Generative Cognitive Representation for Embodied Agents

Anshul Joshi
School of Computing
University of Utah
Salt Lake City, UT 84112
Email: joshi@cs.utah.edu

Thomas C. Henderson
School of Computing
University of Utah
Salt Lake City, UT 84112
Email: tch@cs.utah.edu

***Abstract*—Symmetry (similarity under transformation) in sensorimotor data streams can be used to derive and represent concepts from percepts, and exploit them in a cognitive framework. This entails finding mechanisms to extract symmetries out of sensorimotor data and convert them into intuitive representations; "intuitive" implying that the representations created should have a semantic meaning to them. We design these concepts in an object-oriented framework. These concepts, known as wreath product comprise of a control group that acts on a fiber group, and are generative concepts in that they encode not only the symmetry information, but also the process employed to achieve those symmetries. Moreover, probabilistic information can be added to these wreath products in the form of *Bayesian networks* for quantifying uncertainty in the data and for inference and generalization, which would help the agent achieve tasks requiring cognitive abilities in the real world. In this paper we present the theory behind such a cognitive framework, propose and implement design of concepts from sensor data, and demonstrate this approach on data collected from Kinect.**

***Keywords*—Robot Cognition, Symmetry analysis, Wreath Product, Generative Representation.**

I. Introduction

Robots with cognitive behavioral requirements need an effective representation of the world they operate in. Vernon et al [8] define Cognition "as a process by which the system achieves robust, adaptive, anticipatory, autonomous behavior, entailing embodied perception and action." Adaptive, anticipatory and autonomous behavior implies that the robot should have capabilities to build its own knowledge base out of presence and interaction in its surroundings, and to build on previously acquired knowledge to achieve further autonomy (structural boostrapping). To achieve such behavior the robot needs innate theories for concept representation and perceptual fusion. This paper aims at addressing the problem of concept representation in cognitive robotics. In our case "knowledge" is a set of concepts, and each concept in turn is a *bundle* of sensorimotor data streams and correlated symmetries present in that data which represent using a wreath product as described by Leyton [4]. The wreath product encodes the symmetries present in the data as well as the process that generates those symmetries, and thus we recognize concepts as generative entities (see section III for details). These generative concepts are key factors to achieve sensor-motor coupling since they have an inherent sequence of actions associated with them that generates the symmetries in the first place! In our representation the perceptual data that satisfy certain group axioms are encapsulated in an object corresponding to a mathematical group which inherently has symmetry associated with it. To build these generative concepts we extract symmetries in the perceptual data and motor signals, and convert them into mathematical group objects (also known as *fiber group* and *control group* objects). These group objects are then structured hierarchically - which is the natural form of a wreath product - and encode the generative process followed to achieve those symmetries.

Our contribution is the implementation of the generative wreath product theory using symmetries in sensorimotor data in order to inform concept formation in embodied robots. We attempt to advance the state-of-the-art in robot cognition using mathematical group and symmetry theories embedded a priori in the robot. These *innate theories* act on sensorimotor data streams to produce concepts and build knowledge over time based on the robot's experience in the world. Rather than having all the knowledge the robot needs to operate built explicitly into it by the system engineer, using innate theories and minimal a priori knowledge allows the robot to respond to unknown situations, thus lending itself to structural bootstrapping. Figure 1 from [2] is a high-level cognitive architecture where innate knowledge is provided to the embodied agent in the form of:

1) **Theories:** Axiom sets; e.g., the four axioms of group theory.

2) **Modeling Processes:** In order to apply a theory to a specific world, the sets, operators, functions, constants, etc. must be mapped to a specific domain. In addition, identifiable sets must be determined from the data, as well as operators. Once a mapping between the world and theory is proposed, the model is validated by checking that the axioms hold for this mapping. For instance, if the robot observes a set of rotations, and uses *apply rotation* as the operator, then it can determine that this set, i.e., the rotations, and operator, *apply rotation*, form a group since every rotation followed by another rotation is also a rotation, the identity for the group is rotation by 0 degrees, the inverse rotation is rotate by the negative of the angle, and rotations are associative.
   $\rightarrow$ Note that the theories and modeling processes allow the recognition of symmetries in the data, that is, groups.

