“Spin Images”

Georgios Papadimitriou
e-mail: papadimgeorge@gmail.com

1. Spin Image Concept

The spin image is a surface representation technique that was initially introduced by Andrew E. Johnson in [1] and is used for surface matching and object recognition in 3-D scenes. Spin images encode the global properties of any surface in an object-oriented coordinate system rather than in a viewer-oriented coordinate system. Object-oriented coordinate systems are coordinate systems fixed on a surface or an object while viewer-oriented coordinate systems are based on the viewpoint of the observer of the surface. By using object-oriented coordinate systems, the description of a surface or an object is view-independent and it does not change as the viewpoint changes [3].

The key element of spin images generation is the use of oriented points [2]. Oriented points (denoted as O) are 3-D surface points which have an associated direction to them. The surface to which these points correspond is represented as a polygonal mesh M with vertices. An oriented point O at a surface mesh vertex is defined by the 3-D position of the surface vertex (denoted as p) and a surface normal (denoted as n) [2]. With p, n defined, we can formulate a 2-D basis (p, n) which will correspond to a local coordinate system. To achieve this we use the tangent plane P through p oriented perpendicularly to n and the line L through p parallel to n [2]. This will result in a (a, β) cylindrical coordinate system where a is the (non-negative) perpendicular distance to L while β is the signed (positive or negative) perpendicular distance to P [2]. To better understand the creation of the cylindrical coordinate system, see *Figure 1*.

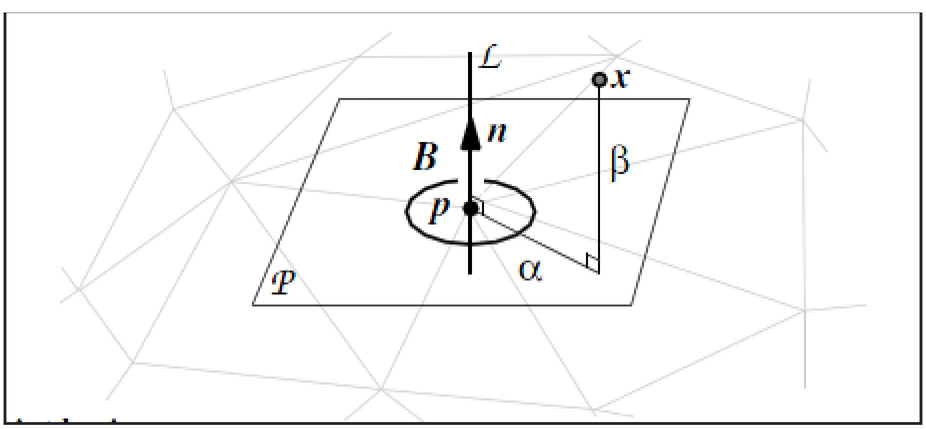

Figure 1: The cylindrical coordinate system and its (p, n) 2-D basis (Taken from [2]).

The oriented point basis is then used to generate a spin map, S_O. A spin map, S_O can be expressed as a projection function of the 3-D points x of an object to 2-D coordinates (a, β) associated with the 2-D basis (p, n) that corresponds to the oriented point O [2]. The projection function can be seen below [2]:

$$S_O: R^3 \rightarrow R^2$$

$$S_O(x) \rightarrow (a, \beta) = \left(\sqrt{\|x-p\|^2 - (n \cdot (x-p)^2}, n \cdot (x-p)\right) \quad (1)$$

To better understand the above procedure, recall that the surface of an object is represented as a mesh M with vertices. By applying the projection function (1) to all the vertices we create a set of points with (a, β) 2-D coordinates. Note that for a spin image to be created we only need one oriented point. The other 3-D points of the surface of the object that are expressed in 2-D coordinates are expressed with respect to that oriented point as you can see by examining function (1). The next step is to create a 2-D array that is going to represent the spin image generated using the 2-D basis. In order to achieve this we accumulate the 2-D points (a, β) into discrete bins (see equations (3) in the next section). Each time a 2-D point is accumulated into a bin we update the 2-D array by incrementing the surrounding bins in the table (see (4) in the next section) [2]. At this point we need to consider the noise in the data. So the contribution of the 2-D point is spread to the four surrounding bins of the 2-D array by using bilinear interpolation (http://en.wikipedia.org/wiki/Bilinear_interpolation) [2]. The creation of a 2-D array representation of a spin image can be seen in *Figure 2* and spin images that are created using three different oriented points O can be seen in *Figure 3*. Note that for a full representation of a surface we need to create spin images for all the oriented points that are defined by the surface mesh.

