Upright Orientation of Man-Made Objects

Hongbo Fu[1] Daniel Cohen-Or[2] Gideon Dror[3] Alla Sheffer[1]

[1]University of British Columbia [2]Tel Aviv University [3]The Academic College of Tel-Aviv-Yaffo

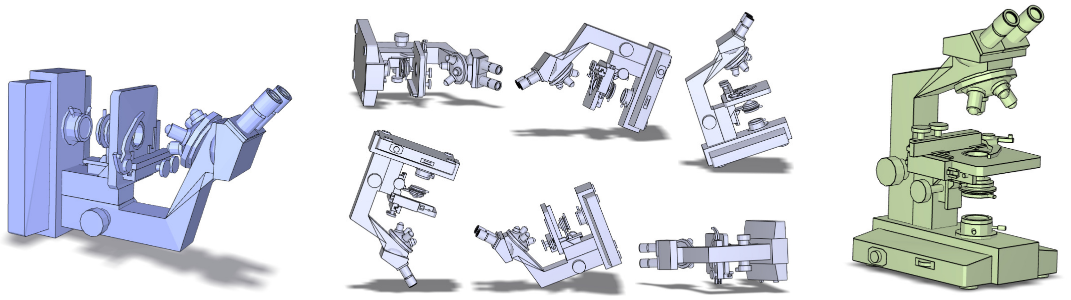


*Figure 1: **Left**: A man-made model with unnatural orientation. **Middle**: Six orientations obtained by aligning the model into a canonical coordinate frame using Principal Component Analysis. **Right:** Our method automatically detects the upright orientation of the model from its geometry alone.*

Abstract

Humans usually associate an upright orientation with objects, placing them in a way that they are most commonly seen in our surroundings. While it is an open challenge to recover the functionality of a shape from its geometry alone, this paper shows that it is often possible to infer its upright orientation by analyzing its geometry. Our key idea is to reduce the two-dimensional (spherical) orientation space to a small set of orientation candidates using functionality-related geometric properties of the object, and then determine the best orientation using an assessment function of several functional geometric attributes defined with respect to each candidate. Specifically we focus on obtaining the upright orientation for man-made objects that typically stand on some flat surface (ground, floor, table, etc.), which include the vast majority of objects in our everyday surroundings. For these types of models orientation candidates can be defined according to static equilibrium. For each candidate, we introduce a set of discriminative attributes linking shape to function. We learn an assessment function of these attributes from a training set using a combination of Random Forest classifier and Support Vector Machine classifier. Experiments demonstrate that our method generalizes well and achieves about 90% prediction accuracy for both a 10-fold cross-validation over the training set and a validation with an independent test set.

1 Introduction

A well-known postulate of design states that *form ever follows function* [Sullivan 1896]. It is unclear to what degree the opposite is true, namely how much of the object functionality can be deduced from its shape alone. While it is questionable whether a computer-based algorithm can effectively recognize a given shape or recover its functionality, this paper poses a more practical question: whether one can infer the upright orientation of a shape *from its geometry alone*. The upright orientation that humans naturally associate with an object is arguably linked to its everyday use or functionality [Blanz et al. 1999]. Without recognizing the full functionality of an object, analyzing and inferring its orientation are challenging tasks.

3D models created by various modeling and scanning systems often have different upright orientations, since models are always created in a particular context (e.g., a customized coordinate system). Given a model, automatically obtaining its upright orientation facilitates the exploration of the object from natural views (e.g., an intuitive fly-around view shown in the accompanying video). It also helps to generate easily recognizable thumbnail images of objects, useful for the management of large 3D shape repositories. An upright orientation reduces the number of degrees of freedom for object alignment in shape retrieval (see Section 2), and makes it easier for users to place objects during complex scene composition [Snyder 1995; Xu et al. 2002].

In this paper, we present an effective solution to the problem of automatic detection of upright orientation for commonly used 3D geometric shapes, which, to the best of our knowledge, has not been studied before. We observe that for many types of objects, searching for the best orientation in the whole (spherical) solution space is unnecessary. Instead, we can reduce the problem of shape orientation to the selection of the best orientation from a small set of orientation candidates. This reduction not only decreases the computational complexity but also better defines the problem. Obviously, the types of models to which an orientation algorithm is applicable are highly dependent on the selection of the candidate set. In this paper, we focus on *standing man-made* models, that is, models that are designed to stand on a flat surface and hence have well-defined upright orientations. This includes most objects in our everyday surroundings with a few exceptions, such as floating objects like ships or airplanes. We observe that the orientation of such an object is influenced by gravity and hence it must have a supporting base on which the object can be steadily positioned. These potential supporting bases naturally form a set of orientation candidates.

Given the candidate bases, a key challenge is to construct a set of quantitative geometric attributes for each candidate such that they are sufficiently discriminative to make the *natural base* stand out. The attributes must generalize across different types of man-made objects. We develop a set of geometric attributes that reflect functional design considerations as discussed in Section 4.2. Since functional considerations can often conflict, none of the attributes on its own has sufficient discrimination power for general upright orientation detection, requiring our system to combine multiple attributes in a principled way. We therefore use a supervised learning approach to design an assessment function combining all of the attributes to determine the most probable base (Section 5). We combine a Random Forest classifier and a Support Vector Machine classifier to learn the assessment function from a training set consisting of a large variety of standing man-made models, for each of which the upright orientation is given. Given a new unoriented model, we use the learned assessment function to compute a score for each of its candidate bases and choose the candidate with the highest score as the best base.

