Content Based Retrieval of VRML Objects – An Iterative and Interactive Approach

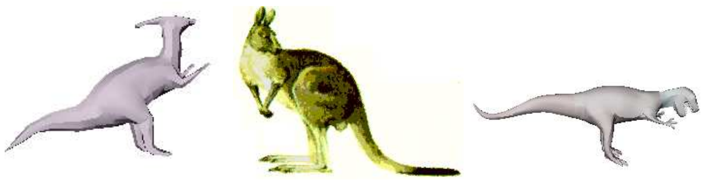
Michael Elad[1], Ayellet Tal[2], and Sigal Ar[2]

[1] HP Laboratories
Haifa, Israel
elad@ee.technion.ac.il
[2] Department of Electrical Engineering
Technion – Israel Institute of Technology
Haifa, Israel
{ayellet, ar}@ee.technion.ac.il

Abstract. We examine the problem of searching a database of three-dimensional objects (given in VRML) for objects similar to a given object. We introduce an algorithm which is both iterative and interactive. Rather than base the search solely on geometric feature similarity, we propose letting the user influence future search results by marking some of the results of the current search as 'relevant' or 'irrelevant', thus indicating personal preferences. A novel approach, based on SVM, is used for the adaptation of the distance measure consistently with these markings, which brings the 'relevant' objects closer and pushes the 'irrelevant' objects farther. We show that in practice very few iterations are needed for the system to converge well on what the user "had in mind".

1 Introduction

The problem of automatically comparing objects and detecting which ones are alike is a difficult problem. After all, "similarity is in the eye of the beholder". Objects' similarity is a subjective matter, dependent on the human viewer, since objects have semantics and are not only geometric entities. Does a parasaurolophus (Figure 1) look more like a kangaroo or more like an allosaurus?



A Parasaurolophus   A Kangaroo   An Allosaurus

Fig. 1. Which one is more alike?

Finding similarity between geometric objects has been a lively topic of research in computational geometry [5, 22, 29]. Objects are considered similar if their geometric features are close, given a metric measuring the distance [6]. A lot of work has been done matching sets of points in two dimensions under various transformation sets [8, 9, 28]. The *Hausdorff distance* has been frequently used to measure the distance between point sets [18, 19].

Polygonal shapes have also been considered. Algorithms for computing the minimum Hausdorff distance between polygons were proposed by [2, 3, 12, 20]. Hausdorff distance has some limitations as it is not robust for outliers. Comparing polygons as turning functions is proposed in [7]. In [4] the *Frechet distance* is used to compare polygons. In [17], a *reflection metric* is defined and computed for two unions of line segments.

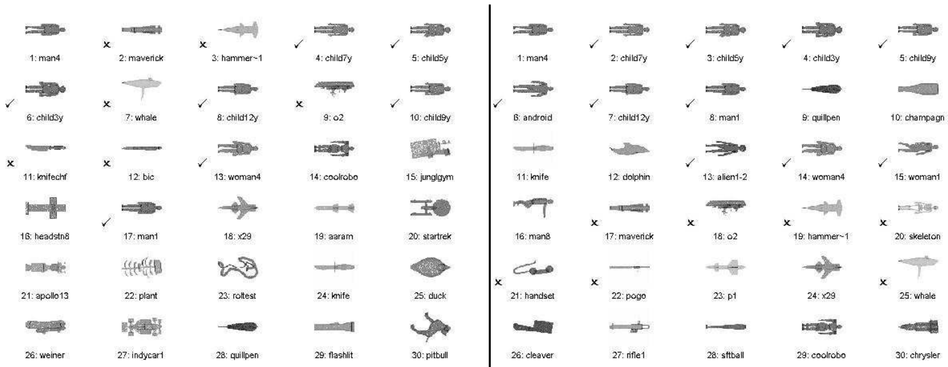
Less work has been invested in the three-dimensional case [11, 13, 20, 26]. In particular, extending methods of comparing polygonal curves in two dimensions to higher dimensions is non-trivial, both in theory and in practice [23]. However, as VRML objects are becoming more popular on the World-Wide Web, this problem is expected to have many applications, one prominent example being in e-commerce.

