Beyond Point Clouds: Scene Understanding by Reasoning Geometry and Physics

Bo Zheng*, Yibiao Zhao†, Joey C. Yu†, Katsushi Ikeuchi*, and Song-Chun Zhu†

* The University of Tokyo, Japan
{zheng, ki}@cvl.iis.u-tokyo.ac.jp

† University of California, Los Angeles (UCLA), USA
{ybzhao,chengchengyu}@ucla.edu, sczhu@stat.ucla.edu

Abstract

In this paper, we present an approach for scene understanding by reasoning physical stability of objects from point cloud. We utilize a simple observation that, by human design, objects in static scenes should be stable with respect to gravity. This assumption is applicable to all scene categories and poses useful constraints for the plausible interpretations (parses) in scene understanding. Our method consists of two major steps: 1) geometric reasoning: recovering solid 3D volumetric primitives from defective point cloud; and 2) physical reasoning: grouping the unstable primitives to physically stable objects by optimizing the stability and the scene prior. We propose to use a novel disconnectivity graph (DG) to represent the energy landscape and use a Swendsen-Wang Cut (MCMC) method for optimization. In experiments, we demonstrate that the algorithm achieves substantially better performance for i) object segmentation, ii) 3D volumetric recovery of the scene, and iii) better parsing result for scene understanding in comparison to state-of-the-art methods in both public dataset and our own new dataset.

1. Introduction

1.1. Motivation and Objectives

Traditional approaches for scene understanding have been mostly focused on segmentation and object recognition from 2D images. Such representations lacks important physical information, such as the 3D volume of the objects, supporting relations, stability, and affordance which are critical for robotics applications: grasping, manipulation and navigation. With the recent development of Kinect camera and the SLAM techniques, there has been growing interest in studying these properties in the literature [17].

In this paper, we present an approach for reasoning physical stability of 3D volumetric objects reconstructed from either a depth image captured by a range camera or a large scale point cloud scene reconstructed by the SLAM technique [17]. We utilize a simple observation that, by human design, objects in static scenes should be stable. For example, a parse graph is said to be valid if the objects, according to its interpretation, do not fall under gravity. If an object is not stable on its own, it must be grouped with attached neighbors or fixed to its supporting base. In addition, while objects are stable physically, they should enjoy a movable space (freedom) for manipulation. Such assumption is applicable to all scene categories and thus pose quite powerful constraints for the plausible interpretations (parses) in scene understanding.

As Fig. 1 shows, our method consists of two main steps.

1) **Geometric reasoning**: recovering solid 3D volumetric primitives from defective point cloud. Firstly we segment and fit the input 2.5D depth map or point cloud to small simple (*e.g.*, planar) surfaces; secondly, we merge convexly connected segments into shape primitives; and thirdly, we form 3D volumetric shape primitives by filling the missing (occluded) voxels, so that each shape primitive can own its physical properties: volume, mass and supporting areas to compute the potential energies in the scene. Fig. 1.(d) shows the 3D primitives in rectangular or cylindrical shapes.

2) **Physical reasoning**: grouping the primitives to physically stable objects by optimizing the stability and the scene prior. We build a contact graph for the neighborhood relations of the primitives as shown in Fig. 1.(e), coloring this graph corresponds to grouping them into objects. For example, the lamp on the desk originally was divided in 3 primitives and will fall under gravity (see result simulated using a physics engine), and become stable when they are grouping into one object – the lamp. So is the computer screen with its base.

To achieve the physical reasoning goal, we make the following novel contributions in comparison to the most recent work in dealing with physical space reasoning [8, 16].

- We define the physical stability function explicitly by studying minimum energy (physical work) need to change the pose and position of an primitive (or ob-

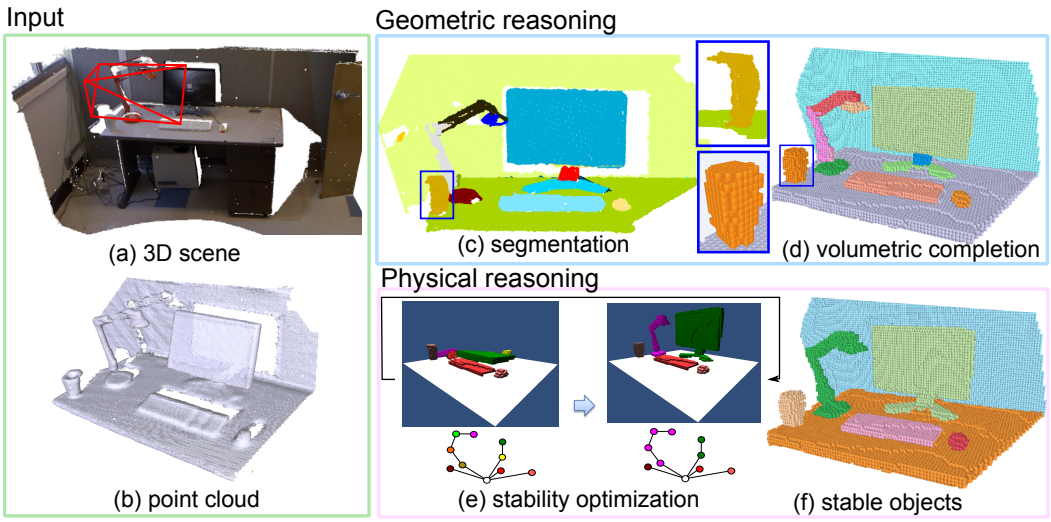

Figure 1. Overview of our method. (a) 3D scene reconstructed by SLAM technique, (b) point cloud as Input. In geometric reasoning, (c) a portion is shown to be segmented by a segment-and-merge approach, with missing voxels, (d) solid primitives by volumetric completion. In physical reasoning, (e) the contact graph are labeled through stability optimization. (f). Final parsing results with stable objects.

ject) from one equilibrium to another, and thus to release potential energy.