We evaluate the effectiveness of our method using both 10-fold cross-validation over a training set of 345 standing models and a validation with an independent test set of 819 standing models (Section 6). Experiments demonstrate that our method generalizes well and successfully predicts the upright orientations of most models with prediction accuracy of about 90% for both validations. Figure 1 illustrates an upright orientation detected with our method. An additional advantage of the method is that it can robustly estimate the reliability of its prediction. For example, limiting our predictions to the 30% most reliable predictions increases the performance to 95% accuracy.

2 Related Work

Pose Normalization. Pose normalization aims to align an object into a canonical coordinate frame. Principal Component Analysis (PCA) is the most commonly used technique for this task: the center of mass is chosen as the origin and the principal axes are chosen as the canonical axes [Duda et al. 2000]. However, it is well known that principal axes are not robust for pose normalization of many models [Kazhdan et al. 2003] and they certainly do not always produce compatible alignments with the upright orientations of objects, as illustrated by Figure 1. More important, PCA does not fully resolve the orientation problem, but simply reduces the orientation candidate set to six candidates. Podolak et al. [2006] demonstrate that principal symmetry axes are more robust than principal axes for the normalization of symmetric objects. Symmetry is a very strong cue for shape orientation in symmetric objects. However, it does not define the upright orientation but only reduces the likely orientation set. Moreover, not all man-made models are symmetric, nor does symmetry necessarily imply their upright orientations (Figure 6). We use symmetry as only one of multiple attributes with respect to each candidate base, relying on other considerations to obtain the unique upright orientation.

Viewpoint Selection. Automatically selecting good viewpoints for 3D models has always been important, especially for applications such as browsing a huge database of 3D models. Previous work either maximizes the visibility of interesting content using metrics like viewpoint entropy [Vázquez et al. 2003], view saliency [Lee et al. 2005] or shape distinction [Shilane and Funkhouser 2007], or minimizes visible redundant information such as symmetry [Podolak et al. 2006] or similarity [Yamauchi et al. 2006]. None of these methods considers shape orientation: due to the rotation-invariant metrics employed, they cannot distinguish between different orientations of a shape around a view direction. Recently, Saleem et al. [2007] proposed an exemplar-based method to correct the orientation of a projected shape in the 2D image plane associated with a given viewpoint (e.g., the best viewpoint). Our orientation method can help to determine the up direction of a camera for viewing an object in a natural way.

Image Orientation. Image orientation aims to automatically derive the upright orientation of a given image, typically among the four possible orientations of 0°, 90°, 180°, and 270° [Wang and Zhang 2004; Luo and Boutell 2005; Lumini and Nanni 2006]. Most existing techniques pose this problem as a four-class classification. They rotate the input image with respect to each possible orientation and represent each rotated image with a high-dimensional (on the order of thousands) feature vector [Wang and Zhang 2004; Lumini and Nanni 2006]. Each feature vector is then fed into a classifier, such as a Support Vector Machine and is assigned a score, with the highest score corresponding to the best orientation. It might be possible to adapt these image orientation methods to 3D shape orientation given an appropriate small set of orientation candidates. However, the high dimension of feature vectors would demand a very large training database of 3D models, much more difficult to provide than a training database of images.

3D Shape Retrieval. Given a query object, 3D shape retrieval systems retrieve objects with similar shapes from large databases of models (see [Tangelder and Veltkamp 2004; Iyer et al. 2005] and the references therein). The challenge here is to design a robust and efficient method for computing the similarity between two shapes over the space of all transformations [Kazhdan et al. 2003]. To address this challenge, most techniques align all of the models into a common coordinate frame before matching them, either automatically (typically based on PCA alignment) or manually. Since our orientation tool generates a consistent upright direction for models, it can reduce the orientation alignment problem from two to one degrees of freedom.

Stable Pose Estimation. Evaluating the stability of an object in contact with a supporting surface is useful for many tasks in machine vision and robotics, especially for the design of robotic parts feeders (see [Moll and Erdmann 2002] and the references therein). There are a number of methods for estimating the probability distribution of an object's stable poses, e.g., through maximizing the so-called capture regions [Kriegman 1997]. Those often rely on repeated dynamic simulation of an object under a random initial configuration. When the supporting surface is flat, the estimation can be equivalently performed on the convex hull of the object. Observing the usefulness of per-base stability for orientation identification, we introduce a simple yet effective static stability measure to avoid complex and often time-consuming dynamic simulation. We note that higher stable-pose probability does not necessarily imply a natural upright orientation (e.g., for the table model in Figure 3).

3 System Overview

Humans usually associate an upright orientation with objects, placing them in a way that they are most commonly seen in our surroundings. The ability to correctly orient objects seems related to objects' functionality and the viewers' familiarity with similar objects, suggesting that orientations of models are learnable. Supervised learning techniques allow us to exploit the fact that functionality is implicitly encoded in objects' geometry and that objects with similar functionality tend to have similar orientations.

