhoods and bin sizes constant. The feature neighborhoods are 20 mesh resolutions. Both the spin images and RIFT are 8x8 bins each. As a reminder, RIFT2-M contains a 2D vector for each bin, whereas, spin images a single count. We have run other tests where RIFT is only 5x5 bins and found similar results.

In the matching process we attempt to filter out inappropriate matches using a uniqueness ratio,[6] which basically requires for each feature in image A, that the closest feature in image B is much better, or more unique, than all other features from image B. We intentionally use a very high uniqueness ratio due to the fact that what worked well for the synthetic datasets often turned out to be too conservative for the real data. We also felt more matches, whether correct or not, might provide additional insight into the pros and cons of each descriptor.

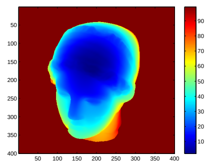
For each dataset, we plot the Euclidean distance error for a given match along the y-axis. The x-axis shows the matches in order of distance error (y value) so that better matches are plotted first. The x values are normalized by the number of matched features and are referred to as "sorted match %". The advantage of these sorted plots is that we can quickly find the best descriptor in each test depending on what is considered an acceptable amount of error.

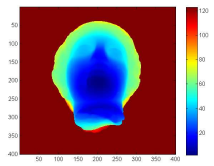
We also show the actual matches between the point clouds using three 2D projections. If matching were perfect, then all the connecting lines should be vertical since the point clouds in the figure are displayed in the same coordinate system.

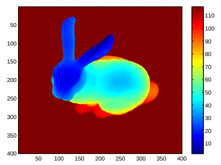
Additionally, we include tables, which list the number of points, number of detected features, and mesh resolution (point cloud density) for each image of each dataset. The mesh resolution is unitless for the synthetic datasets and in meters for the real datasets.

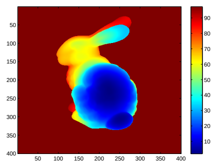
5.1 Synthetic datasets

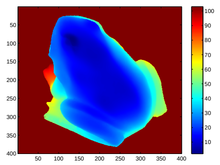
We chose five synthetic datasets that contain a pair of range images to work with. The Igea dataset is shown in Figures 2(a) and 2(b). It contains some mirror features, due to symmetry, such as eyes and ears. Most of the surfaces are smooth, but not flat. The Bunny dataset is shown in Figures 2(c) and 2(d). It contains very little symmetry due to the pose of the model and mostly smooth surfaces. There are also some interesting self-occlusions from the disparity of the bunny ears. The Toad dataset is shown in Figures 2(e) and 2(f), which has similar properties to the Bunny dataset, but with a more symmetrical pose. The Manta dataset, shown

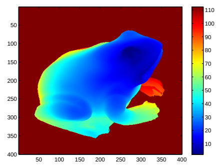

(a) Igea 1

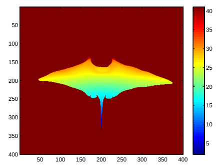

(b) Igea 2

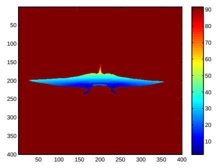

(c) Bunny 1

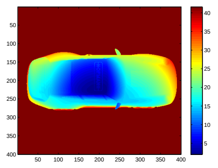

(d) Bunny 2


(e) Toad 1


(f) Toad 2


(g) Manta 1


(h) Manta 2


(i) Porsche 1

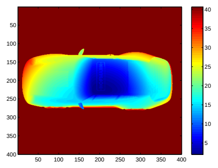
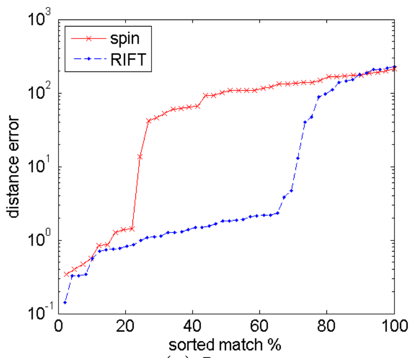
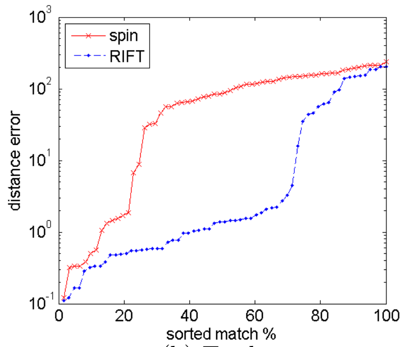

(j) Porsche 2

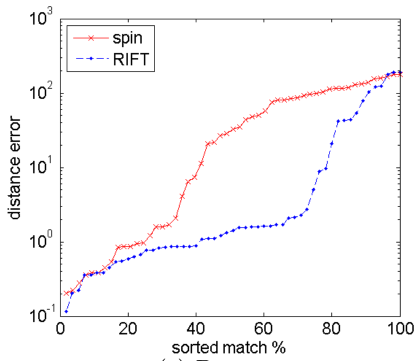
Figure 2. The synthetic datasets used to test our feature descriptor, shown as range maps. The colors represent the distance to the sensor.



