



Descriptor Based Classification of Shapes in Terms of Style and Function

Welnicka, Katarzyna; Bærentzen, Jakob Andreas; Aanæs, Henrik; Larsen, Rasmus

Publication date:
2011

Document Version
Publisher's PDF, also known as Version of record

[Link back to DTU Orbit](#)

Citation (APA):

Welnicka, K., Bærentzen, J. A., Aanæs, H., & Larsen, R. (2011). Descriptor Based Classification of Shapes in Terms of Style and Function. Kgs. Lyngby, Denmark: Technical University of Denmark (DTU). (IMM-Technical Report-2011; No. 17).

DTU Library

Technical Information Center of Denmark

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Descriptor Based Classification of Shapes in Terms of Style and Function.

Katarzyna Wełnicka, DTU Informatics,
Jakob Andreas Bærentzen, DTU Informatics
Henrik Aanæs, DTU Informatics
Rasmus Larsen, DTU Informatics

October 3, 2011

IMM-Technical Report-2011-17

1 Introduction

It seems that great shifts in how human beings use technology often create a push for changes to the way we divide work between human beings and technology. Chemical film has all but disappeared and almost everybody takes digital photos which they proceed to put online for easy sharing with friends and family. Together with a number of other trends which have contributed to the vast amount of online and locally stored digital photos, this has made automatic recognition of people in images an important research topic - in spite of the fact that recognition is one of the tasks generally left to human beings, since we excel at recognition. We believe that recognition of the style of a 3D object is something that is also likely to be increasingly useful in the foreseeable future. Optical scanning methodologies make the generation of 3D content more feasible than previously, and it is easy to envision digital artists wanting to compile content for a 3D scene or composite object being in need of a method for searching for an object not just of a specific function but also a specific style.

The scope broadens further if we look beyond man made objects. It seems clear that, say, the various limbs of a specific human being have some commonality that separate them from those of another person. Thus, one could argue that an individual represents a style. Style in the context of biological variation is something that we explore in the work presented here. Specifically, we investigate whether we can define a style class for the teeth of a person.

Unfortunately, style is subtle and we do not hope to be able to automatically extract a description of style from 3D objects. Furthermore, we avoid using explicit ways of describing style. Recognizing the style of an object based on some textual or otherwise encoded information might be a feasible approach in some cases such as, for instance, recognizing to which order a given classical

greek column belongs. But, relying on explicit information about a given style would require us either to solve the above problem of automatically extracting style information from shapes or to rely on human beings to encode style - a task that we believe would be both tedious and difficult.

Instead, we rely on examples in the work presented here. This requires that we have example (training) objects for each style. It also requires that we have an orthogonal class of functions, since, as we discuss below, the function of the object (what it is) clearly also has a profound impact on shape. Thus, our work can be summed up as example based classification of digital 3D shapes in both style and function categories.

1.1 Understanding style

Human beings excel at the task of recognizing objects, and, in fact, we are also very good at detecting the style of an object. However, an operational description of style that would allow a computer to detect the style of an object seems to be elusive, at least in general. On the other hand, it is encouraging that humans seem to be able to intuitively recognize the style of an object, and this is why we believe that it is feasible to attack the problem using statistical methods. Our hypothesis is that by using statistics of shape descriptors, we can compute properties of shapes and then train a classifier to discriminate between styles based on these descriptors.

But, there are two significant obstacles that need to be recognized and dealt with before proceeding.

1. Style is not a purely local or global effect. If only we could say to which parts of an object's geometry style has an influence, designing descriptors would be much easier. Unfortunately, that does not seem to be possible. If we think of style as being akin to a gene, the effect on the object's phenotype (metaphorically speaking) could be to the proportions of its parts, whether its edges and corners are sharp or rounded, whether smooth parts are curved or flat, whether it is embellished or not, or whether the surface is smooth or rough. Thus, we cannot simply use local shape descriptors and assume that these capture all the style information.
2. Perhaps a bigger problem is the fact that style is by no means the only thing that determines the shape of an object. Conversely, the function of the object would normally be the biggest contributing factor. By function we understand what the object is recognized as, i.e. the noun one would generally associate with the object, e.g. car, table, chair, tooth. Using a signal processing metaphor, the style is a, sometimes faint, signal superimposed on the stronger function signal.