Let us assume a training set of objects together with their prescribed upright orientations. The question is how the upright orientations of new objects can be inferred from the training set. The simplest way would be using the "by example" principle, that is, by matching the given object with one of the training examples and then by aligning them to determine the orientation. However, the task of matching shapes is highly complex and not reliable enough, and the alignment of two given shapes is a challenging task on its own.

Rather than directly discerning the orientation of a given object inside the full S^2 orientation space, we first reduce the problem to a discrete space of orientation candidates and then use a learning approach to select the best orientation. Since the objects that we are considering stand on a flat surface, each of them has a *natural base*,

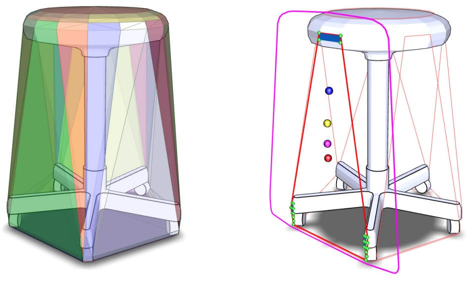


*Figure 2: **Left**: A stool model and its convex hull segmented into planar regions (shown in different colors). **Right**: The reduced set of supporting polygons (in red), corresponding to 7 candidate bases.*

namely a set of coplanar points on the model that touch the flat surface. Moreover, the base points lie on the convex hull of the model, forming some of its facets. Given an object, we thus first extract a set of candidate bases by analyzing the object's convex hull. The number of extracted candidates is typically fairly small, at most a dozen per object.

Each candidate base is associated with a feature vector of geometric properties. These are computed by analyzing the object geometry with respect to the candidate base and are related to functional considerations of the base's suitability. To make our method generalize well, we introduce properties that are abstracted from concrete shapes and tailored for orientation discrimination only. To learn the correlation between the feature vectors and the natural bases, we use a training set consisting of candidate bases associated with feature vectors and tagged with a binary label that indicates whether the candidate is a natural base or not.

The problem is now transformed into a supervised classification problem of finding the natural base given the features of the different candidates, for which there exists a wide variety of well-studied algorithms. The features extracted are not independent (for example we have three features related to symmetry) and have very different distributions. With these concerns in mind and following empirical testing we propose to combine a Random Forest classifier [Breiman 2001] and a Support Vector Machine classifier [Vapnik 1995], which together provide enhanced generalization performance. With the trained classifier, each candidate base of an object is scored and the upright orientation is finally determined by the candidate with the highest score.

4 Shape Analysis and Feature Extraction

This section describes the selection of candidate bases for a given object and then presents a set of attributes, defined with respect to each candidate base.

4.1 Selecting Candidate Bases

A candidate base is a set of coplanar points on the model lying on its convex hull. Since objects are acted on by gravity and the base has width, namely its points are not all collinear, each base defines a *supporting plane*. We define the *supporting polygon* of a base as the 2D convex hull of the base points in the supporting plane. Each base uniquely determines an upright direction as the normal of the supporting plane pointing toward our model. The supporting polygons correspond to faces of the object convex hull and hence are easily computed [1]. To merge coplanar hull facets and eliminate noise we simplify the convex hull using the Variational Shape Approximation method [Cohen-Steiner et al. 2004] before extracting the supporting polygons (Figure 2).

[1]We use CGAL [Fabri et al. 2000] to compute 2D and 3D convex hulls.

The set of candidate bases can be further reduced by taking into account stability considerations. For a rigid object in contact with a supporting plane and acted upon by gravity, the necessary condition for static equilibrium is that its center of mass lies over the supporting polygon. Therefore, we restrict the orientation candidates to the set of supporting polygons that lead to stable poses. We compute the center of mass as the weighted average of the face barycenters of the mesh. For each candidate base we check whether the projection of the center of mass to the supporting plane is inside the supporting polygon and discard the candidates for which the projection is outside. Figure 2 illustrates the orientation candidates of a stool.

4.2 Feature Extraction

Given the set of candidate bases, we assign them geometric attributes, which convey information that distinguishes the natural base from the other candidate bases. The attributes are derived using a combination of functional and aesthetic considerations. As the input meshes might suffer from poor sampling, inconsistent vertex/face normals, or isolated components, we cannot rely on attributes whose computation involves second- or higher-order surface derivatives, such as curvatures and feature lines. Note that each attribute should be appropriately normalized within individual objects, since the subsequent assessment-function learning needs to use attributes derived from different objects which may vary considerably in scale and proportions.

Static Stability. Most standing objects are designed to be reasonably stable with respect to small perturbing forces. It is commonly accepted that static stability is related to the proportion of the mass of the object located above the base. Since we have no knowledge of the materials involved, we use a geometric estimate of this property. We consider three geometric entities (illustrated in Figure 2) which, when combined, provide such a metric: the center of mass projected onto the supporting plane of the base (in yellow), the supporting polygon (in red), and the projection of the convex hull of the object to the supporting plane (in pink). The latter serves as a proxy for the projection of the object itself to the plane, which is more time-consuming to compute. Stability is typically better if the projected center of mass is far away from the boundary of the supporting polygon. Similarly, stability improves if the supporting polygon covers a large part of the convex-hull projection. Taking both considerations into account we measure *stability* as $\min_{0\leq\theta<2\pi} d_{inner}(\theta)/d_{outer}(\theta)$, where $d_{inner}(\theta)$ and $d_{outer}(\theta)$ are the distances from the projected center of mass to the boundary of the supporting polygon and the boundary of the projected hull respectively, along a direction determined by θ. This metric is naturally normalized to be between zero and one.