- We introduce disconnectivity graph (DG) from physics (Spin-glass) to represent the energy landscapes.
- We solve the complex optimization problem by the cluster sampling method Swendsen-Wang cut in image segmentation [2] to maximize global stability.
- We collect a new dataset for large scenes by depth sensors for scene understanding and will release the data and annotations to the public.

In experiments, we demonstrate that the algorithm achieve a substantially better performance for i) object segmentation, ii) 3D volumetric recovery of the scene, and iii) better parsing result for scene understanding in comparison to state-of-the-art methods in both public dataset [16] and our own new dataset.

1.2. Related work

Our work is related to 3 research streams in the literature.

1. Geometric reasoning. Our approach for geometry reasoning is related to a set of segmentation methods (*e.g.*, [12, 1, 18]). Most of the existing methods are focused on classifying point clouds for object category recognition, not for 3D volumetric completion. For work in 3D geometric reason, Attene *et al*. [1] extracts 3D geometric primitives (planes or cylinders) from 3D mesh. In comparison, our method is more faithful to the original geometric shape of object in the point cloud data. There have been also interesting work in constructing 3D scene layouts from 2D images for indoor scenes, such as Zhao and Zhu [21], Lee *et al*. [15, 14], Hedau *et al*. [11]. Furukawa *et al*. [7] also performed volumetric reasoning with the Manhattan-world assumption on the problem of multi-view stereo. In comparison, our volumetric reasoning is based on complex point cloud data and provides more accurate 3D physical properties, *e.g.*, masses, gravity potentials, contact area,*etc*..

2. Physical reasoning. The vision communities have studied the physical properties based on single image for the "block world" in the past three decades [3, 8, 9, 21, 15, 14]). E.g. Biederman *et al*. [3] studied human sensitivity of objects that violate certain physical relations. Our goal of inferring physical relations is most closely related to Gupta *et al*. [8] who infer volumetric shapes, occlusion, and support relations in outdoor scenes inspired by physical reasoning from a 2D image, and Silberman *et al*. [16] who infer the support relations between objects from single depth image using supervised learning with many prior features. In contrast, our work is the first that defines explicitly the mathematical model for object stability. Without supervised learning process, our method is able to infer the 3D objects with maximum stability.

3. Intuitive physics model. Recent psychology studies suggested that approximate Newtonian principles underlie human judgements about dynamics and stability [6, 10]. Hamrick *et al*. [10] showed that knowledge of Newtonian principles and probabilistic representations are generally applied for human physical reasoning, and the intuitive physics model is an important perspective for human-level complex scene understanding. However, to our best knowledge, there is little work that mathematically defines intuitive physics models for real scene understanding. Physics

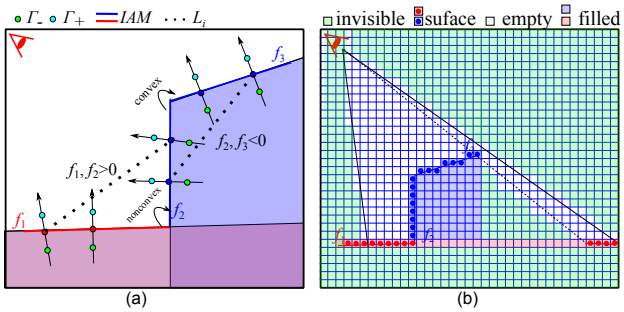

Figure 2. (a) Two 1-degree IAMs f_1 and f_2 (in blue and red lines respectively) are fitted to the 3-Layer point cloud. The light red and blue areas denote in which functions f_1, f_2 and f_3 are minus. (b) Invisible space estimation and voxel completion. Four types of voxels are estimated: invisible voxels (light green), empty voxels (white), surface voxels (red and blue dots), and the voxels filled in the invisible space (colored square in light red or blue).

engines in graphics can accurately simulate the motion of objects under gravity, but it is computationally expensive for the purpose of measuring object stability.

2. Geometric reasoning

Given a point cloud of scene, the goal of geometric reasoning is to infer the object primitives (*e.g.*, the colored objects in Fig. 1 (d)), such as that each primitive can own physical properties (*e.g.*, volume, mass, supporting area, *etc.*). We infer the object primitives with two major steps: 1) point cloud segmentation and 2) Volumetric completion.

2.1. Segmentation with implicit algebraic models

We first adopt implicit algebraic models (IAMs) [4] to separate point cloud into several simple surfaces. We adopt a split-and-merge strategy as: 1) splitting the point cloud into simple and smooth regions by IAM fitting, and then 2) merging the regions which are "convexly" connected each other. As a 2D example illustrated in Fig. 2.(a), suppose the 2D point cloud is first split into three line segments with first-order IAM fitting: f_1, f_2 and f_3, and then f_2 and f_3 are merged together, since they are "convexly" connected.