Much of the research on similarity has been done in the context of images [24, 1]. On one hand, finding similarities between three-dimensional models seems like it should be easier than finding similarities between their projections to images. After all, the whole object can be "seen", thus occlusions, self-occlusions, lighting effects and reflections, are avoided. On the other hand, three-dimensional models can be harder to handle since they do not have a simple parameterization, and registration and feature correspondence are more difficult to find.

In this paper, we focus on finding similarities between three-dimensional geometric objects. Rather than dealing solely with the choice of features and the definition of a distance measure on these features, we want to give the user the added ability of influencing the search as it is being conducted by means of relevance feedback [14, 21, 25]. We suggest an iterative method where each user can specify how relevant the results are according to this user's preferences.

The scheme we propose is as follows. Given an object, the system searches a database for similar ones. Once a set of results is obtained, the user is presented with the best (most-similar) objects and is given a chance to mark a subset of them as 'relevant' or as 'irrelevant'. Using this feedback from the user, the distance function is updated and another iteration of the search may be conducted. A new set of results, of the updated search, is presented to the user to mark and re-iterate the process, if need be. Thus, with the same database and the same object, each user may get different objects as the closest to the chosen one.

Figure 2 illustrates the intent of the algorithm, as well as some results obtained by the system implementing it. In Figure 2(a), an initial set of the top thirty similar objects is presented, when a search for figures similar to "man4" is conducted on a database of more than a 1000 VRML objects. After the user marked a subset of the results as relevant or irrelevant, and a couple of iterations were run, the final set of results is produced and shown in Figure 2(b). The latter set of results can be contrasted with the initial set. Not only more human figures were found among the top thirty, but also their ranking among the top thirty has been improved. Moreover, figures which differ geometrically, such as 13 ("allien2") where the legs are open, and 16 ("man8") where one leg is folded and an arm is pointing forward, were selected and reflect sematic similarity.



(a) Initial results (b) Final results

Fig. 2. Searching the database for objects similar to 1:man4

Three main issues are tackled in this paper. The first issue is the choice of features. The goal is to have a representation which is compact, suffices to uniquely identify each object, and reflects similarities (and dissimilarities) between objects. Such features can be considered as a *signature* associated with each three-dimensional object, just as a few keywords are associated with each document and are used when searching large databases or the web. The second main issue is the choice of a distance measure which should not only reflect object similarities/dissimilarities well, but should also be amenable to adaptations, to enable refinement of searches. The third issue is finding an adaptation rule on the distance measure.

We will show that the moments of the three-dimensional objects' surfaces, up to some order, are a relevant choice for features as only a handful of them are needed to represent the essence of the data. We will also show that the weighted Euclidean distance leads to an efficient and effective adaptation scheme, based on the user's feedback. Finally, based on the user's input, we find the optimal weights for the weighted Euclidean metric in a way which reflects the user's desire and warps the space such that the 'relevant' results are brought closer and the 'irrelevant' results are pushed farther. Using ideas based on Support Vector Machine learning algorithms (SVM) [30], we formulate an optimization problem in which the weights are the unknowns and their optimal values correspond to the maximal margin between the distances. The obtained optimization problem is convex (QP), thus ensuring a unique solution.

We view the contributions of this paper to be threefold. First, We propose a novel algorithm for retrieving three-dimensional VRML objects, an area which is relatively unexplored. Second, we propose to use moments as features representing three-dimensional objects, and in particular we show how to use moments within a relevance feedback scheme. While moments can have problems in pictures due to occlusions and self-occlusions, they can very well characterize three-dimensional models which are given wholly. Last but not least, we show how to use relevance feedback in the context of SVM.

We tested the algorithm with a database containing over a 1000 objects, given in VRML. Simulations of the search mechanism exhibit very promising results. With very few iterations (usually, 1–4), the search process converges on what sample users had in mind for these searches.