3) **Wreath Products:** Innate knowledge provides mechanisms to construct wreath products from symmetries detected in the data. Details will be given below, but as an example consider the 2D boundary of a square shape. The symmetry elements (groups) provided by sensorimotor data would be points on the boundary, translation symmetries provide the line segments, and a rotation symmetry group Z_4 maps each line segment onto the others. The wreath product is formed by the constraints imposed by the relations of the symmetry elements. The evaluation of the concept involves a characterization of its likelihood and this includes a Bayesian formulation (details given in section VI).

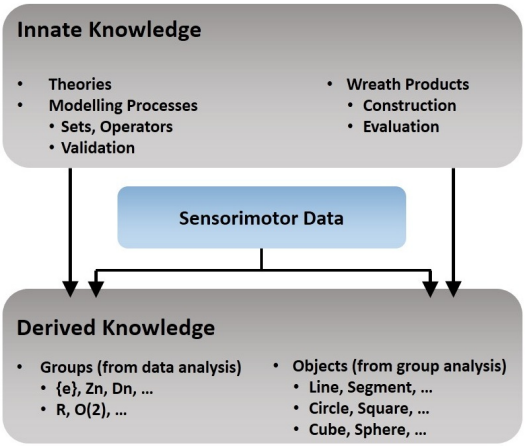

Fig. 1. High Level Cognitive Process.

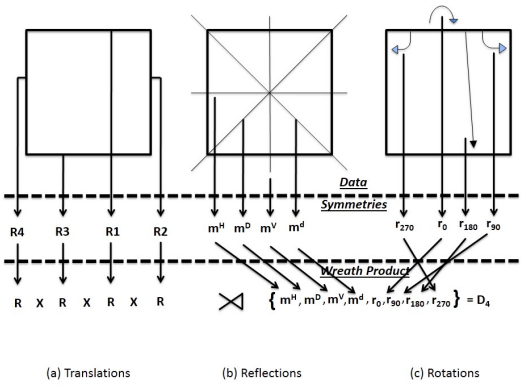

Fig. 2. Symmetry Detection and Wreath Product Formation for a 2D square.

II. Background And Related Work

A large number of cognitive archtitectures have been proposed over the last few decades, and they can be split into three high-level categories: (1) symbolic, (2) emergent dynamic systems, and (3) hybrid. For a careful review of these, see [9]. Binford ([1]) proposes a directed acyclic graph (DAG) approach to represent Bayesian networks based on generalized cylinders. These Bayesian networks accrue probabilities for physical models and relations based on a match to a database of a priori physical objects and relations. In [2] we have given an introduction to wreath product concept representation and symmetry expression in the concept, as well as illustrated the theory of converting a wreath product representation into an equivalent Bayesian net; our approach in [2] differs from [1] in that we use more general wreath products that can represent generalized cylinders, and also encode the actuation used to generate those shapes; furthermore, we propose to derive Bayesian network directly from hierarchical structure of the wreath product. The current paper is an extension of our aforementioned paper, where we expand by building a working system comprised of an object-oriented wreath product representation.

III. Generative Concepts

The wreath product framework is laid out by Leyton [4] where he proposes that the wreath product captures the notion of a generative concept. A wreath product is a group formed by a splitting extension of the direct product of the fiber group which is acted on by a control group (usually a permutation group). Moreover, the wreath product may be derived from perception data, and also includes the actuation sequence that was used to generate that data (or features of the data).