Splitting point cloud. The objective in this process can be considered as to find out the 3D regions, and each of them can be well fitted by an IAM.

The IAM fitting for each region can be formulated in least squares optimization using the 3-Layer method proposed by Blane *et al*. [4]. As shown in Figure 2.(a), it first generate two extra point layers: Γ_-(green points) and Γ_+ (light blue points) along the normals of points in the original region M (red and blue points). Then an IAM can be fit to M by linear least-squared method with linear constraints:

$$f(\mathbf{p}_i) = \begin{cases} 0, & \mathbf{p}_i \in M \\ +d_i, & \mathbf{p}_i \in \Gamma_+ \\ -d_i, & \mathbf{p}_i \in \Gamma_- \end{cases}, \quad (1)$$

where f is an implicit polynomial, $\pm d$ is the Euclidean distance how long the two points move along the normals in opposite directions. Therefore, as shown in Fig. 2 (a), each IAM fit can split the space into two parts: "inside" (colored with negative value) and "outside" (uncolored (white) with positive value).

For splitting point cloud into pieces, we adopt region growing scheme [18]. Our method can be described as: starting from several given seeds, the regions grow until there is no unlabeled point can be fitted by certain IAM. In this paper, we adopt the IAM of 1 or 2 degree, *i.e.*, planes or second order algebraic surfaces and the IAM fitting algorithm proposed by Zheng *et al*. [22] to select the models in a degree-increasing manner.

Merging "convexly" connect regions. The splitting strategy seems separating the points to be object faces (*e.g.*, a box can be split into six faces). However we can further merge the "convexly" connected regions to better represent object parts (primitives).

To this end, we first define "convex connection" of two regions as follow:

Definition 1. *for any line segment L whose two ends are in two connected regions with IAM fits f_i and f_j respectively, if the points on this line, $\{\forall \mathbf{p}_l | \mathbf{p}_l \in L\}$, satisfy $f_i(\mathbf{p}_l) < 0$ and $f_j(\mathbf{p}_l) < 0$, then we say regions i and j are convexly connected.*

To detect the convex connection, as shown in Fig. 2 (a), we first randomly sample several line points (in dark dot lines) between connected regions, and then check them if satisfy the convexly connected relationship defined above. In practice, we merge the convex connections when the following condition is satisfied:

$$\frac{\#\{\mathbf{p} | \mathbf{p}_l \in L \wedge f_i(\mathbf{p}_l) < 0 \wedge f_j(\mathbf{p}_l) < 0\}}{\#\{\mathbf{p} | \mathbf{p}_l \in L\}} > \delta, \quad (2)$$

where the ratio threshold δ is set as 0.6 according the sensor noise. In Fig 2 (a), since the dark points connecting f_2 and f_3 are submerged by both minus regions of them.

2.2. Volumetric space completion

To obtain the physical properties for each object primitive (*e.g.*, size, mass *etc.*), we need volumetric representation but not surface segments. Thus, we complete each surface segment into a volumetric (voxel-based) primitive under three assumptions: a) *Occlusion assumption*: voxels occluded by the observed point cloud could be parts of objects. b) *Solid assumption*: hollow object is not preferred

(e.g., plane should not with holes, or a box should be solid). c) *Manhattan assumption*: most object shapes are aligned with Manhattan axes.

Voxel generation and gravity direction We first generate voxels for each segment obtained by above point cloud segmentation by 1) detecting Manhattan axes [7], 2) constructing voxels from point cloud along Manhattan axes by octree construction method [19], and 3) detecting gravity direction. To detect gravity direction, we simply choose the one with smallest angle to the vertical axis of sensor coordinate system.

Invisible (occluded) space estimation. The space behind the point clouds and beyond the view angles is not visible from the camera's perspective. However this invisible space is very helpful for completing the missing voxels from occlusion. Inspired by Furukawa's method in [7], the Manhattan space is carved by the point cloud into three spaces (as shown in Figure 2(b)): Object surface $\mathbb{S}$ (colored-dots voxels), Invisible space $\mathbb{U}$ (light green voxels) and Visible space $\mathbb{E}$ (white voxels).

Voxels filling. We complete an object primitive from each labeled surface segment. Suppose each convex surface segment is the visible part of a primitive, we complete invisible part by filling voxels in a visual hull which is occluded by the surface under two assumptions: 1) as lights travel in lines, the voxels complected are behind the point clouds, as shown in Fig. 2.(b); 2) a primitive should be completed if it can be seen from at least two directions of Manhattan axes. Therefore our algorithm can be simply described as:

Loop: for each invisible voxel $v_i \in \mathbb{U}$, $i = 1, 2, \ldots$

1) From v_i, searching the voxels along 6 directions of Manhattan axes, to collect six nearest surface voxels $\{v_j \in \mathbb{S}\}$ $(j \leq 6)$.

2) Checking the label for each v_j, if there exist more than two same labels, then assign this label to voxel v_i.