The rest of this paper is organized as follows. Section 2 describes the object features we use. Section 3 describes the similarity measure and the actual iterative search process. In Section 4 we show some runs of the algorithm on a database of three-dimensional VRML objects, and explain the workings of the algorithm by way of demonstration. Finally, Section 5 concludes the paper.

2 Object Features

We assume an object is given in VRML, i.e. it is a three-dimensional object represented by a set of vertices and a set of polygonal faces embedded in three dimensions. The features we choose to represent the objects are the moments, computed for object surfaces, assuming all objects are hollow. For object D, we denote the surface by ∂D. D's (p, q, r)-th moment is the given by

$$m_{pqr} = \int_{\partial D} x^p y^q z^r dxdydz \ . \tag{1}$$

The set of moments $\{m_{pqr}\}$ have a property of fundamental importance: they uniquely determine, and are uniquely determined by, the object. Thus, the triple sequence of moments constitutes a full and complete object description, and a partial object description can be obtained by using some subset of them [16].

Sampling to Approximate the Moments

At the crux of our algorithm lies the computation of a subset of the (p, q, r) moments of each object, which are used as the feature set. Thus, we perform a pre-processing stage where the features are calculated for each database object.

A practical way to evaluate the integral defining moments is to compute it analytically for each facet of the object, and then sum over all the facets. We use an alternative approach, yielding an approximation of the moments. The algorithm draws a sequence of points, (x, y, z), distributed uniformly over the object's surface. The number of points drawn from each of the object's facets is proportional to its relative surface area. If we denote the list of points for a given object by $\{x_i, y_i, z_i\}_{i=1}^{N}$, the (p,q,r)-th moment is then approximated by

$$\hat{m}_{pqr} = \frac{1}{N}\sum_{i=1}^{N} x_i^p y_i^q z_i^r \tag{2}$$

Normalizing the Objects

We want the similarity measure to be invariant to spatial position, scale and rotation of the different objects. We therefore need to normalize the feature vectors of all objects.

The first moments m_{100}, m_{010} and m_{001} represent the object's center of mass. Thus, the normalization starts by estimating the first moments for each object represented as a set of surface sample points, and subtracting them from each of these points:

$$\forall i = 1, 2, \ldots, N \;\; [x_i, y_i, z_i]^T \leftarrow [x_i - \hat{m}_{100}, y_i - \hat{m}_{010}, z_i - \hat{m}_{001}]^T \; . \tag{3}$$

This amounts to positioning all objects so that their center of mass is at coordinates $(0,0,0)$, thus removing any dependence on translation, or spatial position. This also sets each of $\hat{m}_{100}$, $\hat{m}_{010}$ and $\hat{m}_{001}$ to 0 for all objects, and thus renders them useless for further computations.

The second moments – m_{200}, m_{020}, m_{002}, m_{110}, m_{101} and m_{011} – represent the object's rotation and scale in the following manner. When the second moments, calculated for the object re-centered at $(0,0,0)$, are ordered into a matrix:

$$M = \begin{bmatrix} m_{200} & m_{110} & m_{101} \\ m_{110} & m_{020} & m_{011} \\ m_{101} & m_{011} & m_{002} \end{bmatrix} \tag{4}$$

and Singular Value Decomposition is performed, the result may be written as:

$$U \Delta U^T = SVD(M) \; , \tag{5}$$

where the unitary matrix U represents the rotation and the diagonal matrix Δ represents the scale in each axis, ordered in decreasing size.

The normalization continues with a second stage approximating the second moments for each object, by computing them from the updated surface point data sets, using equation 2, into $\hat{M}$. After performing the SVD decomposition of the second moment matrix $\hat{M}$, we multiply each point by U to rotate the object back to a canonic position. We also divide each point by $\Delta(1,1)$ to rescale the object so that its largest scale is 1. To summarize, each point is replaced by

$$[x_i, y_i, z_i]^T \leftarrow \frac{1}{\Delta(1,1)} \cdot U \cdot [x_i, y_i, z_i]^T \; . \tag{6}$$

Finally, the algorithm should also determine each object's orientation, relative to each axis. To do this, we count the number of points on each side of the center of the body. In order to normalize such that all the objects have the same orientation, we flip each object so that it is "heavier" on the positive side. In counting the number of points and flipping according to it, we are actually forcing the median center to be on a predetermined side relatively to the center of mass.