From the first obstacle, we conclude that we need to use a broad range of descriptors in most cases and that we need to use descriptors which describe both local and global properties of shapes. From the second obstacle, we conclude that we cannot hope, in general, to achieve style discrimination if we do not

| | s_0f_0 | s_1f_0 | s_0f_1 | s_1f_1 | s_0f_2 | s_1f_2 |
|----------|----------|----------|----------|----------|----------|----------|
| s_0f_0 | 0 | 0.3 | 2.2 | 2.4 | 3.5 | 3.7 |
| s_1f_0 | 0.3 | 0 | 2.3 | 2.2 | 3.8 | 3.6 |
| s_0f_1 | 2.2 | 2.3 | 0 | 0.2 | 4.0 | 4.4 |
| s_1f_1 | 2.4 | 2.2 | 0.2 | 0 | 4.3 | 3.8 |
| s_0f_2 | 3.5 | 3.8 | 4.0 | 4.3 | 0 | 0.4 |
| s_1f_2 | 3.7 | 3.6 | 4.4 | 3.8 | 0.4 | 0 |

Figure 1: A matrix of exemplar distances between the shapes. Objects with the same function are much closer than the ones sharing the same style

take function into account. On a very abstract level, our approach is to first gain the ability to compute distances between pairs of objects in "descriptor space". This is a more general approach than always requiring that we have shape descriptors as, say, a multidimensional vector because we can sometimes compute a meaningful distance between two shapes directly (e.g. warping one to the other) but not easily fix a set of coordinates for the two shapes in descriptor space. Given a set of objects where some objects share style and others function, a matrix of such distances might look as illustrated in Figure 1.

In this example style is indeed a fainter signal than function, and if we naively assume that the object with minimal distance to the knobby circle is another knobby object we are disappointed. As we see in Figure 2 it is the plain circle that is closest and then the knobby cube.






| | | | | |
|---|---|---|---|---|
|  |  |  |  |  |
| style | s_0 | s_1 | s_0 | s_1 |
| distance | 0.2 | 2.2 | 2.4 | 3.8 |

Figure 2: Style classification with a naive approach if we forget totally about the function and take into account only style. The shape would be classified to s_0 , as those shapes seem to be closer than shapes having other styles.

Of course, this example is completely contrived and the distances made up. However, it corresponds well with our experience as our results will show, and if we do take function into account, we fare much better. Say, we have a query object of a given function and say we exclude objects of the *same* function, then for a group of objects that all share a common function (different from the function of the query object) the object(s) in that group which has the same style as the query object are likely to be closer to the query object than the objects which have a different style – in addition to having a different function.

At least this is our hypothesis which is illustrated in Figure 3.






| | | | |
|---|---|---|-------|
|  |  |  | f_0 |
| style | s_1 | s_0 | |
| distance | 2.2 | 2.4 | |
| |  |  | f_2 |
| style | s_1 | s_0 | |
| distance | 3.8 | 4.3 | |

Figure 3: Because we grouped shapes to have the same function the difference in the distances is mostly caused by the style effect so the query shape is classified correctly as having the style s_1 .

In summary, our hypothetical model is that

$$\text{Shape} = \text{Function} + \text{Style} + \text{Noise} \quad (1)$$

where addition should be construed as a more abstract composition operator. In the formula above, we readily admit that function probably has the greater influence. Taking that into account, we believe that we can approach the daunting task of creating a statistical model for style.

1.2 Related Work

Style and function separation in the context of man made three dimensional shapes was recently mentioned by Xu et al. [19], where the style of an object is defined by the proportions (anisotropic scaling) of its parts. It seems to be a very intuitive and reasonable approach but this does not exhaust the subject.

In many shape processing articles, even if the problem of style is not addressed in an explicit way there are situations where the space of given shapes is broken down into two different independent classification systems. In the deformation transfer [14] different kinds of animals can take similar poses, in which case it is quite easy to localize them, as the type of animal is described by an intrinsic metric of the shape surface, and the pose is its embedding in three dimensional space. The idea of geometric texture [1] fits within this framework as it aims to separate overall shape from its geometric details. Application of example based priors for surface reconstruction [4, 13] can also be seen as imposing style to the object.