3. Modelling object stability

3.1. Energy landscapes

A 3D object (or primitive) has a potential energy defined by gravity and its state (pose and center) supported by neighboring object in 3D space. The object is said to be *in equilibrium* when its current state is a local minimum (stable) or local maximum (unstable) of this potential function (See Fig 4 for illustration). This equilibrium can be broken by external work (*e.g.*, nature disturbance) and then the object moves to a new equilibrium and releases energy. Without loss of generality, we divide the change in two cases.

Case I: pose change. In Fig. 3, the chair in (a) is in a stable equilibrium and its pose is changed with external work to raise its center of mass. We define the energy change needed to the state change $\mathbf{x}_0 \rightarrow \mathbf{x}_1$ by

$$\mathcal{E}_r(\mathbf{x}_o \rightarrow \mathbf{x}_1) = (R\mathbf{c} - \mathbf{t}_1) \cdot m\mathbf{g}, \tag{3}$$

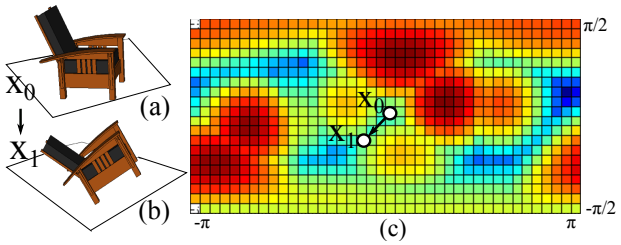

Figure 3. (a) A chair in a "stable" state $\mathbf{x}_0$ is moved to (b) an "unstable" state $\mathbf{x}_1$. (c) The landscape of potential energy is calculated by Eq. (3) over two rotation angles where $\mathbf{x}_0$ is a local minimum and $\mathbf{x}_1$ is a saddle point passing which, the chair will fall to a deeper energy basin (blue).

where R is rotation matrix; $\mathbf{c}$ is center of mass, $\mathbf{g} = (0, 0, 1)^T$ is the gravity direction, $\mathbf{t}_1$ is the lowest contact point on the support region (its legs). We visualize the energy landscape on the sphere (ϕ, θ): $S^2 \rightarrow \mathbb{R}$ in Fig. 3.(c) using the two pose angles $(\phi \in [-\pi\ \pi], \theta \in [-\pi/2, \pi/2])$. Blue color means lower energy and red means high energy. Such energy can be computed for any rigid objects by bounding the object with a convex hull. We refer to the early work of Kriegman [13] for further details.

Case II: position change. Imaging a cup on a desk at stable equilibrium state $\mathbf{x}_0$, one can push it to the edge of the table. Then it falls to the ground and releases energy to reach a deeper minimum state $\mathbf{x}_1$. The energy change needed to move the cup is

$$\mathcal{E}_t(\mathbf{x}_o \rightarrow \mathbf{x}_1) = (\mathbf{c} - \mathbf{t}) \cdot m\mathbf{g} - f, \tag{4}$$

where $\mathbf{t} \in \mathbb{R}^3$ is the translation parameter (shortest distance to the edge of the desk), and f is friction defined as $f = f_c\sqrt{(t_1 - c_1)^2 + (t_2 - c_2)^2}$ given the friction coefficient f_c. Note for common indoor scenes, we choose f_c as 0.3 as common material such as wood. Therefore the energy landscape can be viewed as a map from 3D space $\mathbb{R}^3 \rightarrow \mathbb{R}$.

In both cases, we observe that *object stability is only local and relative*, and can be changed subject to disturbance (gravity, wind, mild earthquake, and human activity).

3.2. Disconnectivity graph representation

The energy map is continuously defined over the object position and pose. For our purpose, we are only interested in how deep its energy basin is at current state (according to the current interpretation of the scene). Therefore, we represent the energy landscape by a so-called disconnectivity graph (DG) which has been used in studying the spin-glass models in physics [20]. In the DG, the vertical lines represent the depth of the energy basins and the horizontal lines connect adjacent basins. The DG can be constructed by an algorithm scanning energy levels from low to high and checking the connectivity of components at each level [20].

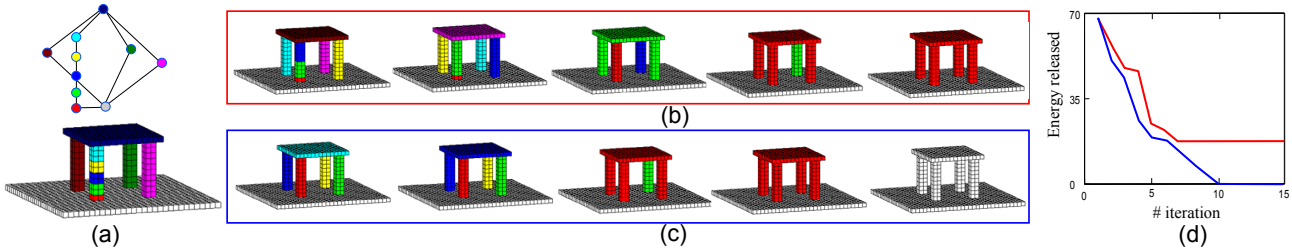

Figure 5. Example of illustrating the Swendsen-Wang sampling process. (a) Initial state with corresponding contact graph. (b) shows the grouping proposals accepted by SWC at different iterations. (c) convergence under larger disturbance W and consequently the table is fixed to the ground. (d) shows two curves of Energy released v.s. number of iteration in SWC sampling corresponding to (b) and (c).