After applying all the normalization stages to each object, the moments are computed once more, up to the pre-specified order. As we indicated, the normalization fixed $\hat{m}_{100}$, $\hat{m}_{010}$, $\hat{m}_{001}$ and $\hat{m}_{200}$ to 0, 0, 0 and 1, respectively, for each and every object. These are therefore no longer useful as object features.

3 Iterative Refinement

Having two finite sets of moments in vectors X and Y, constituting partial descriptions of database objects D_X and D_Y respectively, we can measure the distance between the objects using (the square of) the Euclidean distance

$$d(D_X, D_Y) = ||X - Y||^2 \tag{7}$$

Using the Euclidean distance alone, the automatic search of the database will indeed produce objects that are geometrically close to the given one. However, these may not be what the human user had in mind when initiating the search. Therefore, we employ further "parameterization" of this distance by adding weights and a bias value:

$$d(D_X, D_Y) = [X - Y]^T W [X - Y] + b \; . \tag{8}$$

where W may be any matrix, yet in the following we assume it is a diagonal matrix.

Given a set of search results, a human user may consider some of them relevant and some of them irrelevant, in spite of them all being geometrically close. The adaptation of the distance function can be done by recalculating distances, based on the user preferences. The additional requirement is that the new distance between the given object and the relevant results be small and, obviously, the new distance between the given object and the irrelevant results needs to be large.

In essence, this is a classification, or a learning problem. A way to formulate the requirements is by defining weights on the components of the distance function and writing a set of constraints. Denote the vector moments of the object for which the system is to search by O, the vectors of moments of the 'relevant' results by $\{G_k\}_{k=1}^{n_G}$ and the vectors of moments of the 'irrelevant' results by $\{B_l\}_{l=1}^{n_B}$. The constraints posed on the weight function are then

$$\begin{aligned} k = 1, 2, \ldots, n_G,\ d(D_O, D_{G_k}) &= [O - G_k]^T W [O - G_k] + b \leq 1 \\ l = 1, 2, \ldots, n_B,\ d(D_O, D_{B_l}) &= [O - B_l]^T W [O - B_l] + b \geq 2 \; . \end{aligned} \tag{9}$$

This creates a margin between the 'relevant' and 'irrelevant' results. The above inequalities are linear with respect to the entries of W.

Denoting the main diagonal of W by ω, we may rewrite the constraints as

$$k = 1, 2, \ldots, n_G, \quad d(D_O, D_{G_k}) = [O - G_k]^2 \cdot \omega + b \leq 1 \tag{10}$$

$$l = 1, 2, \ldots, n_B, \quad d(D_O, D_{B_l}) = [O - B_l]^2 \cdot \omega + b \geq 2 \ ,$$

where the notation V^2, for a vector V, means multiplying each vector entry by itself.

An additional constraint is that the entries of W are all non-negative. Note though, that we do not require b to be non-negative, and may therefore end up with a non-metric similarity measure.

It can be shown (see for example [27, 10, 15, 30]), that the maximal margin of separation between the two sets of results is achieved by the ω with the smallest (square of the) norm – $min_\omega(||\omega||^2)$. Choosing the ω with the smallest norm also renders the solution to the constraint system robust to the number of examples from each of the two subsets – 'relevant' and 'irrelevant', and also the size of the rest of the database. This is good when the above constraints are insufficient, when $n_G + n_B << \mathcal{M}$, $\mathcal{M}$ being the arity of the feature vectors. That is, there are more unknowns than inequalities, and therefore multiple possible solutions $\{\omega, b\}$ all satisfying the constraints.