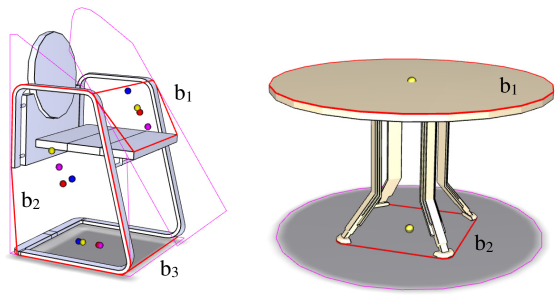


*Figure 3: **Left**: Candidate bases b_2 (side) and b_3 (bottom) have better stability than b_1 (front); b_1 and b_3 have better symmetry than b_2 (in terms of collinearity distance). **Right:** A negative example for the stability attribute: the natural base b_2 has worse stability than b_1. b_1 and b_2 have the same degree of symmetry (the four characteristic points coincide).*

Figure 3 shows examples of candidate bases with different degrees of stability. While stability is clearly important, it is not the only consideration in determining orientation. For instance, placing nearly every table upside down results in greater stability than placing it on its feet.

Symmetry. Symmetry plays an important role in human visual perception and as such provides a strong cue to upright orientation detection. As observed by Podolak et al. [2006], if objects exhibit reflective symmetry, the symmetry plane is typically vertical, or in our case, orthogonal to the natural base. Considering that direct symmetry computation [Mitra et al. 2006; Podolak et al. 2006; Simari et al. 2006] is time-consuming, we trade off precision for computational efficiency. Specifically, we reduce the symmetry estimation with respect to a given base to measuring distances, involving four characteristic points in the supporting plane. These four points, illustrated in Figure 2, are:

- ◇ the projected center of mass (in yellow);
- ◇ the barycenter (in red) of the supporting polygon;
- ◇ the barycenter (in pink) of the convex hull projection to the supporting plane;
- ◇ the barycenter (in blue) of the actual base (facets in blue and vertices in green).

We observe that perfect symmetry of a model with respect to a reflection plane perpendicular to the base implies collinearity of the above four characteristic points, since they all should lie *in* both the reflection plane and the supporting plane. Moreover, if there are multiple reflection planes, the four points must coincide.

According to these observations, we introduce two global symmetry-related attributes: the *coincidence distance* (the average distance of the four points to their centroid) and the *collinearity distance* (the average distance of the four points to their least-squares best-fit line). The attributes are normalized by the radius of the bounding sphere of the model. We also note that for many models, the natural base itself is symmetric. Consider, for example, the legs of a chair, a base of a vase, and so on. In our setup *base symmetry* can be estimated by the distance between the barycenters of the actual base and the supporting polygon. This distance effectively measures the deviation of the base from uniform mass distribution. To obtain a scale-invariant metric this distance is also normalized by the radius of the model's bounding sphere. Symmetry is extremely useful in discarding relatively bad candidates (e.g., side bases of the stool model in Figure 2, side base b_2 of the chair model in Figure 3), but used alone it does not distinguish well between candidate bases orthogonal to the reflection plane(s) (e.g., bases b_1 and b_3 of the chair model in Figure 3).

Parallelism. Psychophysical experiments have indicated that humans tend to integrate local orientation information to identify the global orientation of an object [Saarinen et al. 1997]. That is, the global orientation is highly connected to the local orientations of the model surface. We observe that having large areas of the surface parallel to the base is often an indicator of a natural base. Thus we introduce a *face parallelism attribute* measured as the area of model faces that are parallel to the supporting plane, for example, the faces in green in Figure 4. We normalize this attribute by the total area of the model. We also notice that for many models, there is a clear vertical axis, with both the top and the bottom planes orthogonal to it, being good candidate bases. This is reflected by our second parallelism-related metric, *base parallelism*, which measures the area of other supporting polygons that are parallel to the reference supporting polygon. This attribute is normalized by the area of the convex hull of the object.

Visibility. As object geometry often reflects its function, we may reasonably assume that the surface of the object exists for a reason. When an object is placed on a supporting plane, part of it becomes inaccessible and is no longer visible. For instance, consider

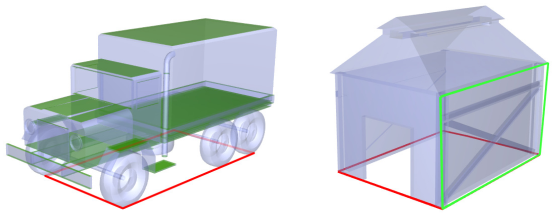


*Figure 4: **Left**: An illustration of face parallelism for the bottom supporting polygon. **Right**: A negative example for the face parallelism attribute: the natural base (in red) has worse face parallelism than the side candidate (in green).*

an upside-down cup: the inside surface of the cup is no longer visible. Similarly, the large surface of the top of an upside-down table is no longer visible. Thus a candidate base is more likely to be optimal if much of the model remains visible after the object is placed on the corresponding supporting plane. To measure visibility we estimate the occlusion caused when adding the supporting polygon to the model. We employ a similar approach to [Sheffer and Hart 2002] to efficiently compute *per-face* visibility. We render the model from 32 uniformly sampled view directions with each face assigned a distinct color and then count the number of hits per face. Faces which are seen in less than a fraction of the views (0.05 was used in our experiments) are considered occluded (e.g., the faces in red in Figure 5). The *per-base* visibility is computed as the sum of areas of visible faces. We normalize this attribute by the mesh visibility without introducing any supporting polygon as an occluder. We make this choice because in some models parts of the model can be hidden in all views.