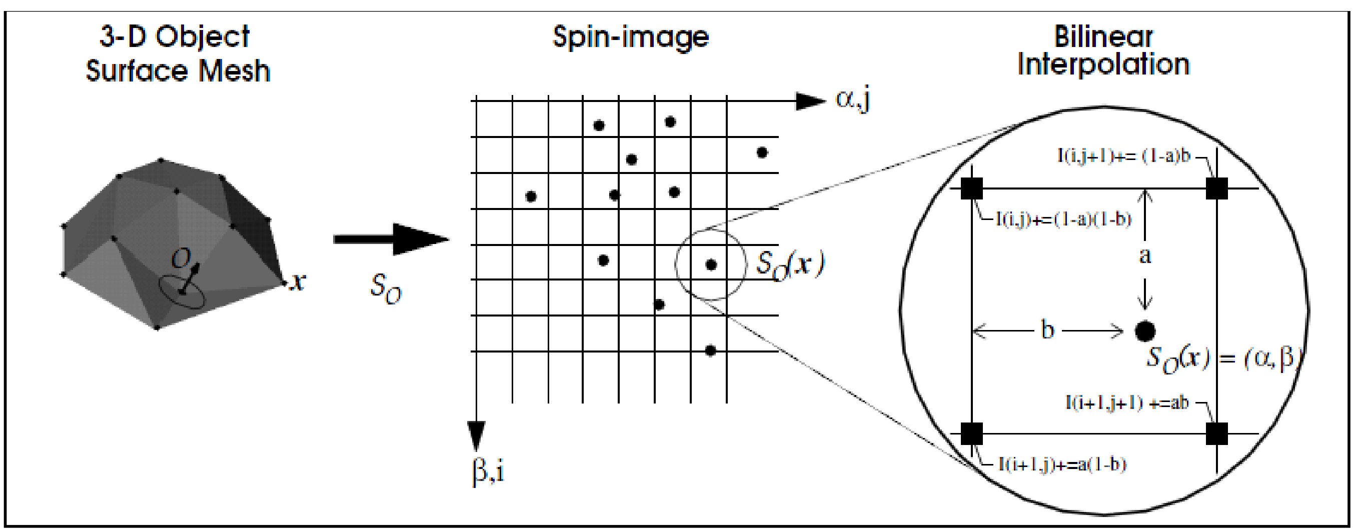

Figure 2: Creation of the 2-D array representation of a spin image (Taken from [2]).

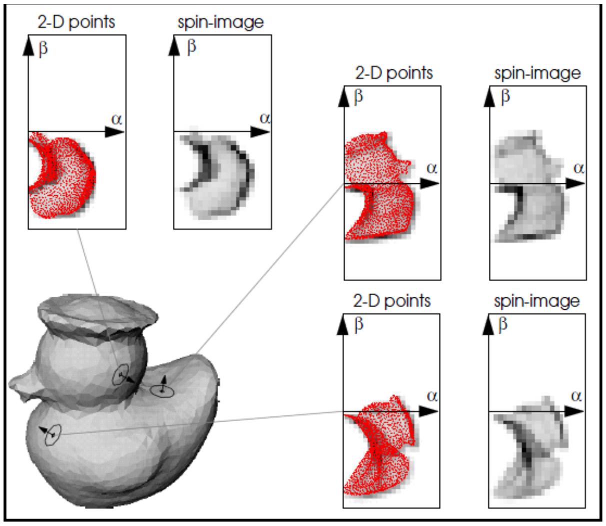

Figure 3: Spin image creation for three different oriented points in the surface of an object (Taken from [2]).

1.2 Spin Image generation parameters and properties

So far, we have introduced the spin image concept and how spin images are generated but we have not mentioned anything about their generation parameters that affect their properties (size, descriptiveness). The size of a spin image that represents an object depends on the bin size and the maximum size of that object expressed in oriented point coordinates (spin-map coordinates) [2]. To determine the maximum size of an object expressed in oriented point coordinates a_{max}, β_{max}, we need to find the maximum α and $|\beta|$ of all the different oriented point bases [2]. The bin size (denoted as b) must be such that the size of the resulting spin image is not very big while its descriptiveness of an object remains relatively high. Moreover the number of bins used in a spin image also affects its size.