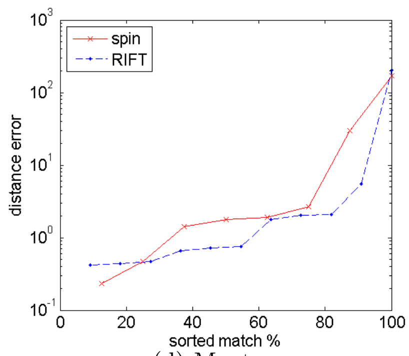
(a) Igea



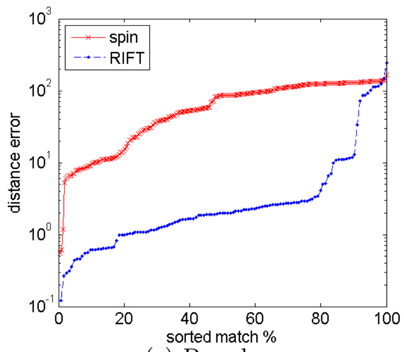
(b) Toad



(c) Bunny

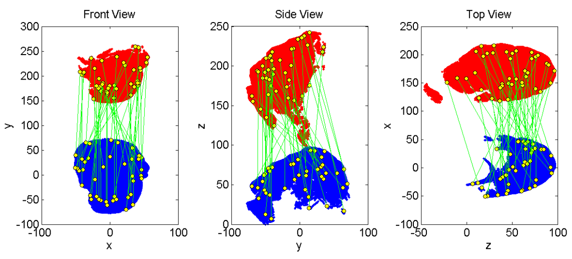


(d) Manta

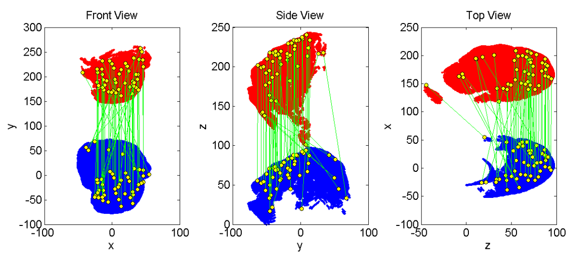


(e) Porsche

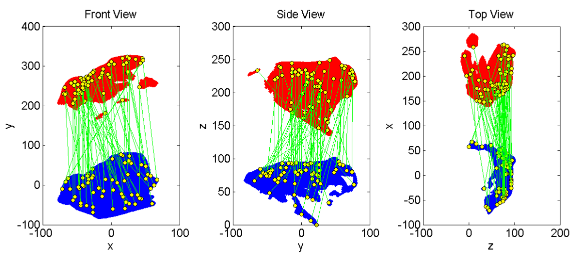
Figure 3. A comparison of RIFT2-M and spin images on the synthetic data sets. In each graph, the y axis is the Euclidean distance error in units of mesh resolution on a log scale. The matches are plotted in order of smallest error first and on the x-axis in terms of percentage of total matches.