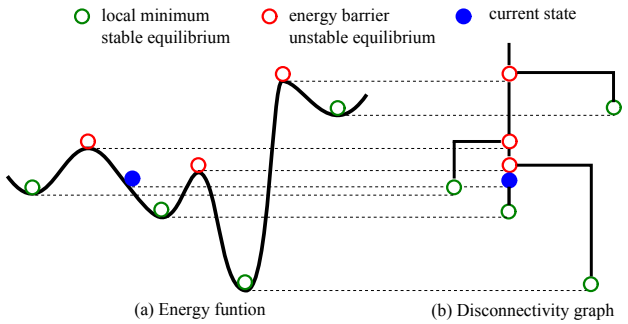

Figure 4. (a) Energy landscapes and its corresponding disconnectivity graph (b).

From the DG, we can conveniently calculate two quantities: *Energy absorption* and *Energy release* during the state changes.

Definition 2. *The energy absorption* $\Delta\mathcal{E}(\mathbf{x}_o \rightarrow \widetilde{\mathbf{x}})$ *is the energy absorbed from the perturbations, which moves the object from the current state* $\mathbf{x}_0$ *to an unstable equilibrium* $\widetilde{\mathbf{x}}$ *(say a local maximum or energy barrier).*

For the chair in Fig.3, its energy absorption is the work needed to push it in one direction to an unstable state $\mathbf{x}_1$. For the cup example, its energy barrier is the work needed (to overcome friction) to push it to the edge. In both cases, the energy depends on the direction and path of movement.

Definition 3. *Energy release* $\Delta\mathcal{E}(\widetilde{\mathbf{x}} \rightarrow \mathbf{x}'_o)$ *is the potential energy released when an object moves from its unstable equilibrium* $\widetilde{\mathbf{x}}$ *to a minimum* $\mathbf{x}'_0$ *which is lower but connected by the energy barrier.*

For example, when the cup falls of from the edge of the table to the ground. The higher the table, the larger the released energy.

With DG, we define object stability in 3D space.

Definition 4. *The* stability $S(a, \mathbf{x}_0, W)$ *of an object* a *at state* $\mathbf{x}_0$ *in the presence of a disturbance work* W *is the maximum energy that it can release when it moves out the energy barrier by the work* W.

$$\begin{aligned} &\mathrm{S}(a, \mathbf{x}_0, W) \\ &= \max_{\mathbf{x}'_0} \triangle\mathcal{E}(\widetilde{\mathbf{x}} \rightarrow \mathbf{x}'_o)\delta([\min_{\widetilde{\mathbf{x}}} \triangle\mathcal{E}(\mathbf{x}_o \rightarrow \widetilde{\mathbf{x}})] \leq W) \end{aligned} \quad (5)$$

where $\delta()$ is an indicator function and $\delta(z) = 1$ if condition z is satisfied otherwise $\delta(z) = 0$. $\triangle\mathcal{E}(\mathbf{x}_o \rightarrow \widetilde{\mathbf{x}})$ is the energy absorbed, if it is overcome by W, then $\delta() = 1$, and thus the energy $\triangle\mathcal{E}(\widetilde{\mathbf{x}} \rightarrow \mathbf{x}'_o)$ is released. We find the easiest direction $\widetilde{\mathbf{x}}$ to minimize the energy barrier and the worst direction $\mathbf{x}'_0$ to maximize the energy release.

4. Physical reasoning

Given a list of 3D volumetric primitives obtained by our geometric reasoning step, we first construct the contact graph, and then the task of physical reasoning can be posed as a well-known graph labelling or partition problem, through which the unstable primitives can be grouped together and assigned the same label to achieve global stability of the whole scene at a certain disturbance level W.

4.1. Contact graph and group labelling

The contact graph is an adjacency graph $G =< V, E >$, where $V = \{v_1, v_2, ..., v_k\}$ is the set of nodes representing the 3D primitives, and E is a set of edges denoting the contact relation between the primitives. An example is shown in Fig.1.(e) where each node corresponds to a primitive in Fig. 1.(c). If a set of nodes $\{v_j\}$ share a same label, that means these primitives are fixed to a single rigid object, denoted by O_i, and the stability is re-calculated according to O_i.

The optimal labelling L^* can be determined by the optimization of a global energy function, for a work level W

$$E(L|G; W) = \sum_{O_i \in L} (\mathrm{S}(O_i, \mathbf{x}(O_i), W) + \mathcal{F}(O_i)) \quad (6)$$

where $\mathbf{x}(O_i)$ is the current state of grouped object O_i. The new term $\mathcal{F}$ represents a penalty function expressing the scene prior and can be decomposed into parts.

$$\mathcal{F}(O_i) = \lambda_1 f_1(O_i) + \lambda_2 f_2(O_i) + \lambda_3 f_3(O_i), \quad (7)$$

where f_1 is the total number of voxels in object O_i; f_2 is the geometric complexity of O_i, which can be simply computed as the summation of the difference of normals for any two connected voxels on its surface; and f_3 is designed by the freedom of object movement on its support area. f_3 can be calculated as the ratio between the support plane and the contact area $\frac{\#S}{\#CA}$ of each pair of primitives $\{v_j, v_k \in \mathcal{O}_i\}$, where one of them is supported by the other. After they are regularized to the scale of objects, the parameters λ_1, λ_2 and λ_3 are set as 0.1, 0.1, and 0.7 in our experiment. Note, the third penalty is designed from the observation that, *e.g.*, a cup should have freedom of movement supported by a desk, and therefore the penalty arise if the mouse is assigned by same label to the table.