The spin image size (i_{max}, j_{max}) can be calculated as follows [2]:

$$i_{max} = \frac{2\beta_{max}}{b} + 1 \qquad j_{max} = \frac{a_{max}}{b} + 1 \tag{2}$$

The way 2-D points (α, β) are accumulated into discrete bins can be seen in the following equations [2]:

$$i = \left\lfloor \frac{\beta_{max} - \beta}{b} \right\rfloor \qquad j = \left\lfloor \frac{a}{b} \right\rfloor \tag{3}$$

The bilinear weights that are used to increment the bins are [2]:

$$\mathrm{a} = \alpha - ib \qquad b = \beta - jb \tag{4}$$

2. Using spin images for 3-D object recognition

Since spin images are used for surface representation it is evident that they can be used for surface matching and consequently for 3-D object recognition in different scenes. From now on, in order to explain how we can achieve object recognition, we will be referring to the objects that we want to recognize in a scene as models and to the set of models as model library [3]. Before launching the recognition procedure, we first need to generate spin images for all our models and store them [3]. To achieve this, we need to represent each model as a polygonal mesh M with vertices. At this point, we generate one spin image for each vertex in each model. As a result, we have a spin image representation for each point on a model surface and consequently a spin image representation for every model (the set of spin images that correspond to each model synthesize the representation of its surface as a whole). The next step is to select a random point (oriented point, O) from the scene, which is also represented using a polygonal mesh M, and generate its spin image. This spin image is then correlated with all the spin images of the model library [3].

For image correlation the following similarity function is applied [2]:

$$C(P,Q) = (\operatorname{atanh}(R(P,Q)))^2 - \lambda\left(\frac{1}{N-3}\right) \qquad (5)$$

where, P, Q are spin images, R is the correlation coefficient, N is the number of pixels that overlap and are used to compute R, $\left(\frac{1}{N-3}\right)$ is a simple function of the number of pixels that are used to compute R and expresses the variance of R while λ is a weight coefficient.

The higher the value of the similarity function, the larger the number of overlapping spin image bins and thus their correlation (similarity) [2]. The highest correlated model library spin image with the scene spin image will result in determining the best matching model and model vertex [3]. This procedure is repeated for a lot of scene spin images and in the end, point correspondences between the scene and the models are being established. Note that there is no unique number of point correspondences that indicates that we should stop. In the approach followed in [3], this procedure stops when approximately 100 point correspondences are established. This number can vary and it highly depends on the density of the surface mesh we used in the first place. The next step is to form groups of all these correspondences and reduce or even get rid of outliers by using geometric consistency [3]. These groups are then used for aligning surfaces with each other by calculating rigid transformations [3]. The final step includes surface matching verification using an ICP (Iterative Closest Point) algorithm (http://en.wikipedia.org/wiki/Iterative_Closest_Point). As a result, we achieve both recognition and localization of the models inside the scene (if any exists). The surface matching procedure can be seen in *Figure 4* while the results of 3-D object recognition can be seen in *Figure 6*. The models that are used in *Figure 6* are taken from the model library shown in *Figure 5*.