In the image processing field, Hertzmann et al. [7] presented a method that, when given three images: one with style A and function 1, one with style B and function 1, and another one with style A and function 2, creates an image with style B and function 2. The same concept has also been explored by this group in the field of curve styles [8]. Other related problems can be found when

dealing with images of fonts, separating lighting conditions from the scene and distinguishing between a spoken language and the accent. All those three cases were examined through bilinear models by Tenenbaum et al. [17].

Tenenbaum’s framework requires establishing one to one correspondences both for the style and for the function - for example fonts are compared through corresponding pixels of their bitmaps. In general, for different types of shapes obtaining such correspondences can be difficult. Similar correspondences need to be established across the styles for Hertzman’s work. Our approach does not require any correspondence finding, which is usually costly and sometimes outright impossible, like in the problem of registering a table to a chair [5].

In general, if the feature space is available, many well established statistical methods can be used for metric learning. For example Linear Discriminant Analysis [9], which modifies the feature space such that, for a given training set containing objects from different classes, the inter class variance is maximized and intra class variance minimized. A similar approach was also used by [18] which makes it possible to define similarity and dissimilarity relationships between selected pairs of objects. In a similar manner, Giorgi et al. [5] customize a way of combining a set of distances between shapes so that user defined similarity is captured. In this work the metric is modified in order to reflect the user defined constraints of nearby or far away shapes. The final metric is taken as a maximum distance from distances given by all of the metrics, however, the particular metrics are scaled according to a similarity feedback provided by the user.

For the task of finding the replacement of an object from the one of the most similar style, a similarity measure needs to be established. As an input to the algorithm we have many hypothetical distance measures, but we do not know which one is the most suitable - having such information is indeed equivalent to solving the initial problem. We assess the relevance of a specific shape descriptor indirectly as a consistency requirement: dissimilarity or similarity between the styles should be reflected in a similar way for different functions. A similar methodology, based on indirect consistency, can be found in [20]. The aim is to remove incorrect mappings between different views of a scene. The quality of these mappings is assessed by analyzing the mapping loops which they form. An inconsistent loop indicates that at least one of the mappings that belong to it is wrong, while a consistent loop means that all mappings are likely correct. Having evaluated the correctness of many loops the bad mappings are spotted through loopy belief propagation. A similar approach is performed by [11] in the space of shape maps.

1.3 Contributions

We introduce a framework for dealing with style by validating how given descriptors relate to style within the context of a given dataset (section 3). As far as we know there was no general framework for such problem, especially when no parameter space is available. We also show how we can deal with style classification, even if we cannot find the descriptors which are purely responsible

for the style, by using additional function information (section 2).

We propose a statistical model for style which takes both function and style into account. Using this model, we can sort objects according to both style and function provided we have example objects for each style and each function. Operationally, this allows us to achieve some tasks. For instance, automatic sorting of chess pieces according to the type of chess and according to the set of pieces to which it belongs. We also solve the task of finding a tooth model which can be used for the design of a prosthesis to replace a missing tooth – even if we do not have a very similar tooth model in the patient’s mouth (section 5.3).

The general methods are introduced in sections 2 and 3. In section 4 we present the descriptors used for our experiments and in section 5 we show the experiments performed on tooth and chess datasets and analyze the obtained results.

2 Style and Function Classification

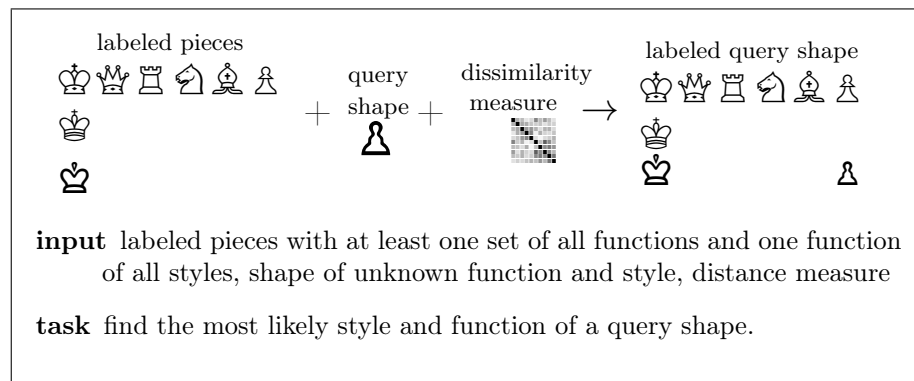


Figure 4: Diagram shows the task of classifying a shape according to style and function. Note that in this framework we need example shapes which serve as definitions of styles and functions and a way to measure the distances between the shapes.