4.2. Inference of Maximum stability

As the label of primitives are coupled with each other, we adopt the graph partition algorithm Swendsen-Wang Cut (SWC) [2] for efficient MCMC inference. To obtain globally optimal $L^*$by the SWC, the next 3 main steps works iteratively until convergence.

(i) *Edge turn-on probability.* Each edge $e \in E$ is associated with a Bernoulli random variable $\mu_e \in \{on, off\}$ indicating whether the edge is turned on or off, and a weight reflecting the possibility of doing so. In this work, for each edge $e =< v_i, v_j >$, we define its turn-on probability as:

$$q_e = p(\mu_e = on|v_i, v_j) = exp(-(F(v_i, v_j)/T), \qquad (8)$$

where T is temperature factor and $F(\cdot, \cdot)$ denotes the feature between two connected primitives. Here we adopt a feature using the ratio between contact area (plane) and object planes as: $F = \frac{\#CA}{max(\#A_i, \#A_j)}$, where CA is the contact area, A_i and A_j are the areas of v_i and v_j on the same plane of CA.

(ii) *Graph Clustering.* Given the current label map, it removes all edges between nodes of different categories. Then all the remaining edges are turned on independently with the probability q_e. Thus, we have a set of connected components (CCPs) Π's, in which all nodes have the same category label.

(iii) *Graph Flipping.* It randomly selects a CCP Π_i from the set formed in step (ii) with a uniform probability, and then flips the labels of all nodes in Π_i to a category $c \in \{1, 2, ..., C\}$. The flip is accepted with probability [2]:

$$\alpha(L \to L') = \min(1, \frac{Q(L' \to L)E(L'|G; W)}{Q(L \to L')E(L|G; W)}). \qquad (9)$$

Fig. 5 illustrates the process of labeling a number of primitives of a table into a single object. SWC starts with an initial graph in (a), and some of the sampling proposals are accepted by the probability (9) shown in (b) and (c), resulted the energy v.s. iterations in (d). It is worth noticing that 1) in case of 5 (b), the little chair is not grouped to floor, since the penalty term A_3 penalize the legs to fix to floor. 2) On the other hand, we increase the disturbance W in (5), the chair is fixed to floor.

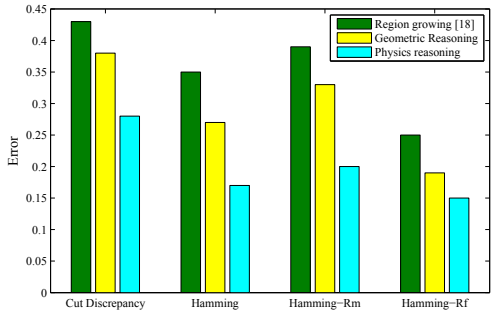

Figure 7. Segmentation accuracy comparison of three methods: Region growing method [18], result of our geometric reasoning and physical reasoning by one "Cut Discrepancy" and three "Hamming Distance".

5. Experimental result

We quantitatively evaluate our method in terms of 1) single depth image segmentation, 2) volumetric completion evaluation, 3) physical inference accuracy evaluation, and 4) intuitive physical reality (by videos in supplementary).

All these evaluations are based on three datasets: i) NYU depth dataset V2 [16] including 1449 RGBD images with manually labeled ground truth, ii) a set synthesized depth map and volumetric images simulated from CAD scene data. iii) 13 reconstructed 3D scene data captured by Kinect Fusion [17] gathered from office and residential rooms with ground truth labeled by a dense mesh coloring.

Evaluating Single depth image segmentation. Two evaluation criterion: "Cut Discrepancy" and "Hamming Distance" mentioned in [5] are adopted. The former measures errors of segment boundaries to ground truth, and the latter measures the consistency of segment interiors to ground truth. As result shown in Fig. 7, our segmentation by physical reasoning is with lower error rate than the another two: region growing segmentation [18], and our geometric reasoning.

Fig. 6 shows some examples for comparing point cloud segmentation result [18] and our result. However it is worth noticing that, beyond the segmentation task, our method can provide richer information such as volumetric information, physical relations, and stabilities *etc*.

Evaluating volumetric completion. For evaluating the accuracy of volumetric completion, we densely sample point clouds from a set of CAD data including 3 indoor scenes. We simulate the volumetric data (as ground truth) and depth images from a certain view (as test images). We calculate the precision and recall which evaluates voxel overlapping