Leyton divides the wreath product into the Fiber Group (*G(F)*), where F is the set of fiber group elements which gets acted upon, and the Control Group (*G(C)*), where C is the set of control group elements that acts on the fiber group elements. Figure 2 shows how this generative process applies to represent the concept of a square shape along with underlying symmetries. The square is understood as being formed by following along one edge (denoted by the real line $\Re$ as translation), then this side rotated 0, 90, 180, or 270 degrees to generate the other sides (see [2] for details). Of course, symmetries and their relations in real data are rarely if ever, perfect, and thus the error in the label must be provided. For example, an R in the figure corresponds to a straight line segment; these are generally comprised of edge pixels which are combined to make the line segment. The error in fit of the line to the point data can serve to measure the linear symmetry in the data. Above that level, the sides may not be perfectly equal in length, or may not form exactly perpendicular angles where they meet. This means that the figure may not be a square, but rather a rhomboid. These uncertainties can be characterized by Bayesian networks which have a priori probabilities as occurence statistics that improve over time (posterior probabilities).

IV. Symmetry Extraction from Range Data

A. Segmentation

Depth images obtained from a *Kinect* sensor have a natural image structure to them (Figure 3) that helps in segmenting objects based on the surface normal variations at neighborhoods of individual pixels (*rangels*).

Given neighborhood patch around a pixel (Figure 5 (a)), the normals in the planar patch are given by $(\bar{b} - \bar{a})$ and $(\bar{c} - \bar{a})$. If the directional derivatives are given by

$$p = \frac{\partial f}{\partial x} \text{ and } q = \frac{\partial f}{\partial y},$$

then the unit surface normal is given by $\frac{[-p,-q,1]}{\sqrt{1+p^2+q^2}}$.

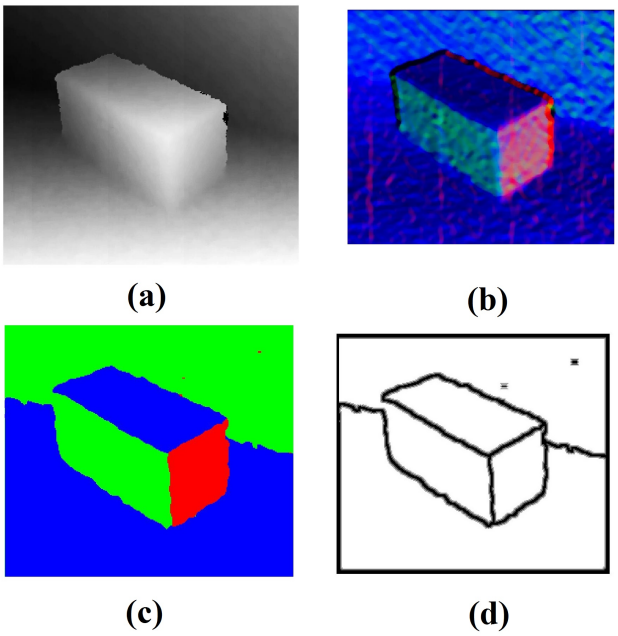

Fig. 3. Depth image analysis of a rectangular box. (a) Raw depth image. (b) Surface normals displayed as an RGB image. (c) Surface normals clustered into planes using *k*-means. (d) Edges found using maximal surface normal deviations.

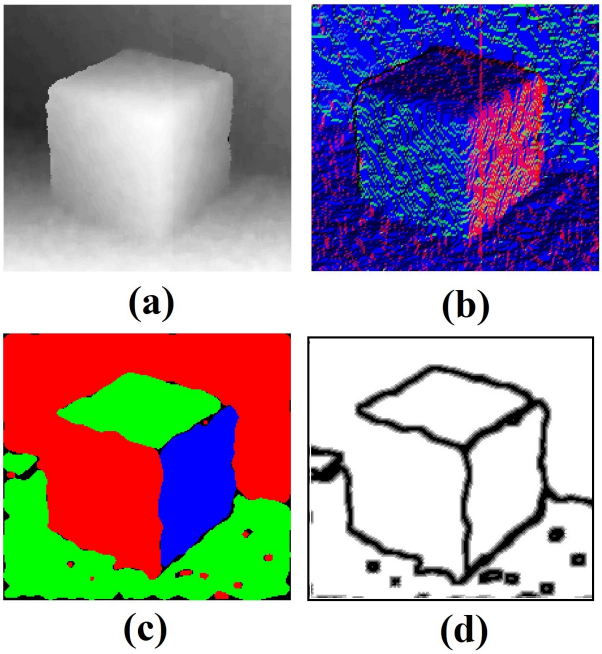

Fig. 4. Depth image analysis of a cube. (a) Raw depth image. (b) Surface normals displayed as an RGB image. (c) Surface normals clustered into planes using *k*-means. (d) Edges found using maximal surface normal deviations.