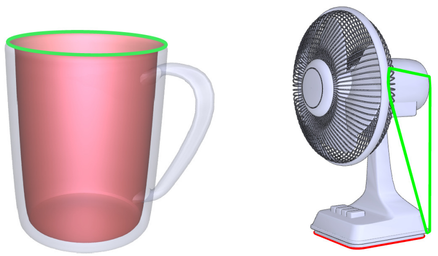


*Figure 5: **Left**: An illustration of the visibility map when introducing the top supporting polygon as an occluder. **Right:** A negative example for the visibility attribute: the natural base (in red) has worse visibility than the side candidate (in green).*

Another attribute that helps to distinguish orientations is the base area, normalized by the entire mesh area. This attribute can be considered visibility-related, since the actual base is occluded when the object is positioned on the corresponding supporting polygon.

Collecting these attributes, we obtain vectors of 8 features per candidate base, which are passed to our learning algorithm described next.

5 Assessment Function Learning

One of the best ways for constructing an assessment function is learning it from examples. To this end we use a supervised learning algorithm, which is able to tune function parameters based on the statistics of a training set. The latter is composed of a set of models with multiple candidate bases and is described in the next section. The vast majority of the models have a single natural base. Therefore, we adopt a representation where the set of candidate bases per model contains a single "correct" base. Notice that this constraint renders the problem into a structured estimation problem.

Preliminary experimentation with exponential models [Wainwright and Jordan 2003], where the output space was designed to be compatible to the above constraint gave mediocre results. We therefore turned to classification algorithms, which produce classifiers that score each candidate base in isolation, without explicitly assigning a natural base to a model. Instead, the base with the highest score is selected as the natural base. Since the attributes are interdependent, linear methods are not expected to perform well. Indeed naïve Bayes, linear regression [Duda et al. 2000] and linear Support Vector Machine (SVM) [Vapnik 1995] all produced unsatisfactory performance. We thus resorted to more complex models and combined a Random Forest classifier and an SVM classifier with a polynomial kernel.

5.1 Random Forest

A Random Forest (RF) [Breiman 2001] is made up of an ensemble of decision trees. The individual trees are trained using a variant of the CART (Classification and Regression Tree) algorithm [Duda et al. 2000]. Each tree is grown as follows: a *bootstrap* sample of n examples, where n is the size of the training set, is sampled with replacement from the training set. Due to the sampling, some training examples are typically represented in the bootstrap sample more than once whereas other examples are missing altogether. A tree of maximal depth is grown on the bootstrap sample. The examples not used for growing the tree, called *out-of-bag* examples, are used to give an unbiased estimate for the classification error of each individual tree, as well as to get an estimate for variable importance. The bootstrap sample is unique per individual tree learning, thus producing a wide variety of decision trees. The final classification of a test example is usually given by majority voting of the ensemble. Furthermore, one may also interpret the fraction of votes per output class as an estimate of its likelihood. The fact that a large number of different trees are voted makes Random Forest resistant to overfitting.

5.2 Support Vector Machine

SVM has been used extensively for a wide range of classification, regression and ranking applications in science, medicine and engineering and has shown excellent empirical performance. SVM has several advantages for the present task. First, it is based on the principle of risk minimization and thus provides good generalization control. Second, using nonlinear kernels, SVM can model nonlinear dependencies among features, which may prove advantageous for the problem at hand. Third, SVM allows natural control on the relative cost of false positives and false negatives. This is quite desirable in our case, since the number of natural bases per model is much smaller than the number of remaining candidate bases. Here we used soft-margin SVM implemented in SVM^{light} [Joachims 1999].

5.3 Combined Classifier

We observe that used on their own the Random Forest classifier and the SVM classifier produce comparable results, with slight advantage to the former. SVM was trained using a quadratic kernel (polynomial kernel of degree 2) and RF was trained using a large number of trees (= 2000) whereas all other parameters attained their defaults values. Further analysis shows two interesting facts. First, the models for which the classifiers err are quite distinct. Second, for certain models, the Random Forest classifier assigns zero score for all the candidate bases. This happens when no candidate base of the model appears to be a typical natural base, and hence none of the trees votes in favor of it.

We take advantage of these two observations and combine the Random Forest and the quadratic SVM classifiers as our final classifier. Formally, the combined classifier is formulated as:

$$Y_{combined}(\mathbf{f}) = \alpha\, Y_{rf}(\mathbf{f}) + (1-\alpha)\, Y_{svm}(\mathbf{f}), \tag{1}$$

where $\mathbf{f} \in \mathbb{R}^8$ is a feature vector of attributes described in the previous section, $Y_{rf}(\mathbf{f}) \in [0,1]$ is the raw output of the Random Forest classifier, and $Y_{svm}(\mathbf{f}) = 1/(1+\exp(-o_{svm}))$ with o_{svm} being the raw output of the quadratic SVM classifier. We use α as a mixing parameter, optimized by performing a linear search using cross-validation. α is 0.6 in our experiments. This combination results in considerable improvement in performance, compared to standard RF or SVM alone.