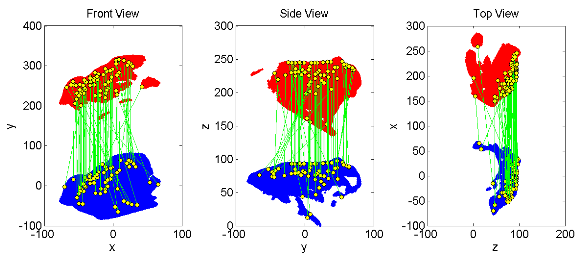
(a) Igea: spin images



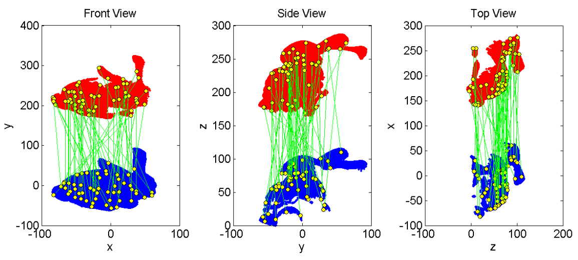
(b) Igea: RIFT2-M



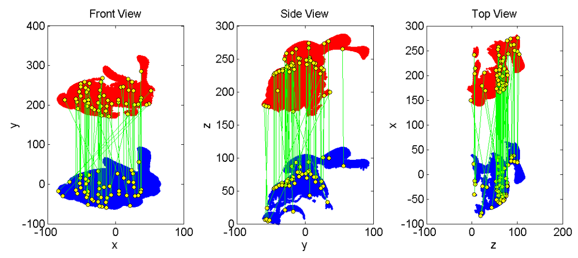
(c) Toad: spin images



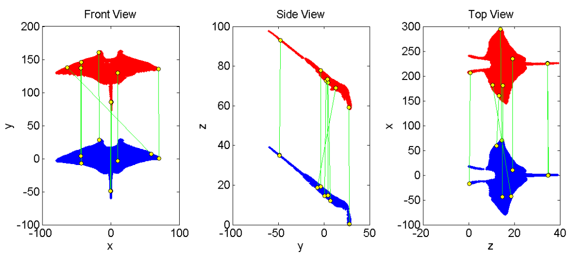
(d) Toad: RIFT2-M



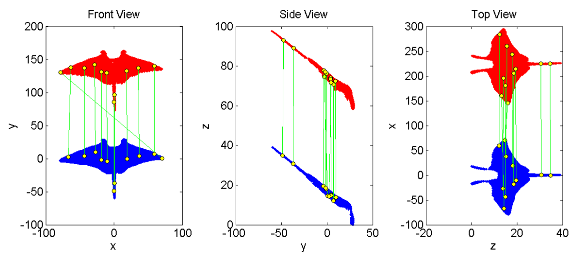
(e) Bunny: spin images



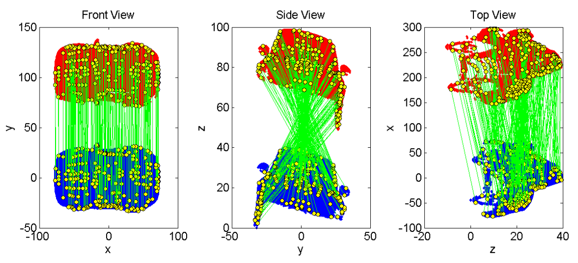
(f) Bunny: RIFT2-M



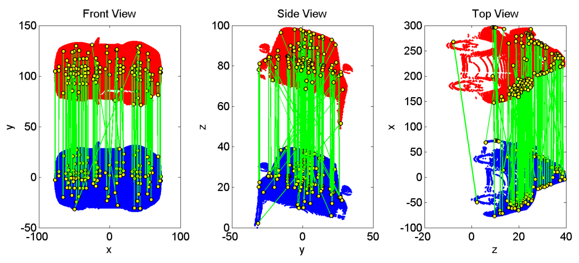
(g) Manta: spin images



(h) Manta: RIFT2-M



(i) Porsche: spin images



(j) Porsche: RIFT2-M

Figure 4. The matches for spin images and RIFT2-M on the synethetic datasets. Each of the three images are 2D projections onto axis pairs. The two point clouds, shown in red and blue are registered, but one is offset along one axis. All matching lines, shown in green, should be vertical. Units are not by mesh resolution.

Table 1. Properties of the synthetic data sets: # points is the number of points in the input dataset and the mesh resolution represents the point density, which is unitless for the synthetic datasets.

image	# points	# detected features	mesh resolution
Igea 1	8644	21	0.2378
Igea 2	9551	16	0.2302
Toad 1	71673	308	0.6131
Toad 2	63082	298	0.5899
Bunny 1	38085	233	.7479
Bunny 2	38087	220	.7573
Manta 1	17832	35	.4872
Manta 2	7299	42	.9709
Porsche 1	51116	320	.4521
Porsche 2	51149	319	.4526