In this section we will show how information about style and function hidden in the same metric can be decoupled. The problem is illustrated in figure 4. We assume here that we have a dissimilarity measure that was produced by some kind of shape descriptor and contains both functional and stylistic information. One example of such a measure was given in the introduction section (see figure 1). Based on this information we want to be able to detect the most likely style and the most likely function of a query shape. Because we do not have explicit descriptions of the style or of the function, we assume they are given by the examples through the training set T .

2.1 Likelihoods Computation

In this section we introduce the likelihood of some shape being of given style and function, which is based on the distances of the unknown shape to shapes from a training dataset. We want to formulate it in a way that the function effect is eliminated when assessing the style and the style effect is eliminated when assessing the function.

We denote the shape of style s and function f as S_f^s and the set of shapes of a function f and style different than s as S_f^{-s} .

Our main observation here is that if we have in the training set shapes which share the same function but have different style, then it is possible that we may factor the function out as it was shown in figure 3.

For example if the distance to a S_a^1 is smaller than a distance to a S_a^2 , we may say that the unknown shape is more likely to be of a style 1 than to be something else.






| | | | | |
|---|---|---|---|---|
|  |  |  |  |  |
| shape | S_c^0 | S_c^1 | S_c^2 | S_c^3 |
| j | 0 | 1 | 2 | 3 |
| $d(S_s^3, S_c^j)$ | 2.4 | 2.7 | 2.1 | 2.7 |
| $l_1^{s=j}(S_s^3)$ | -0.2 | -0.9 | 0.8 | -0.9 |

Figure 5: Likelihoods calculated within a training set of circles for a query object S_s^3 . The biggest value is reached when the distance is the smallest

The partial likelihood $l_i^{s=j}(x : k)$ of unknown shape x to be of style j , when we have two example shapes of the same function i of which one S_i^j is of a style j and another one S_i^k is of a style k other than j , is equal to:

$$l_i^{s=j}(x : k) = d(x, S_i^k) - d(x, S_i^j).$$

In our training dataset T , for a given function i , we may have more than just one shape not being of a function j so we take the mean plus the minimum of all of the partial likelihoods:

$$l_i^{s=j}(x) = \text{mean}_{k \in \{-j\}: S_k^i \in T} l_i^{s=j}(x : k) + \min_{k \in \{-j\}: S_k^i \in T} l_i^{s=j}(x : k).$$

Note that minimum is equal to the distance to the closest of the known shapes from function i and style other than j , minus the distance to S_j^i . If the style j is the closest of the shapes from that style, then the minimum will be positive, otherwise it will be negative. The mean value stabilizes the results by taking into account distance measures of all of the shapes of this function but different style. So the first term is an average distance of the unknown shape to a shape from the training dataset having a function i and not being the style j minus the distance to S_j^i .

The use of differences of the distances instead of the direct use of distances is caused by the additive model expressed in the equation 1. Because the distance of two shapes is caused both by functional and stylistic differences, we want to remove the impact of the function by using the difference of distances to shapes having the same function but different styles.

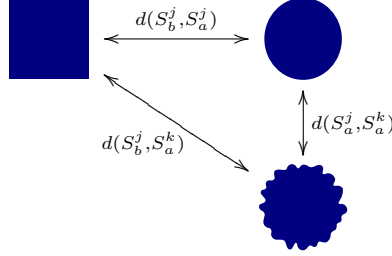


Figure 6: Graphical representation of the diagonal property. The distance on the diagonal should be greater than the vertical and horizontal distances. In this way even when the inter-style (vertical) distances are much smaller than the inter-function (horizontal) distances we are still able to capture the stylistic difference, when we know function labels.