6 Implementation and Results

In this section, we describe the construction of the training set for assessment function learning and the validation results.

6.1 Training

Our learning algorithm does not rely on the classification of training models. However, having many unbiased classes of models does help the learning process to better understand the essence of orientation identification among different kinds of models, thus boosting generalizability. We construct a training set of standing man-made models using the Princeton Shape Benchmark (PSB) [Shilane et al. 2004], a publicly available database of polygonal models. For training purposes, we select from it a representative sample of models covering the major classes of man-made models present in PSB. In all, there are 345 unique training models with around five candidate bases per model. The candidate bases of each training model are manually labeled as natural or not.

6.2 Validation

We validate the effectiveness of our method using the prediction accuracy, which measures the ratio of the number of models correctly oriented to the number of models in set. Note that for a model with multiple natural bases (e.g., the bottom and top candidate bases of the bookshelf in Figure 6 are both considered natural), a correct detection means that *any* of these bases is identified.

	SVM	RF	Combined Classifier
Training	81.9%	83.4%	**90.5%**
Test	78.6%	85.3%	**87.5%**

Table 1: Prediction accuracies with different learning methods.

K-fold Cross-Validation. We perform a 10-fold cross-validation on the the training set. Achieving high prediction accuracy on the training set is not an easy task, as demonstrated when using individual attributes to identify upright orientations. For example, with the stability attribute alone, the prediction accuracy using SVM alone is only 50.1%[2]. Our combined classifier with all the attributes achieves 90.5% prediction accuracy, compared to 83.4% with RF alone and 81.9% with quadratic SVM alone.

Validation with Independent Test Set. We also use an independent test set to validate our method. The test set consists of standing models, including models from PSB excluded from the training set. We introduce additional test models taken from well-known model repositories (e.g., Google 3D Warehouse, Turbo Squid, 3D Cafe). We check all the test models to guarantee that there is no duplication between the training and test sets. We tested 819 models with on average 5.75 candidate bases per model. We use the assessment function trained with the whole training set to detect the orientations of the test models. Table 1 compares the prediction accuracies with different learning methods.

Our algorithm is highly generalizable. Not only is it able to orient diverse objects within function classes present in the training set, such as tables or cars, but is successfully applicable to objects

[2]It is impossible to run the combined classifier on a single feature, since RF needs more features to train.

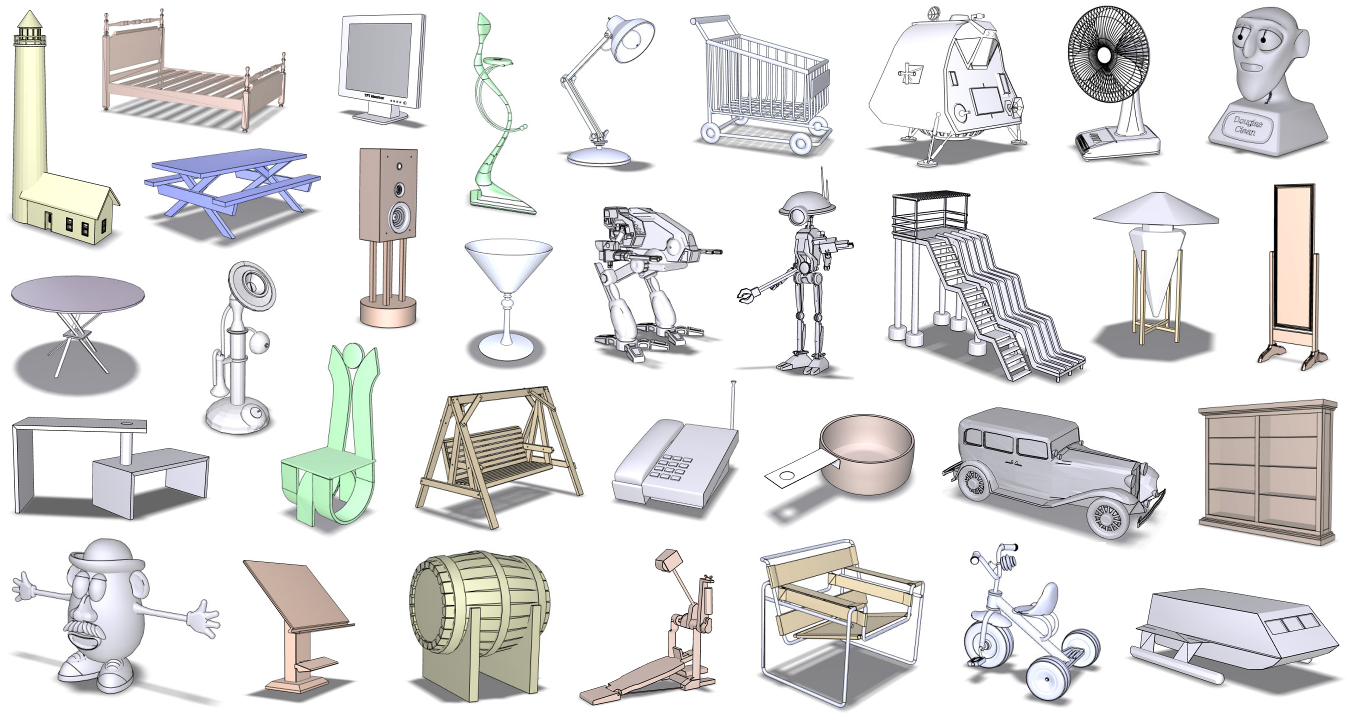
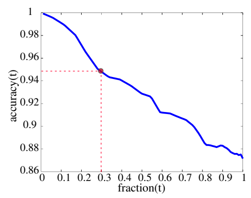