in Figures 2(g) and 2(h), is a test to handle surfaces with a significant difference in sampling density due to the angle of incidence from the sensor. The last dataset of a Porsche, shown in Figures 2(i) and 2(j), contains a mixture of very sharp corners and smooth curves. There is a 180 degree rotation between the two images. There is a very high degree of symmetry in the Porsche and some repeated features, such as the wheels. Table 1 contains the properties of the synthetic datasets. The comparison of RIFT2-M and spin images in terms of Euclidean distance error for the synthetic datasets is shown in Figure 3. If we chose 5 mesh resolutions as our threshold of a good match, then RIFT2-M appears to have about a 70-80% success rate in every dataset, whereas spin images perform poorly (below 40%) with the exception of the Manta dataset shown in Figure 3(d). The dataset in which spin images perform the most poorly is the Porsche dataset shown in Figure 3(e). Figure 4 displays the matches along three 2D projections. We can see qualitatively that RIFT2-M is performing better than spin images in all of the synethetic datasets. In Figures 4(g) and 4(h) we can see that both RIFT2-M and spin images make one mistake due to symmetry along the x direction. However, either one of these could be due to features not being detected. The same cannot be said for the Porsche dataset shown in Figures 4(i) and 4(j) Spin images are consistently finding matches on the wrong side of the Porsche, whereas RIFT2-M is making this type of mistake at a far lower rate.

5.2 Real data

We also test our descriptor on real data from the Jigsaw system.[8] Figure 5 shows views of the data from the point of view of the sensor. The ground has been cropped out to reduce the large amount of mostly featureless information. The datapoints are colored according to elevation, not range to sensor. A table summarizing the properties of each dataset are shown in Table 2.

The Tank dataset, shown in Figures 5(a) and 5(b), is of a tank in the open. The turret is turned a bit, which reduces the symmetry of the image. A truck is shown in Figures 5(c) and 5(d). The trailer is basically a rectangular box with little or no interesting features, but the front of the truck contains some rich information. The last dataset, shown in Figures 5(e) and 5(f) is of a Humvee with some clutter surrounding it, including some electrical boxes and a fence. There is also some ground left in since it was uneven and removing it completely would have also removed part of the Humvee.

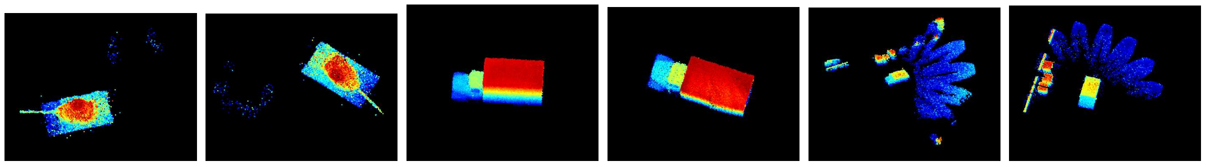
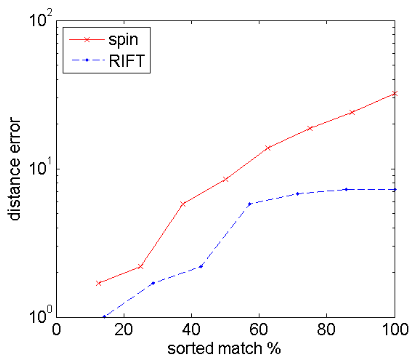

(a) Tank 1 (b) Tank 2 (c) Truck 1 (d) Truck 2 (e) Humvee 1 (f) Humvee 2

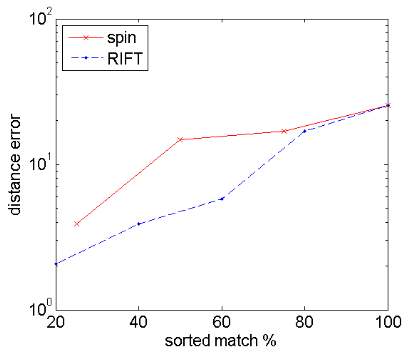
Figure 5. Real data taken from the JIGSAW system. The views are from the perspective of the sensor.

Table 2. Properties of the datasets taken from the Jigsaw system: # points is the number of points in the dataset, # features is the number of features that were detected, and mesh res is the mesh resolution[1],[6] which is in units of meters.