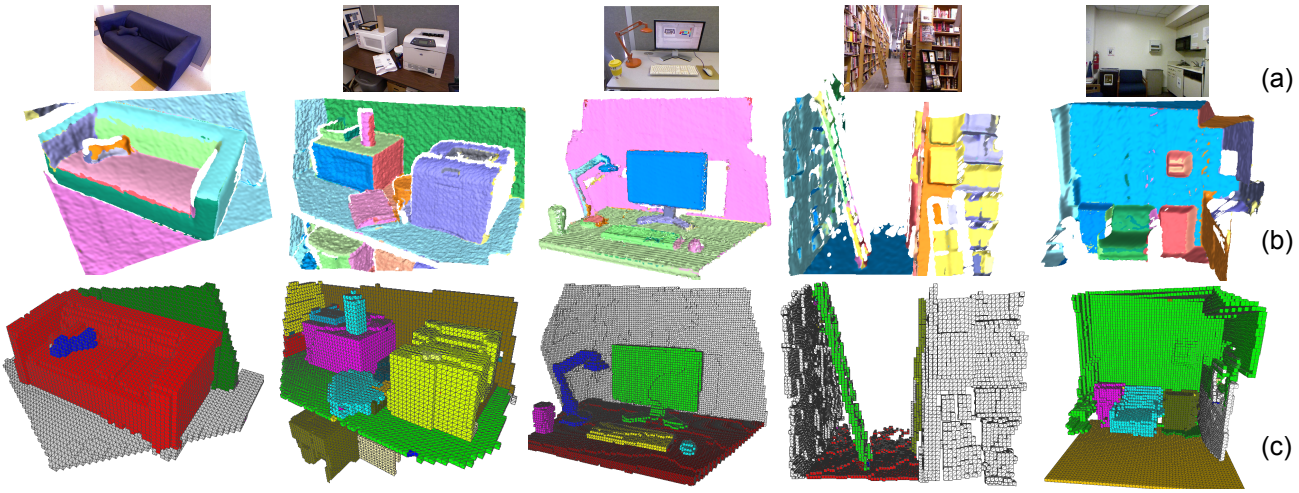

Figure 6. Segmentation result for single depth images. (a) RBG images for reference. (b) segmentation result by region growing [18]. (c) stable volumetric objects by physical reasoning.

	Octree [19]	Invisible space	Vol. com.
Precision	98.5%	47.7%	**94.1%**
Recall	7.8%	95.1%	**87.4%**

Table 1. Precision and recall of Volumetric completion. Comparison of three method: 1) voxel-based representation generated by Octree algorithm [19], 2) voxels in surface and invisible space (sec. 2.2), and 3) our volumetric completion.

relations	Discriminative	Greedy	SWC
fixed joint	20.5%	66%	**81.8%**
support	42.2%	60.3%	**78.1%**

Table 2. Results of inferring the fixed joints and support relations between primitives. Accuracy is measured by nodes of contact graph whose label is correctly inferred divided by the total number of labeled nodes.

between ground truth and the volumetric completion of testing data. Tab. 5 shows the result that our method has much better accuracy than traditional Octree method such as [19].

Evaluating physical inference accuracy. Because the physical relations are defined in terms of our contact graph, we map the ground-truth labels to the nodes of contact graphs obtained by geometric reasoning. Than we evaluate our physical reasoning against two baselines: discriminative methods of using 3D feature priors as similar as one in [16], and greedy inference method such as marching pursuit algorithm for physical inference. The result shown in Tab. 5 is evaluated by the average over 13 scene data captured by Kinect Fusion.

Figure 8 (a)-(d) and (e)-(j) show two examples from the results. Here we discuss some irregular cases by close-ups in the figures.

Case I: Figure 8 (c) the ball is fixed onto the handle of sofa. The reason can be considered as: stability of the "ball" is very low measured by Eq. (5). The unstable state is calculated out as that it trends to release much potential energy (draw from the sofa) by absorbing little possible energy (*e.g.*, the disturbance by human activity).

Case II: Figure 8 (d) the "air pump" unstably stands on floor but is an independent object, because although its stability is very low, the penalty designed in Eq.(7) penalized it to be fixed onto floor. So is the lamp not fixed to table in Figure 8 (h).

Case III: Figure 8 (g) the "empty Kinect box" with its base is fixed together onto the shelf, because of the miss segmentation of base, *i.e.*, the lower part of base is miss merged to top of shelf.

Case IV: Figure 8 (i) voxels under the "chair" are completed with respect to stability. The reasons are: 1) our algorithm reasons the hidden part occluded in invisible space. 2) the inference of hidden part is not accurate geometrically, but it helps to form a stable object physically. In contrast, original point cloud shown in Figure 8 (j) misses more data.

6. Conclusion

We presented a novel approach for scene understanding by reasoning the stability and unsafeness using intuitive mechanics with the novel representations of disconnectivity graph and disturbance field. Our work is based on a seemingly simple but power observation that objects, by human design, are created to be stable and have maximum utility (such as freedom of move). We demonstrated its feasibility in experiments and show that this provides an interesting way for object grouping when it is hard to pre-define all possible object shapes and appearance in an object category.