Given the surface normals at each pixel, significant changes in depth result from significant change in the direction of surface normal. Surface normal edge detection technique developed by Uckermann et al. [7] is used to find these edges. Their method finds the difference in neighboring normals as $\cos(\angle(\hat{n}_1, \hat{n}_2)) = \hat{n}_1.\hat{n}_2$, in eight directions of a given pixel (Figure 5 (b)). For a given direction, say North (N), the score for that direction is given by $\cos(\theta_N) = \frac{1}{3}\sum_{i=1}^{3} \hat{n}_{x,y}.\hat{n}_{x,y+i}$. The strongest angular deviation between a pixel and its neighbors in all directions is selected by

$$\min(\cos(\theta_N), \cos(\theta_N E), ..., \cos(\theta_W), \cos(\theta_N W))$$

which is thresholded at a value of 0.85 where anything less than 0.85 denotes a depth edge at that location.

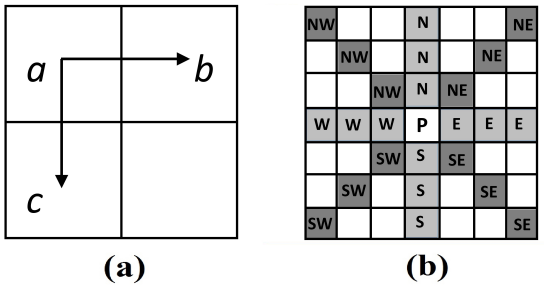

Fig. 5. (a) Neighborhood of a pixel. Cross product of vectors $(\bar{b} - \bar{a})$ and $(\bar{c} - \bar{a})$ is the normal at pixel designated by point a. (b) The 8 directions in which normal deviation is calculated.

Figure 3 (a) shows raw Kinect depth image of the rectangular box and Figure 3 (b) shows the surface normals displayed as an RGB image. Due to inherent noise in the depth estimation, surface normal calculation contains noise as is visible in the RGB image resulting in false positive edges. To solve this problem we use *k*-means (where *k*=3) to cluster surface normals having similar orientations belonging to same plane (Figure 3 (c)). These clustered normals are again subjected to surface normal edge calculations to yield a clean edge map Figure 3 (d). Figure 4 shows similar analysis for a square box.

B. Symmetry Operators

We segment the planes based on the cluster results seen above. For a set of points belonging to each plane, the normal to the plane is determined using least squares fit of the points. For a pair of these normals (one for each plane) the line of intersection of the two planes is determined using the cross product of this pair. Figure 6 shows the 3 lines (for 3 faces of the box). For each of these lines, the points on the surface normal edges of the rectangular box that are closer than a certain threshold distance (20 in our case) are marked as belonging to one edge of the box. Thus points displaying closeness within a cylindrical (radius = threshold) volume are considered for further processing.

In order to retrieve the higher order symmetry groups on a planar face, we use the Frieze Expansion Pattern (FEP) proposed by [3]. (See figure 7 (b) and figure 8).

The local minima lines (Figure 8 (d)), when projected back to the original convex hull represent the reflection planes (Z_2 symmetry group) of the face (Figure 7 (b)). Note, however, that local maxima in the FEP would correspond to end points of sides; another similarity measure - Planar-Reflective Symmetry Transform (PRST) [5] - can be used to distinguish between square and rectangular faces when reflected across diagonals.