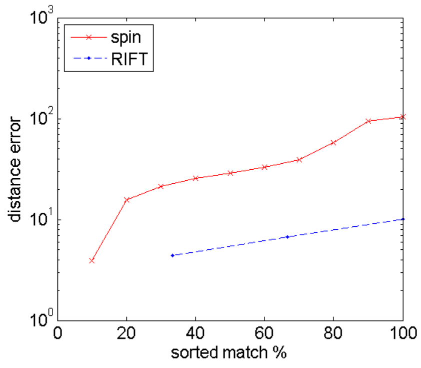
image	# points	# features	mesh res (m)
Tank 1	8644	21	0.2378
Tank 2	9551	16	0.2302
Truck 1	6982	24	0.0863
Truck 2	6818	23	0.0837
Humvee 1	27057	79	.2324
Humvee 2	23786	50	.1363



(a) Tank



(b) Truck



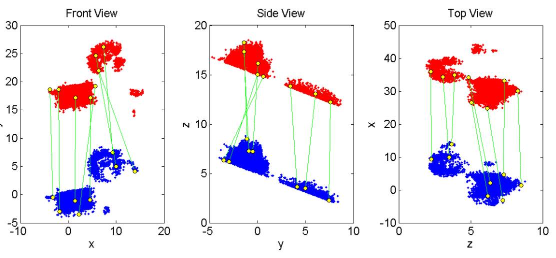
(c) Humvee

Figure 6. A comparison of RIFT2-M and spin images on the Jigsaw data sets. In each graph, the y axis is the Euclidean distance error in units of mesh resolution on a log scale. The matches are plotted in order of smallest error first and on the x-axis in terms of percentage of total matches.

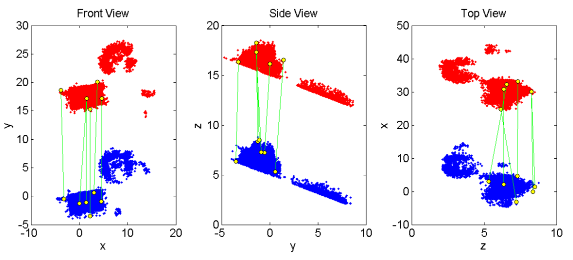
The comparison of RIFT2-M and spin images in terms of Euclidean distance error for the Jigsaw datasets is shown in Figure 6. If we chose 5 mesh resolutions as we did for the synthetic datasets, both RIFT2-M and spin images would have a less than 50% success rate. Despite that, we can still see that RIFT2-M is performing better. For the tank dataset, all 7 of the RIFT2-M matches are below or slightly above 7 mesh resolutions (about 1.7 meters) of error and in contrast, spin images find only 3 out of 8 matches below this error. More than 60% of the RIFT2-M matches in the Truck dataset are below 6 mesh resolutions (about 0.5 meters) of error, but only 1 out of 4 of the spin image matches are below this error. Finally, in the Humvee dataset, all 3 of the RIFT2-M matches are below 10 mesh resolutions (about 2.3 meters), but 90% of the spin image matches have more error. Figure 7 displays the matches along three 2D projections. In Figure 7(a), especially in the side view, we can see that spin images are finding matches on the wrong side of the tank. RIFT2-M appears to be making more mistakes along the z-axis, shown in the top view of Figure 7(b). Figure 7(d) shows that RIFT2-M has made a couple of big mistakes along the x-axis, but it at least the match is along the same surface of the truck. The spin image matches in Figure 7(c), however, seems to have matched the top of the truck to its side. The first thing to notice in Figures 7(e) and 7(f) is that RIFT2-M has only found a few matches, but they are very consistent and spin images finds many very poor matches along with some good ones. Both descriptors find only one match on the humvee itself.

6. DISCUSSION AND FUTURE WORK

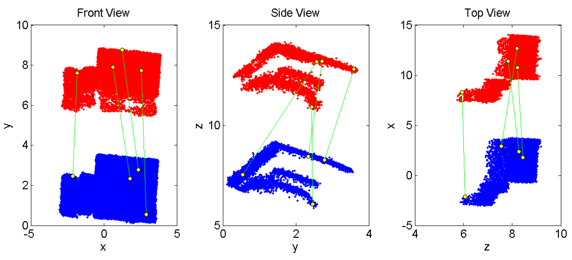
The results from the synthetic data suggest that RIFT is more successful at surface feature representation than spin images. The results of the Porsche dataset may provide evidence as to why. Figure 4(i), shows that spin images provide good matches along the x and z axes, but always incorrect matches along the y axis. This is due to the amount of symmetry in both the Porsche model and the camera views.