Figure 6: Examples of correct upright orientations found by our method.

that belong to other function classes. For example, even though the training set contains no humanoid models, almost all the humanoid figures in the test set are oriented correctly. Figures 1 through 6 show dozens of models correctly oriented by our algorithm. See the accompanying video and supplemental images for the whole training and test sets with the failed models highlighted. Note that although our feature set is specifically chosen for man-made objects, our algorithm also works well for certain natural objects (e.g. 4-legged animals and human figures in rest pose).

As with other learning based approaches, it is difficult to precisely analyze the failure modes of our method. However, inspecting the models on which it failed, we could suggest several possible reasons for failure. In around 20% of the failed cases more than one orientation is plausible and only subtle details indicate the natural orientation. Most of those models have box-like shapes (e.g., refrigerator and cabinet models). In addition, for about 30% of the models (e.g., the bed and sofa models in Figure 7) geometric features do not seem to provide enough information to distinguish between plausible orientations. We speculate that geometry alone might not be sufficient in these cases. We found that about 5% of the models have representation problems (e.g., one-sided representation, holes, and noise), leading to incorrect visibility computation or unstable convex-hull computation. We speculate that the rest of models failed mainly due to the strong conflict among the current features (e.g., between the visibility and stability features for the mirror model in Figure 7), generally demanding new features.

Confidence Dependent Performance. For many applications estimating the confidence of prediction could considerably enhance their practicality. For example, in cases when prediction is estimated to be less reliable, manual intervention can be sought. The figure to the right illustrates the performance of our classifier for various levels of prediction confidence on the test set. Specifically, for a given score level t, we calculate the prediction accuracy on a subset $S(t)$ of models, where the highest score of their candidate bases is larger than the threshold t. For simplicity, the accuracy is plotted against the relative size of $S(t)$ (namely $S(t)/S(0)$), so that the rightmost point on the graph corresponds to all models (no rejection) whereas the leftmost point corresponds to maximal rejection, i.e. only models with the most reliable predication. Limiting our predictions to the 30% most reliable predictions increases the performance to about 95% accuracy.



Avg. # Vertices	Avg. # Faces	Candidate Selection	Feature Extraction	Testing	Total
17,148	22,358	7.09s	3.53s	2.2ms	10.62s

Table 2: Average model statics and timing.

Timings. Table 2 gives the model statistics and detailed timings of our experiments, measured on an Intel Core 2 2.13GHz desktop with 2GB of RAM using a single thread implementation. The timings are computed as average times per model. Computing the object convex hulls is the bottleneck of candidate selection. It can be accelerated by simplifying the models being processed in advance. The bottleneck of feature extraction is the visibility computation, which can be sped up using graphics hardware implementation. After training set construction and labeling, it took our method 12.00 seconds to train the assessment function on the entire set.

7 Conclusion and Future Work

This work addresses for the first time, to the best of our knowledge, the problem of computing the upright orientation of 3D models. We provide an effective solution to this problem, in the case of standing objects, by selecting the best orientation from a small set of candidates determined from model geometry alone. This selection process is readily learnable from a training set of models with a labeled feature vector of geometric attributes defined with respect to each candidate. We have demonstrated that the necessary geometric attributes for orientation discrimination are largely unrelated to specific model shapes, thus making our method highly generalizable and leading to high prediction accuracy for many kinds of

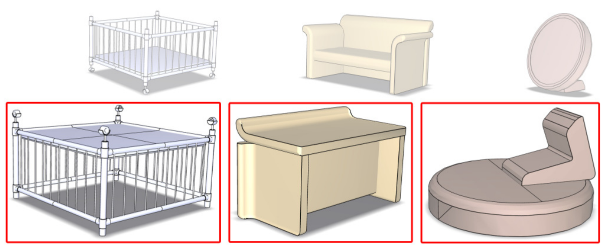


Figure 7: Examples of models oriented incorrectly by our method. We misorient only about 10% of the models tested.

models. Theoretically, it might be possible to further improve prediction accuracy by exploring additional geometric attributes. This paper focuses on orientation detection for standing man-made models. It will be very interesting to find similar ways to orient other types of man-made or even natural shapes by using different methods to form orientation candidates. For example, it might be possible to employ (visually) dominant line and plane features to define candidate directions in these cases.

Acknowledgements

We would like to thank the reviewers for their valuable comments, Derek Bradley and Ian South-Dickinson for their great help during video production, and Wei-Lwun Lu, Jia Chen and Kai Zhang for the illuminating discussion on the learning method. This work was partially supported by grants from NSERC, MITACS NCE, GEOIDE NCE, National ICT Australia, the Israeli Ministry of Science, and the Israel Science Foundation.