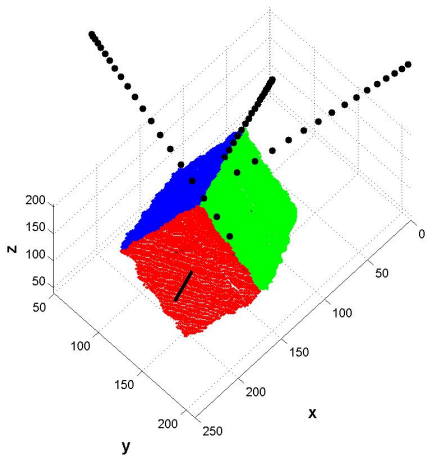

Fig. 6. The 3 orthogonal (dotted) lines parallel to, and close to surface normal edges (line of intersection of 2 faces of the box). Normal to one of the face is also visible as a solid line

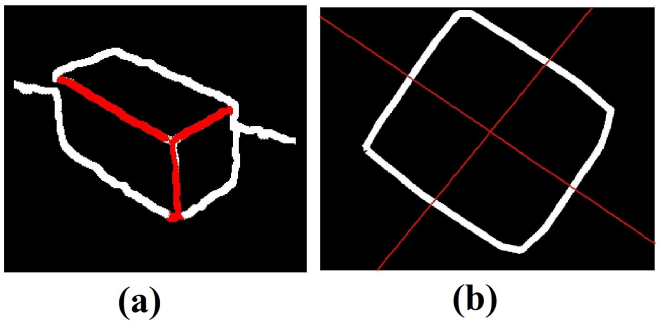

Fig. 7. (a) Points close to plane-intersection lines are shown in red. All marked points are less than 20 units of distance away from plane-intersection line. (b) Z_2 reflection axes represented by the red lines

C. Focal Interest Operator (FIO)

The process for detecting translational symmetry is given in algorithm 1.

Data:
$\mathcal{P} \leftarrow$ All points displaying symmetry
$E \leftarrow \phi$ (Objects belonging to $\{e\}$))
$\mathcal{R} \leftarrow \phi$ (Objects belonging to R)
for $\forall p \in \mathcal{P}$ **do**
  $d \leftarrow \phi$
  for $\theta = 0$ *to* 2π **do**
    $d = d \cup distance\_to\_edge$
  end
  $max\_distance = max(d)$
  $max\_dir \leftarrow direction(max\_distance)$
  $Pts \leftarrow translate\_along\_max\_dir(max\_dir)$
  $E \leftarrow Create\_objects(Pts, \{e\})$
  $\mathcal{R} \leftarrow assign(E)$
end

Algorithm 1: Extract Symmetry Group Representation

The symmetry extraction phase involves the use of a point of focus as an interest point around which we look for

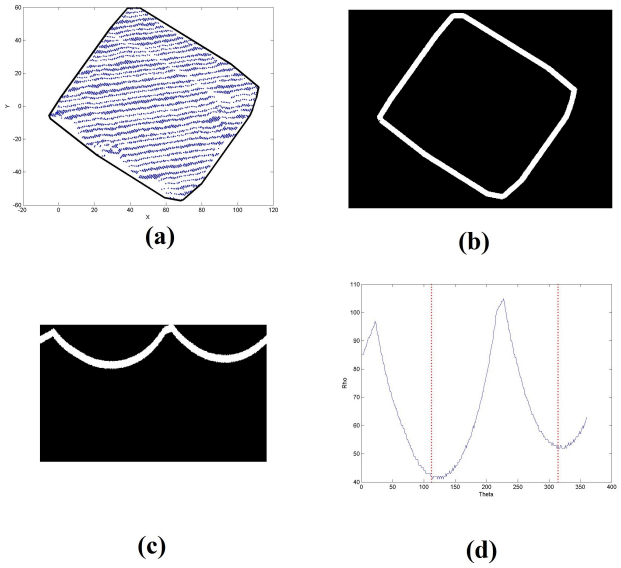

Fig. 8. FEP analysis: (a) A rectangular face projected onto a plane and the convex hull of the resulting points. (b) The convex hull. (c) Convex hull converted to an FEP. (d) Distance to edge of FEP plotted. Local minima of the FEP correspond to mid points of each side.