Thus, at each refinement iteration we essentially need to solve the following for ω:
Minimize $\quad ||\omega||^2$
Subject to:

$$k = 1, 2, \ldots, n_G, \quad d(D_O, D_{G_k}) = [O - G_k]^2 \cdot \omega + b \leq 1$$

$$l = 1, 2, \ldots, n_B, \quad d(D_O, D_{B_l}) = [O - B_l]^2 \cdot \omega + b \geq 2 \tag{11}$$

$$\omega \geq 0$$

This Quadratic Optimization problem may be solved either directly or through the dual problem [30, 15, 10], which proves easier when the number of constraints is much lower than the number of unknowns, i.e., $n_G + n_B << \mathcal{M}$. The use of the bias in the formulation is crucial since it frees us from considering the boundary values and therefore choosing these values to be 1 and 2 does not lose generality.

Iterating The Search

The system may use the new, refined distance function to perform a new search, offering the user a set of results to better suit personal preferences. The user may, on this new set of results, mark preferences as was done for the previous search results. The new 'relevant' and 'irrelevant' results sets may now be used to further refine the distance function.

There is no limit by the system on the number of refinement iterations allowed. However, our experiments showed that very few iterations were needed for any example, before a human user is satisfied with the proposed search results.

4 Experimentation and Results

We tested the algorithm with a database containing over a 1000 objects, given in VRML. First, the database was pre-processed. All objects were sampled, with $10,000$ points representing each object, and then normalized as described in Section 2. Further pre-processing included computing a feature vector of the moments up to a pre-specified order (usually, 4–7 is sufficient), for each object. At that point every VRML object has a small signature (the vector of features) associated with it. Note that in the World-Wide Web setup every signature need be calculated only once, upon creation, and then used as a key for all subsequent searches, without ever having to access the VRML object itself.

At each test search, we chose an object from the database, and asked the system to produce the thirty closest objects. These results are organized by their increasing distance from the desired object. Given these, we marked some as 'relevant', and others as 'irrelevant', according to what seems reasonable to a human viewer. We then re-iterated the search to take these preferences into account, as described above in Section 3.

In Figure 3 we show the Allosaurus, an object to be searched. In Figures 4–7 we show the search results, and their evolution from one iteration to the next, based on the user's markings. First, Figure 4 shows the results of the first search iteration, with unweighted Euclidean distance. These results include four dinosaurs (in addition to the one being searched), and one of them was ranked as 28th. We marked these four dinosaurs as 'relevant' and four other objects as 'irrelevant'. Figure 5 shows the best thirty results of the new search, after the distance function had been weighted according to our preferences. This added three more dinosaurs and they all ranked high. The last two iterations brought in all the dinosaurs of the database (ten, including the original allosaurus) and all of them are ranked at the top. As can be seen from this example, the search results improve with each iteration both in terms of the match of the results to the desired one and also in terms of the order in which the results are given (reflecting their closeness to the desired object). Note, again, that the dinosaurs may differ geometrically. Yet, they are considered close because the user considered them semantically so.

5 Conclusions

We propose a new algorithm for determining the content-based similarity of three-dimensional objects, given in a standard representation, such as VRML. Three main issues are considered. The first is the selection of features that can represent the object in a compact manner, and can thus be considered as a signature for the object, similar to the way keywords representing documents. The second issue is the selection of a distance measure that is adaptable and can enable learning in a way that takes semantics into account. The third issue is he selection of an adaptation rule on the distance measure.

Using the normalized moments as features, a weighted-Euclidean function as the distance measure, and the SVM algorithm to 'train' the distance measure, we introduce a novel learning mechanism, that enables adaptation of the distance measure, based on a user's preferences.

Experiments on a database of more than a 1000 objects exhibit very good results. Not only did more "semantically similar" objects rank close to the query object, from among those found to be "geometrically similar", but also objects which differ geometrically rank high. An order of moments of only 4–7 is sufficient to represent the objects well. Less than five iterations need be run before the system converges to what the user had in mind.