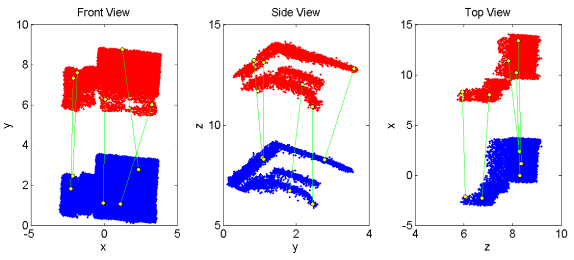
(a) Tank: spin images



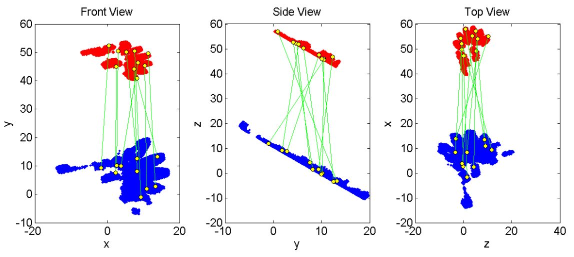
(b) Tank: RIFT2-M



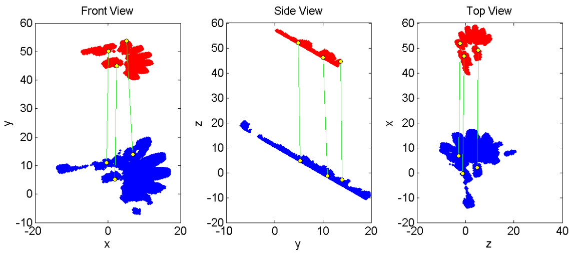
(c) Truck: spin images



(d) Truck: RIFT2-M



(e) Humvee: spin images



(f) Humvee: RIFT2-M

Figure 7. The matches for spin images and RIFT2-M on the Jigsaw datasets. Each of the three images are 2D projections onto axis pairs. The two point clouds, shown in red and blue are registered, but one is offset along one axis. All matching lines, shown in green, should be vertical. Units are in meters.

The results from the Manta dataset may seem unexpectedly good, since the dataset was chosen specifically to test disparate mesh resolutions between images. Spin images may be handling this problem with normalization since the Manta is nearly flat. We expect that spin images would not work quite as well, if the dataset contained sharp edges or corners.

The results for the real datasets show promise for RIFT; use of the RIFT descriptor leads to a significantly higher rate of correct matches than can be obtained via spin images. Not surprisingly, overall performance of RIFT on real data sets is diminished compared to the synthetic data. Moreover, the success with the real data was strongly dependent on the noise filtering, and more sensitive to feature detection parameters than the synthetic data. We believe that this suggests more effort should be spent on the feature detection and matching steps, and also on normal vector estimation for noisy data.

In the experiments with both real and synthetic data, we observed that RIFT appears to be more conservative with our greedy feature matching. We considered that spin images may behave similarly to RIFT with a larger uniqueness ratio.[6] However, we found that changing the uniqueness ratio did not affect the matching results significantly in our experiments.

Finally, it should be noted that our RIFT-based system should be easily adapted for object recognition and object tracking purposes. Other potential applications of interest include: detecting multiple copies of the same object in a single view, sensor localization, and object detection in very large and highly cluttered datasets.

ACKNOWLEDGMENTS

This work is sponsored by the United States Air Force under AF Contract # FA8721-05-C-0002. Opinions, interpretations, recommendations and conclusions are those of the authors and are not necessarily endorsed by the United States Government.

REFERENCES

1. A. Johnson, *Spin-Images: A Representation for 3-D Surface Matching.* PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, August 1997.
2. A. P. Ashbrook, R. B. Fisher, C. Robertson, and N. Werghi, "Finding surface correspondance for object recognition and registration using pairwise geometric histograms," in *ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume II*, pp. 674–686, Springer-Verlag, (London, UK), 1998.
3. A. S. Mian, M. Bennamoun, and R. A. Owens, "A novel representation and feature matching algorithm for automatic pairwise registration of range images," *International Journal of Computer Vision* **66**(1), pp. 19–40, 2006.
4. A. Makadia, A. I. Patterson, and K. Daniilidis, "Fully automatic registration of 3d point clouds," in *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1297–1304, IEEE Computer Society, (Washington, DC, USA), 2006.
5. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision* **60**(2), pp. 91–110, 2004.
6. L. Skelly, "Rotation invariant 3d feature description," Master's thesis, Boston University, Computer Science Department, 2007.
7. G. Hetzel, B. Leibe, P. Levi, and B. Schiele, "3d object recognition from range images using local feature histograms," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR'01)*, **2**, pp. 394–399, 2001.
8. R. M. Marino and J. William R. Davis, "Jigsaw: A foliage-penetrating 3d imaging laser radar system," *Lincoln Laboratory Journal* **15**(1), pp. 23–36, 2005.