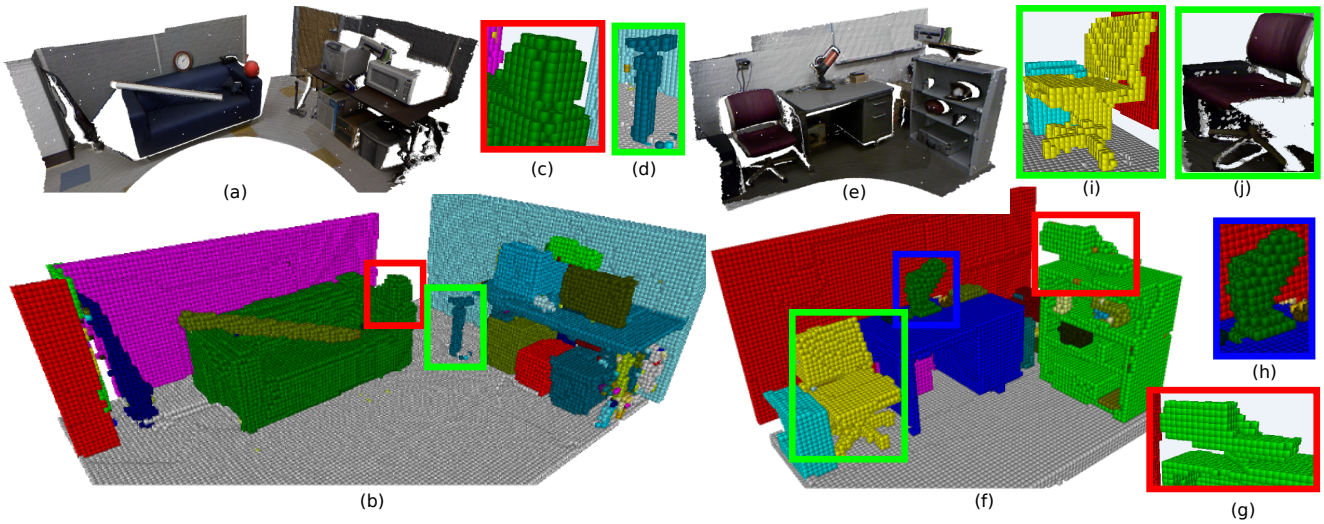

Figure 8. Example result. (a) and (e): data input. (b) and (f): volumetric representation of stable objects. (c): the ball is fixed onto the handle of sofa. (d): the "pump" is unstable (see text). (i): a irregular case of (g). (j): hidden voxels under chair compared to (h).

Acknowledgment

This work is supported by MURI ONR N00014-10-1-0933 and DARPA MSEE grant FA 8650-11-1-7149, USA; Next-generation Energies for Tohoku Recovery (NET) and SCOPE, Japan.

References

[1] M. Attene, B. Falcidieno, and M. Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. *THE VISUAL COMPUTER*, 22:181–193, 2006.

[2] A. Barbu and S. C. Zhu. Generalizing swendsen-wang to sampling arbitrary posterior probabilities. *TPAMI*, 27:1239–1253, 2005.

[3] I. Biederman, R. J. Mezzanotte, and J. C. Rabinowitz. Scene perception: Detecting and judging objects undergoing relational violations. *Cog. Psy.*, 14(2):143 – 177, 1982.

[4] M. Blane, Z. B. Lei, and D. B. Cooper. The 3L Algorithm for Fitting Implicit Polynomial Curves and Surfaces to Data. *TPAMI*, 22(3):298–313, 2000.

[5] X. Chen, A. Golovinskiy, and T. Funkhouser. A benchmark for 3D mesh segmentation. In *SIGGRAPH*, 2009.

[6] R. Fleming, M. Barnett-Cowan, and H. Bülthoff. Perceived object stability is affected by the internal representation of gravity. *Perception*, 39:109, 8 2010.

[7] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Manhattan-world stereo. In *CVPR*, 2009.

[8] A. Gupta, A. Efros, and M. Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *ECCV*, 2010.

[9] A. Gupta, S. Satkin, A. Efros, and M. Hebert. From 3D Scene Geometry to Human Workspace. In *CVPR*, 2011.

[10] J. Hamrick, P. Battaglia, and J. Tenenbaum. Internal physics models guide probabilistic judgments about object dynamics. In *Conf. Cog. Sc.*, 2011.

[11] V. Hedau, D. Hoiem, and D. Forsyth. Thinking inside the box: Using appearance models and context based on room geometry. In *ECCV*, 2010.

[12] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3-d object dataset: Putting the kinect to work. In *ICCV Workshop*, 2011.

[13] D. J. Kriegman. Let them fall where they may: Capture regions of curved objects and polyhedra. *International Journal of Robotics Research*, 16:448–472, 1995.

[14] D. Lee, A. Gupta, M. Hebert, and T. Kanade. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces advances in neural information processing systems. *Cambridge: MIT Press*, pages 609–616, 2010.

[15] D. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *CVPR*, 2009.

[16] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012.

[17] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*, 2011.

[18] J. Poppinga, N. Vaskevicius, A. Birk, and K. Pathak. Fast plane detection and polygonalization in noisy 3D range images. In *IROS*, 2008.

[19] R. Sagawa, K. Nishino, and K. Ikeuchi. Adaptively merging large-scale range data with reflectance properties. *TPAMI*, 27:392–405, 2005.

[20] D. Wales. *Energy Landscapes: Applications to Clusters, Biomolecules and Glasses*. Cambridge Molecular Science. Cambridge University Press, 2004.

[21] Y. Zhao and S. C. Zhu. Image parsing via stochastic scene grammar. In *NIPS*, 2011.

[22] B. Zheng, J. Takamatsu, and K. Ikeuchi. An Adaptive and Stable Method for Fitting Implicit Polynomial Curves and Surfaces. *PAMI*, 32(3):561–568, 2010.