The main aspects of this work can be extended. Other features, such as topological traits, colors and textures can be considered as well. We apply the basic search paradigm in one simple version, namely let the search system present the user with results and allow the user to mark any results the user wishes as either 'relevant' or 'irrelevant'. However, one may think of other variants warranting experimentation: Let the user mark only the results viewed as 'relevant' and consider all other search results proposed by the system as 'irrelevant'; Impose further restriction, requiring not only that all 'relevant' results are close to the searched object, but also that they be close to each other; Impose yet further restrictions, requiring, in addition, that each 'relevant' result be far from each 'irrelevant' result.

References

1. IEEE Computer (1995). special issue on content based image retrieval, 28, 9.
2. Agarwal P.K. , Sharir M. and Toledo S. *Applications of parametric searching in geometric optimization*, Proc. 3rd ACM-SIAM Sympos. Discrete Algorithms, 1992, 72–82.
3. Alt H., Behrends B. and Blömer J. *Approximate matching of polygonal shapes*, Proc. 7th Annu. ACM Sympos. Comput. Geom, 1991, 186–193
4. Alt H. and Godau M. *Measuring the resemblance of polygonal curves*, 8th ACM Symposium on Computational Geometry (1992), 102-109.
5. Alt H. and L.J. Guibas. Resemblance of Geometric Objects. In J.-R. Sack and J. Urrutia, editors *Handbook of Computational Geometry.* North-Holland., Amsterdam, 2000.
6. Alt H., Melhorn K., Wagener H. and Welzl E. *Congruence, Similarity and symmetries of geometric objects*, Discrete Computational Geometry, Vol 3 (1988), 237-256.
7. Arkin E.M., Chew L.P., Huttenlocher D.P., Kedem K. and Mitchell J.S.B. *An efficiently computable metric for comparing polygonal shapes*, IEEE Trans. Pattern Anal. Mach. Intell., 13(3), 1991, 209–216.
8. Arkin E.M., Kedem K., Mitchell J.S.B., Sprinzak J. and Werman M. *Matching points into pairwise-disjoint noise regions: combinatorial bounds and algorithms* ORSA J. Comput., 4(4), 1992, 375–386.
9. Atallah M.J. *A matching problem in the plane*, J. Comput. Systems Sci. 31 (1985), 63-70.
10. Bertsekas D.B. *Non-Linear Programming*, Athena Scientific, Belmont Massachusetts, 1995.
11. Chew L.P., Dor D., Efrat A. and Kedem K. *Geometric pattern matching in d-dimensional space*, Third European Symposium on Algorithms, ESA '95 (1995), 264-279.