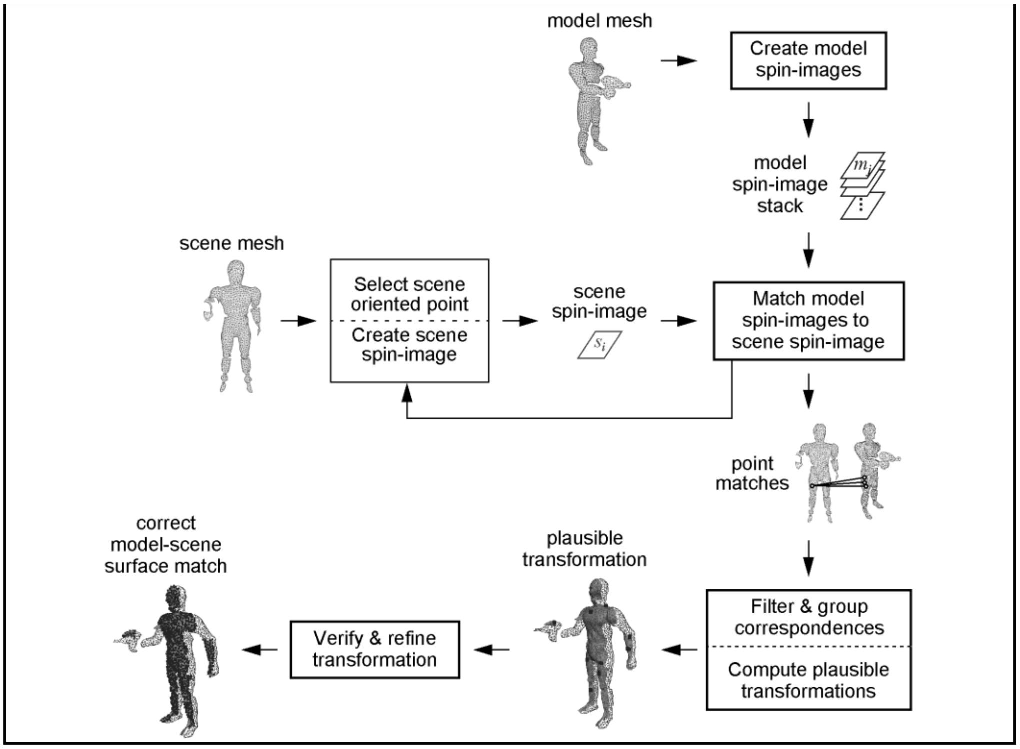

Figure 4: The surface matching procedure (Taken from [3]).

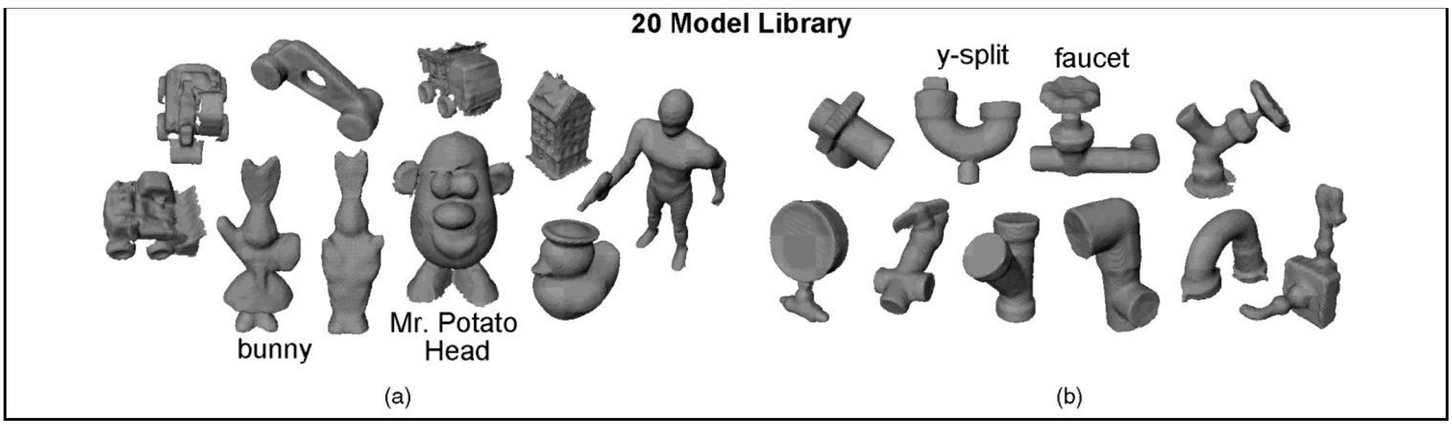

Figure 5: A typical model library with 20 models (Taken from [3]).