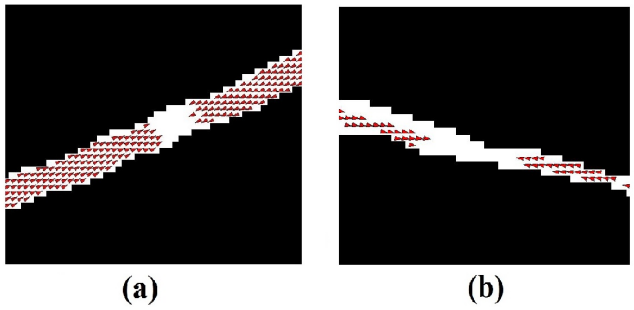

Fig. 9. Translational symmetries in points belonging to an edge of a (a) Rectangular Box. (b) Square Box.

symmetries. Initially the focal point is randomly chosen from a set of points in the data that display geometric symmetries in 3D space (e.g.: points within a cylindrical volume above). The amount of movement of the focus point is subject to the constraint of maintaining, at all times, a symmetry measure above a certain threshold. If $\bar{d}_t = [(x_t - x_{t-1}), (y_t - y_{t-1}), (z_t - z_{t-1})]^T$, is the vector representing the direction in which the focal point is moving at time t, the direction in which the FIO should move at time $t+1$ is given by maximizing a symmetry detection function, f, which considers the neighborhood of focal point at $[x_t, y_t, z_t]^T$, given by D_t

$$\bar{d}_{t+1} = \arg\max_{\bar{d}_t} f(D_t)$$

This FIO will move in the direction of maximum symmetry until:

1) the function f can no longer be maximized above a certain threshold, or
2) there are no more data points to move to, in the direction given by d_{t+1}.

The list of points, say P, comprising the path of FIO from start (time 0) to end (time t) is recorded and those points are assigned to a group class based in the following manner:

1) Each point $\in$ P is assigned a class $\{e\}$, which represents the identity element of a group. Call set of all $\{e\}$s as E.
2) If E satisfies the equation of a line within a certain error threshold, then E is assigned a class $\mathfrak{R}$ which represents a line.

We use FIO approach to detect translational symmetries in points. In our example, the translations detected by algorithm 1 - applied to the edges detected within a cylindrical volume - are shown for a rectangular box (figure 9 (a)) and for a square box (figure 9 (b)) by red arrows indicating the direction of maximal symmetry measure. Note that two directions are detected since both are valid translations maximizing translational symmetry. Figure 10 demonstrates the translational symmetry detection at pixel level on range data for the rectangular box. The set of identity elements (pixels, or $\{e\}$) are assigned as a fiber group to the control group R.

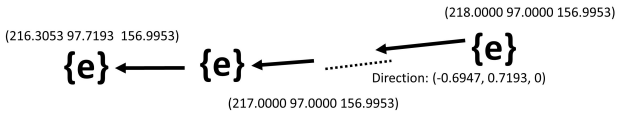

Fig. 10. Translation in pixel space on depth data.

V. Wreath Product Representation: Object-Oriented Design

We follow the object-oriented approach to realize the theoretical wreath-product framework. In this approach, every node of the wreath-product tree {*fiber group control group*} is an object with its own class definition, which is semantically equivalent to a mathematical group.

The wreath product is constructed as a tree of such object instances where a control group object instance is the parent of one or more fiber group object instance(s), which in turn can be the control group of its own fiber group(s). This hierarchical tree structure lends itself for a direct conversion to a Bayesian network with probabilities assigned to the individual nodes.

The most basic class in our representational semantics is $\{e\}$ which represents a point in space. Some of the properties associated with $\{e\}$ are:

1) **Position:** Location of the point in 3D space.
2) **Normal:** Normal at this point.
3) **Color:** RGB value of the pixel.
4) **Gray value:** Grayscale value of the pixel.
5) **Binary value:** Indicates whether the pixel is turned on or off (corresponds to the Z_2 group).
6) **Textures:** Texture patches around this point. This will be a list of texture properties with associated texture values.
7) **Control Group:** Indicates parents (control group), which this object is a fiber group of.
8) **Control Group:** Indicates parents (control group), which this object is a fiber group of.
9) **Property List:** [Name Value] pairs of additional ad hoc properties that might be associated with this object during operation.