12. Chew L.P., Goodrich M.T., Huttenlocher D.P., Kedem K., Kleinberg J.M. and Kravets D. *Geometric pattern matching under Euclidean motion* Comput. Geom. Theory Appl., 7, 1997, 113–124.
13. Chew L.P. and Kedem K. *Improvements on approximate pattern matching problems*, Third Scandinavian Workshop on Algorithm Theory, (1992), 318-325.
14. Cox I.J., Miller M.L., Minka T.P., Papathomas T.V. and Yianilos, P.N. The Bayesian Image Retrieval System, PicHunter: Theory, Implementation, and Psychophysical Experiments. IEEE Transactions on Image Processing, Vol 9, No 1, January 2000.
15. Cortes C. and Vapnik V. *Support Vector Networks*, Machine Learning, Vol. 20 No. 3, September 1995, 273–297.
M.
16. Duda R.M. and Hart P.E. *Pattern Classification and Scene Analysis*, Wiley, 1973.
17. Hagedoorn M. Overmars M. and Veltkamp R.C. *A Robust Affine Invariant Similarity Measure Based on Visibility* 16th European Workshop Comput. Geom. 2000, 112–116.
18. Huttenlocher D.P. and Kedem K. *Computing the minimum Hausdorff distance for point sets under translation*, ISAAC, (1996).
19. Huttenlocher D.P., Kedem K. and Kleinberg J. *On dynamic Voronoi diagrams the minimum Hausdorff distance for point sets under Euclidean motion in the plane*, 8th ACM Symposium on Computational Geometry (1992), 110-119.
20. Huttenlocher D.P., Kedem K. and Sharir M. *The upper envelope of Voronoi surfaces and its applications*, Discrete Computational Geometry, 9, (1993), 267-291.
21. Minka T. P. and Picard R. W. Interactive learning using a "society of models. Pattern Recognition, 30(4), 1997
22. O'Rourke J. and Toussaint G.T. Pattern Recognition. In J. O'Rourke and J. E. Goodman, editors, *Discrete and Computational Geometry*, pages 797–813. CRC Press., New York, 1997.
23. Paquet E. and Rioux M. *A Content Based Serach Engine for VRML Databases*, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1998.
24. Rui Y. and Huang T. S. and Chang S-F. Image Retrieval: Current Techniques, Promising Directions, and Open Issues. Journal of Visual Communication and Image Representation, 10, (1), 1999, pp. 39-62.
25. Rui Y. Huang T. S. and Mehrotta S. Content based image retrieval with relevance feedback in MARS. Proceedings of IEEE ICIP '97.
26. Schwarz J.T. and Sharir M. *Identification of partially obscured objects in two and three dimensions by matching of noisy characteristic curves*, Int. J. Robotics Research, 6(2) (1987), 29–44.
27. Scholkopf B. Support Vector Machines IEEE Intelligent Systems, Vol.13, No. 4, (July-August, 1998), 18–28
28. Sprinzak J. and Werman M. *Exact Point Matching*, Proc. Israeli Symposium on AI, Vision and Pattern Recognition, Elsevier Science Publishers. 1990.
29. Toussaint G.T. Computational Geometry and Computer Vision. In B. Melter, A. Rosenfeld and P. Bhattacharya, editors, *Vision Geometry*, pages 213–224. Amer. Math. Soc., Providence, 1991.
30. Vapnik V. *The Nature of Statistical Learning Theory*, Springer-Verlag, Ney-York, 1995.