Such an approach is based on the assumption that the distance to a shape having different both style and function properties should be greater than the distance to a shape being different just with respect to style (or function). Graphically speaking, if we put two shapes of the same style in the same row and two shapes of the same function in the same column of a table, it means that the distances computed across the diagonals would always be greater than the ones between shapes displaced only vertically or horizontally (see fig. 6).

Definition 1. *The metric d has a diagonal property on the set of shapes $\mathbf{S} = \{S_f^s : f \in F, s \in S\}$, if for any $j, k \in S$ and $a, b \in F$ we have*

$$d(S_a^j, S_b^k) \geq d(S_a^j, S_b^j)$$

$$d(S_a^j, S_b^k) \geq d(S_a^j, S_a^k)$$

Lemma 1. *If d on \mathbf{S} has a diagonal property, then for $S_h^j \in \mathbf{S}$ for all $k \in -j$ we have $l_i^{s=j}(S_h^j) \geq l_i^{s=k}(S_h^j)$*

Proof. Let's denote the mean part of the $l_i^{s=j}(x)$ as $l_i^{s=j}(x)_{\text{mean}}$ and the minimum part as the $l_i^{s=j}(x)_{\text{min}}$. So we have $l_i^{s=j}(x) = l_i^{s=j}(x)_{\text{mean}} + l_i^{s=j}(x)_{\text{min}}$.

For the mean part:

$$\begin{aligned}
l_i^{s=j}(x)_{\text{mean}} &= \frac{\sum_{m \in -\mathbf{j}} d(x, S_i^m)}{|i|_T - 1} - d(x, S_i^j) \\
&= \frac{\sum_{m \in -\mathbf{k}} d(x, S_i^m) + d(x, S_i^k) - d(x, S_i^j)}{|i|_T - 1} - d(x, S_i^j) \\
&= l_i^{s=k}(x)_{\text{mean}} + \frac{|i|_T}{|i|_T - 1} (d(x, S_i^k) - d(x, S_i^j))
\end{aligned}$$

Where $|i|_T$ denotes the number of shapes in the training dataset that have a function i . By putting $x = S_h^j$, and having $d(S_h^j, S_i^k) - d(S_h^j, S_i^j) \geq 0$ from the diagonal property, we get:

$$l_i^{s=j}(S_h^j)_{\text{mean}} \geq l_i^{s=k}(S_h^j)_{\text{mean}} \quad (2)$$

$$\begin{aligned}
l_i^{s=j}(x)_{\min} &= \min_{m \in -\mathbf{j}} d(x, S_i^m) - d(x, S_i^j) \\
&= \min \left(d(x, S_i^k), \min_{m \in -\mathbf{j} \cap -\mathbf{k}} d(x, S_i^m) \right) - d(x, S_i^j)
\end{aligned}$$

analogically

$$l_i^{s=k}(x)_{\min} = \min \left(d(x, S_i^j), \min_{m \in -\mathbf{j} \cap -\mathbf{k}} d(x, S_i^m) \right) - d(x, S_i^k)$$

Applying a diagonal property $d(S_h^j, S_i^k) \geq d(S_h^j, S_i^j)$ and using the fact that $a \geq b$ implies $\min(a, c) \geq \min(b, c)$ for any c we have:

$$l_i^{s=j}(S_h^j)_{\min} \geq l_i^{s=k}(S_h^j)_{\min} \quad (3)$$

And from inequalities 3 and 2 we have also: $l_i^{s=j}(S_h^j)_{\min} \geq l_i^{s=k}(S_h^j)_{\min}$ \square

In order to gather information from all of the training functions, we take the mean value plus the maximum of all the styles for which in the training set there is a style j and some shapes not being of style j .

$$l^{s=j}(x) = \max_{i: S_j^i, S_{-j}^i \in T} l_i^{s=j}(x) + \text{mean}_{i: S_j^i, S_{-j}^i \in T} l_i^{s=j}(x).$$

Here, by taking the maximum, we are favoring the function for which the style j is most likely. The mean is again added to get the distance information from all known styles.

There might be cases when we do not have enough information in the training set for establishing likelihoods. This happens when there is no set which has a training representative for the style j and for some shape which is not of a function $-j$. In such a case we set the likelihood to zero.