To represent a line, we would need the group $\mathfrak{R}$ which represents the control group that moves the fiber group $\{e\}$ along the reals ($\mathfrak{R}$). A line is thus represented as $\{e\} \wr \mathfrak{R}$. The class $\mathfrak{R}$ will have the following properties:

1) **Endpoints:** A line can be represented by its two end points in space.
2) **Line parameters:** Parametric equation $(ax + by + cz + d = 0)$ coefficients (a, b, c, d), can be stored.

The resulting object-oriented representation of a line will be a hierarchical tree structure where the parent node (the control group object $\mathfrak{R}$) has its children as alll the points (the fiber group objects $\{e\}$). In theory there will be infinitely many points along the reals ($\mathfrak{R}$), however in practice the data will be discretized to a certain granularity (eg., pixels in images or points in point clouds) thus limiting the number of $G(F)$ belonging to $G(C)$.

A rectangular face of the rectangular box example above can be represented as a WP shown in Figure 11. Using the translational symmetry ($\{e\} \wr R$) and reflection symmetry feature (Z_2) extracted above, the entire WP can now be assembled to represent a rectangular face.

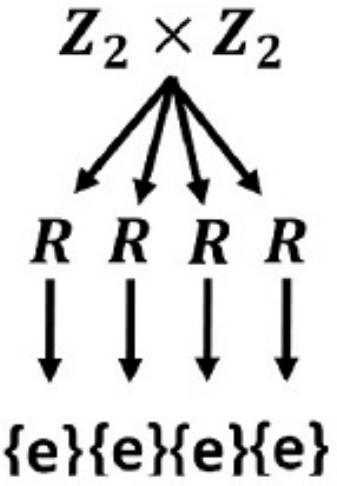

Fig. 11. Wreath Product for a rectangular face.

In order to represent permutation, and other symmetry groups, we exploit the fact that any transformation of data points can be expressed as a transformation matrix acting on those points. Since a group consists of a set and an operator that satisfy certain group axioms, we can view this set as the set of matrix transformation that operate on points through multiplication operator.

VI. Conclusions and Future Work

We have given the motivation behind creating a symmetry-based cognitive framework along with a demonstration of a working WP generation flow for simple world objects. Symmetry is ubiquitous in most man-made operational environments and many natural operational environments. The present work is a stepping stone towards achieving symmetry exploitation for cognitive concept representation. However, most environments are more complex than the one presented in this paper and better segmentation methods need to be developed to group interest points that belong to one feature of a world object (e.g.: an edge/plane/section that belongs to a part of an object). Once these points are discovered, various symmetry operators can be used to extract symmetry groups out of real-world data.

Another challenge in detecting local symmetries is the noise inherent in any depth imaging device (e.g.: *Kinect*). Since *Kinect* uses structured lighting (pattern projection) and triangulation to detect depth, it is susceptible to ambient light effect, range limitations, and disparity calculation errors, which results in noisy depth images. Generalized denoising techniques need to be developed that would work in any scenario. Any remaining noise needs to be quantified and incorporated in WP concepts such that it will support inference and generalization. This implementation would be tested on our *SYMMBOT* platform – a small rover currently under development – which would use the *Kinect* sensor and operate in the real world taking data and extracting symmetry information out it.

We conclude by acknowledging two issues that pertain to using the proposed techniques and the robot platform for building our cognitive framework.

A. Uncertainty Quantification

WPs have a tree structure which lends itself to an overlay of a Bayesian network on top of this WP structure. As demonstrated in [2], this overlay would help in quantifying uncertainty at each level (feature or subset of feature) in our data, and thus help in generalization of concepts. Depth data is inherently noisy due to the structured light mechanism used to generate it, and it reflects in the object segmentation technique discussed above. This noise needs to be quantified as a distribution around the true depth (and the true edge of the object).