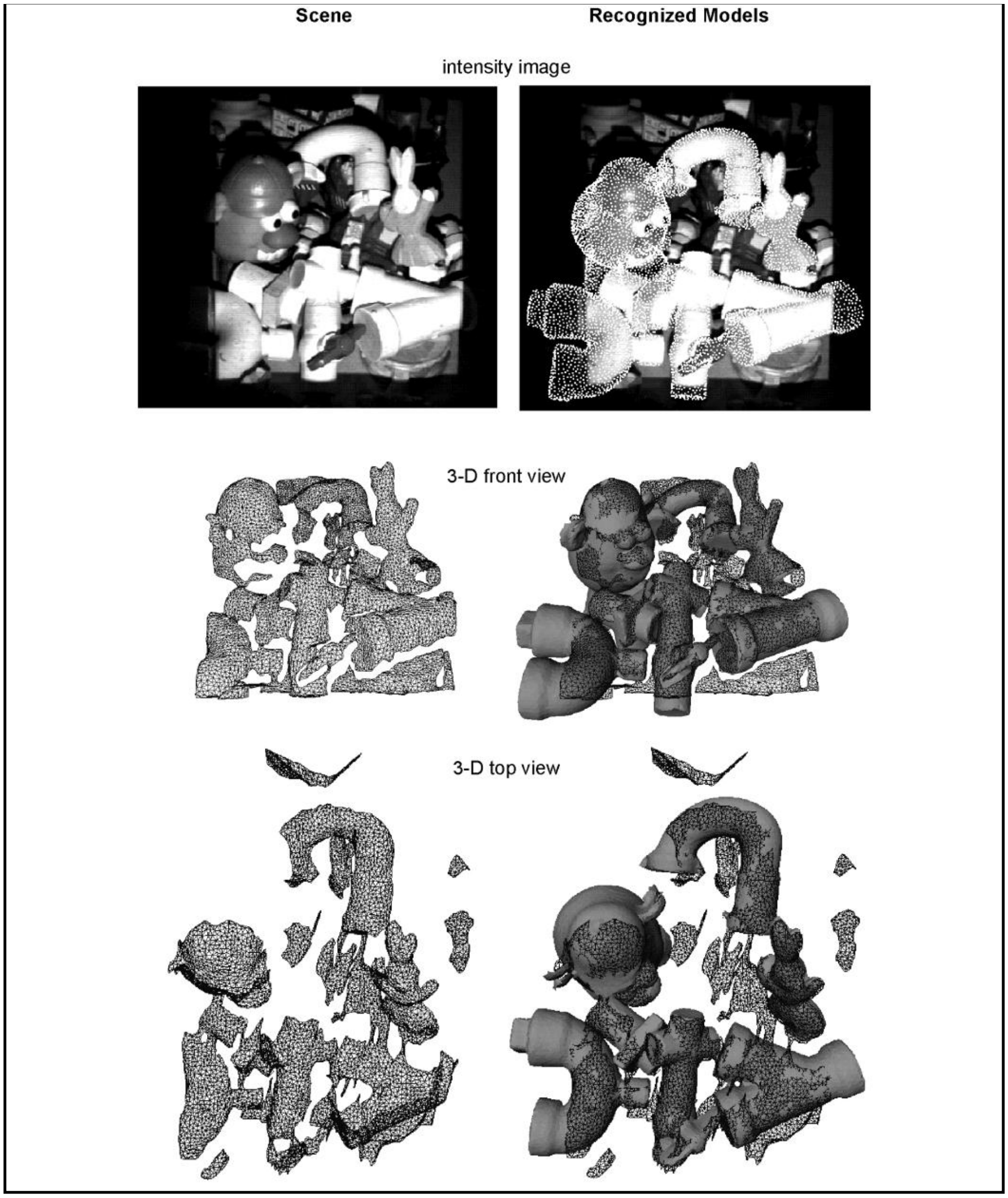

Figure 6: 3-D object recognition (Taken from [3]).

Note that in order to speed up the matching process we can compress spin images by using principal component analysis (PCA) [4]. Image compression is very important to 3-D object recognition when we are dealing with large model libraries since prior to a spin image being matched to a model in a model library it is correlated with all the spin images of the library. As a result, large model libraries lead to a linear growth of the matching procedure in terms of number of correlations [3] which consequently makes recognition of objects in a scene inefficient.

To conclude, we can say that using spin images for 3-D object recognition is very effective since the results, even in scenes that are cluttered and contain occlusion, signify high success rate.

References

[1] A. Johnson, "*Spin-images: a representation for 3-D surface matching",* Ph.D. Dissertation, The Robotics Institute, Carnegie Mellon University, August 1997.

[2] A. Johnson and M. Hebert, *"Surface Matching for Object Recognition in Complex Three-Dimensional Scenes"*, Image and Vision Computing, vol. 16, pp. 635-651, 1998.

[3] A. Johnson and M. Hebert, *"Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes",* IEEE transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 5, May 1999.

[4] H. Murase and S. Nayar, *"Visual Learning and Recognition of 3-D Objects From Appearance"*, International Journal of Computer Vision, vol. 14, pp. 5-24, 1995.