References

BLANZ, V., TARR, M. J., AND BÜLTHOFF, H. H. 1999. What object attributes determine canonical views? *Perception 28*, 5, 575–599.

BREIMAN, L. 2001. Random forests. *Machine Learning 45*, 1, 5–32.

COHEN-STEINER, D., ALLIEZ, P., AND DESBRUN, M. 2004. Variational shape approximation. *ACM Trans. Graph. 23*, 3, 905–914.

DUDA, R. O., HART, P. E., AND STORK, D. G. 2000. *Pattern Classification, Second Edition*. Wiley-Interscience, October.

FABRI, A., GIEZEMAN, G.-J., KETTNER, L., SCHIRRA, S., AND SCHÖNHERR, S. 2000. On the design of CGAL, a computational geometry algorithms library. *Softw. Pract. Exper. 30*, 11, 1167–1202.

IYER, N., JAYANTI, S., LOU, K., KALYANARAMAN, Y., AND RAMANI, K. 2005. Three-dimensional shape searching: state-of-the-art review and future trends. *Computer-Aided Design 37*, 5, 509–530.

JOACHIMS, T. 1999. Making large-scale support vector machine learning practical. In *Advances in Kernel Methods: Support Vector Machines*. MIT Press.

KAZHDAN, M., FUNKHOUSER, T., AND RUSINKIEWICZ, S. 2003. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *SGP '03*, 156–164.

KRIEGMAN, D. J. 1997. Let them fall where they may: capture regions of curved objects and polyhedra. *Int. J. Rob. Res. 16*, 4, 448–472.

LEE, C. H., VARSHNEY, A., AND JACOBS, D. W. 2005. Mesh saliency. *ACM Trans. Graph. 24*, 3, 659–666.

LUMINI, A., AND NANNI, L. 2006. Detector of image orientation based on Borda Count. *Pattern Recogn. Lett. 27*, 3, 180–186.

LUO, J., AND BOUTELL, M. 2005. Automatic image orientation detection via confidence-based integration of low-level and semantic cues. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 5, 715–726.

MITRA, N. J., GUIBAS, L. J., AND PAULY, M. 2006. Partial and approximate symmetry detection for 3D geometry. *ACM Trans. Graph. 25*, 3, 560–568.

MOLL, M., AND ERDMANN, M. 2002. Manipulation of pose distributions. *The International Journal of Robotics Research 21*, 3, 277–292.

PODOLAK, J., SHILANE, P., GOLOVINSKIY, A., RUSINKIEWICZ, S., AND FUNKHOUSER, T. 2006. A planar-reflective symmetry transform for 3D shapes. *ACM Trans. Graph. 25*, 3, 549–559.

SAARINEN, J., LEVI, D. M., AND SHEN, B. 1997. Integration of local pattern elements into a global shape in human vision. *Proc. Natl. Acad. Sci. USA 94*, 8267–8271.

SALEEM, W., WANG, D., BELYAEV, A., AND SEIDEL, H.-P. 2007. Automatic 2D shape orientation by example. In *SMI '07*, 221–225.

SHEFFER, A., AND HART, J. 2002. Seamster: inconspicuous low-distortion texture seam layout. In *IEEE Visualization*, 291–298.

SHILANE, P., AND FUNKHOUSER, T. 2007. Distinctive regions of 3D surfaces. *ACM Transactions on Graphics 26*, 2, 7.

SHILANE, P., MIN, P., KAZHDAN, M., AND FUNKHOUSER, T. 2004. The Princeton shape benchmark. In *SMI '04*, 167–178.

SIMARI, P., KALOGERAKIS, E., AND SINGH, K. 2006. Folding meshes: hierarchical mesh segmentation based on planar symmetry. In *SGP '06*, 111–119.

SNYDER, J. M. 1995. An interactive tool for placing curved surfaces without interpenetration. In *SIGGRAPH '95*, 209–218.

SULLIVAN, L. H. 1896. The tall office building artistically considered. *Lippincott's Magazine*.

TANGELDER, J., AND VELTKAMP, R. 2004. A survey of content based 3D shape retrieval methods. In *Shape Modeling Applications*, 145–156.

VAPNIK, V. N. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA.

VÁZQUEZ, P.-P., FEIXAS, M., SBERT, M., AND HEIDRICH, W. 2003. Automatic view selection using viewpoint entropy and its application to image-based modelling. *Computer Graphics Forum 22*, 4, 689–700.

WAINWRIGHT, M. J., AND JORDAN, M. I. 2003. Graphical models, exponential families, and variational inference. Tech. Rep. TR 649, Department of Statistics, UC Berkeley.

WANG, Y. M., AND ZHANG, H. 2004. Detecting image orientation based on low-level visual content. *Computer Vision and Image Understanding 93*, 3, 328–346.

XU, K., STEWART, J., AND FIUME, E. 2002. Constraint-based automatic placement for scene composition. In *GI '02*.

YAMAUCHI, H., SALEEM, W., YOSHIZAWA, S., KARNI, Z., BELYAEV, A., AND SEIDEL, H.-P. 2006. Towards stable and salient multi-view representation of 3D shapes. In *SMI '06*, 265–270.