B. Point Clouds

Our goal is to adapt depth image analysis to analyze point clouds. True 3D data consists of (X, Y, Z) points rather than $(row, column, Z)$ depth pixels in case of a depth image. Point clouds do not have an inherent image structure associated to them as depth images do, and although point clouds can be converted to depth images, this process results in loss of information as pixels have to be interpolated, some 3D points need to be discarded and depth scaling issues occur. However point clouds have true 3D structure information associated to them, unlike depth images, and have the potential to generate more accurate WP structure annotations.

We use the optimized RANSAC method for shape detection formulated by Schnabel et al. [6] to achieve an initial segmentation. This method uses an octree structure to limit RANSAC minimal subset selection to points that have spatial proximity to each other, and uses an optimized scoring function for fitting shape models to these subsets of points. Thus shape detection is achieved in realtime. Although their method extracts the labeled points corresponding to, and along with, the types of shapes detected, we only use the labeled 3D points for further processing. The segmented point cloud can be seen in Figure 12 (Top) with unsegmented point clouds of a rectangular box (observed in the direction of z-axis), a cylinder (observed in the direction of y-axis) and a sphere (observed in the direction of -z-axis). (Bottom) The segmented point cloud with labelled points for corresponding shapes.

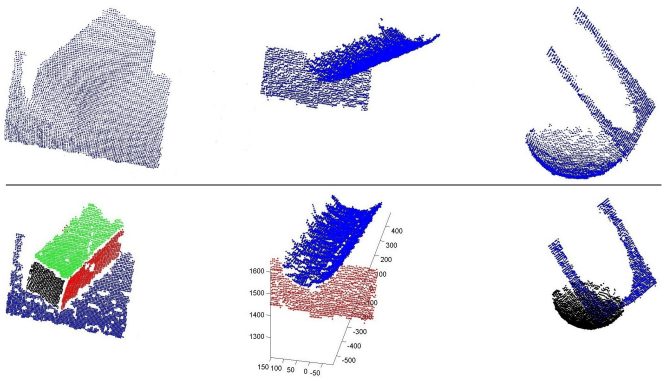

Fig. 12. (Top) Point clouds for box, cylinder and sphere, respectively. (Bottom) Corresponding segmented point clouds.

Acknowledgment

This work was supported by AFOSR-FA9550-12-1-0291

References

[1] T. O. Binford, T. S. Levitt, and W. B. Mann. Bayesian Inference in Model-Based Machine Vision. *Conference on Uncertainty in Artificial Intelligence*, Seattle, United States of America 1987.

[2] A. Joshi, T.C. Henderson and W. Wang. Robot Cognition using Bayesian Symmetry Networks. *International Conference on Agents and Artificial Intelligence*, Angers, France. March 2014.

[3] L. Seungkyu, R. T. Collins, and Y. Liu. Rotation symmetry group detection via frequency analysis of frieze-expansions. *Computer Vision and Pattern Recognition*, 2008. IEEE Conference on. IEEE, 2008.

[4] M. Leyton. *A generative theory of shape*. Vol. 2145. Springer, 2001.

[5] J. Podolak et al. A planar-reflective symmetry transform for 3D shapes. *ACM Transactions on Graphics (TOG)*. Vol. 25. No. 3. ACM, 2006.

[6] R. Schnabel, W. Roland, and R. Klein. Efficient RANSAC for PointCloud Shape Detection. *Computer Graphics Forum*. Vol. 26. No. 2. Blackwell Publishing Ltd, 2007.

[7] A. Uckermann, C. Elbrechter, R. Haschke, and H. Ritter. 3D scene segmentation for autonomous robot grasping. *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on* (pp. 1734-1740). IEEE.

[8] D. Vernon, G. Metta, and G. Sandini. A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. *Evolutionary Computation*, IEEE Transactions on 11.2 (2007): 151-180.

[9] D. Vernon. Enaction as a conceptual framework for developmental cognitive robotics. *Paladyn* 1(2): 89-98, (2010).