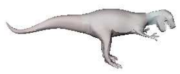


Fig. 3. Allosaurus

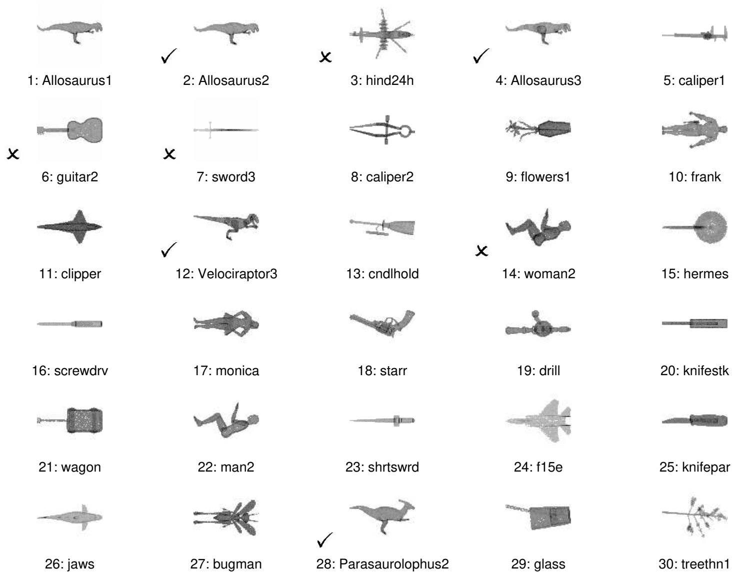


Fig. 4. Allosaurus – Results of the initial search

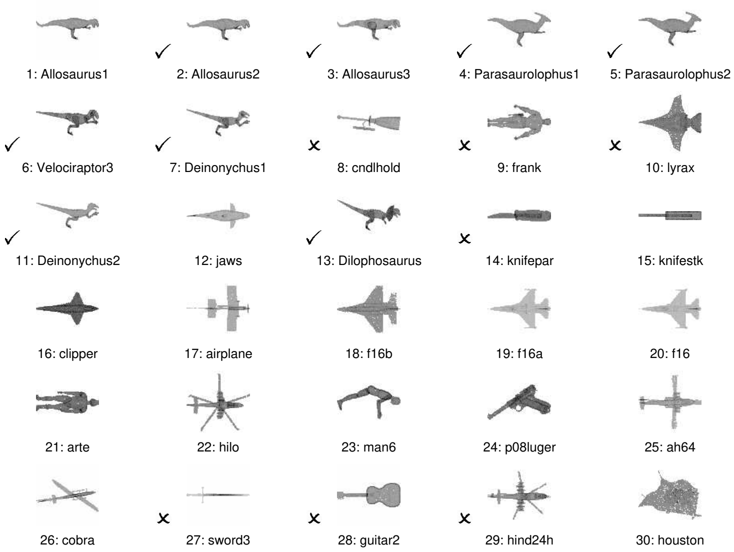


Fig. 5. Allosaurus – Results of $2nd$ search iteration.

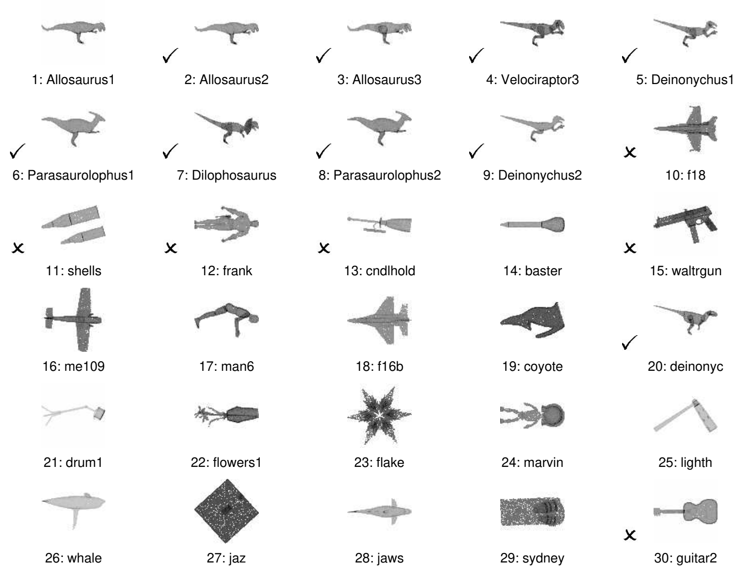


Fig. 6. Allosaurus – Results of $3rd$ search iteration

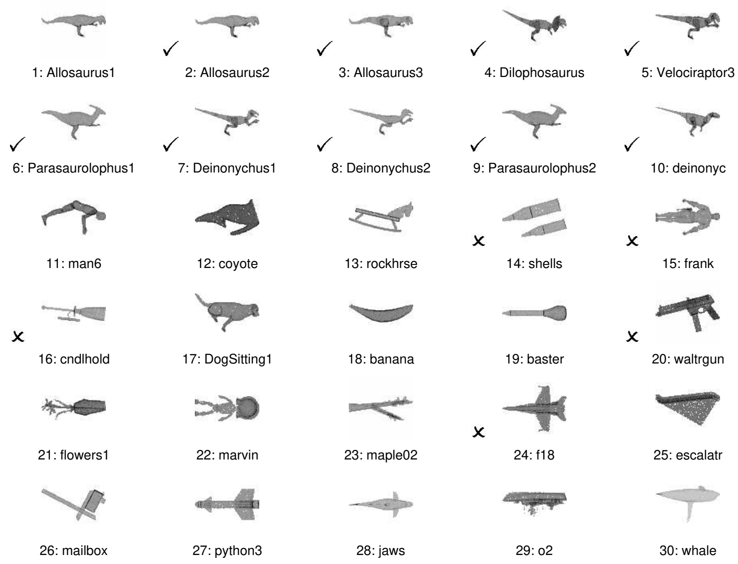


Fig. 7. Allosaurus – Results of the $4th$ (final) search iteration