The whole problem might be inverted and the likelihood computation of "x being the function i " is done in an analogous manner. Then for a given x the cost of assigning to it style j and function i is equal to: $l_{f=i}^{s=j}(x) = l_{f=i}(x) + l^{s=j}(x)$.

2.2 Exploiting Uniqueness

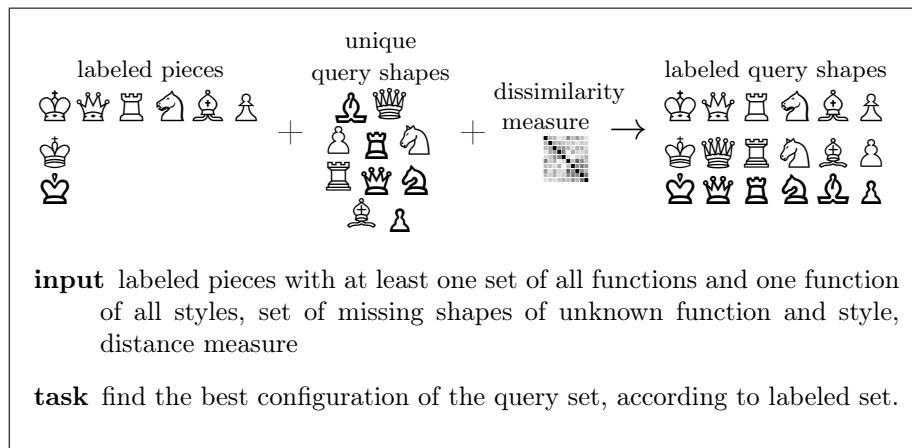


Figure 7: Diagram shows the task of sorting the query shapes. Each shape can be assigned only to one label. This constraint makes the task much easier than when treating all query shapes separately.

If the dissimilarity measure on the given dataset has the diagonal property then the likelihood will never fail to show the style and function of an unknown object. In many applications this is not the case: for at least some percentage of style and function pairs the property will be violated. So the correct style or the correct function will not get the largest likelihood score. If we however address the problem not in separate queries but assume that we need to assess the style and function for a bunch of shapes (figure 7) of which we know that there are no repeating shapes in our query then our results can be much improved.

This approach follows a general concept that a uniqueness constraint makes the problem much easier. We use the likelihoods as negative costs and solve the minimum linear assignment problem [2] for the unknown labels and loose shapes.

2.2.1 Multiple Step Assignment

An assignment problem with the costs defined above does not make use of the information about all of the distances between the shapes. Only information about the distances to the training shapes is taken into account. It might be an advantage when we do not want to compute the distances between all of the shapes, but if we already have computed all of the distances we may want to include them to make our algorithm better.

If we are able to locate the shapes for which we can expect that the initial matching went correctly we can add those into the training dataset with the labels obtained by the initial assignment. We do not have an oracle which

tells which pieces were assigned correctly and which were not. If we had such an oracle it would also automatically solve our assignment problem. However with some additional measures we can assume that there are pieces which were labeled more reliably than the other ones.

In order to estimate the labeling reliability, we calculate the diagonal cost of the assignment which we define as the average sum of similarities between all the shapes having the same style or function labels.

Definition 2. Let $A(S)_i^j$ denotes the shape S which has a function label i and style label j obtained by an assignment A . Then a diagonal cost of such assignment is:

$$dc(A) = \frac{\sum_{A(S)_k^m} \left(\sum_{A(S)_k^{j \in -m}} d(A(S)_k^m, A(S)_k^j) + \sum_{A(S)_{i \in -k}^m} d(A(S)_k^m, A(S)_i^m) \right)}{\sum_{A(S)_k^m} \left(\sum_{A(S)_k^{j \in -m}} 1 + \sum_{A(S)_{i \in -k}^m} 1 \right)}$$

For a hypothetical unknown shape we might add it for a moment to the training set with the labels that we got as the solutions of the initial problem. Then as the labeling reliability we could calculate what is the diagonal cost when assignment is solved with the use of this piece. However, we discovered that instead of calculating diagonal costs directly it is better do the inverse assignment, which is performed by swapping the unknown data with the known, solving the inverse problem and then calculating the diagonal cost.

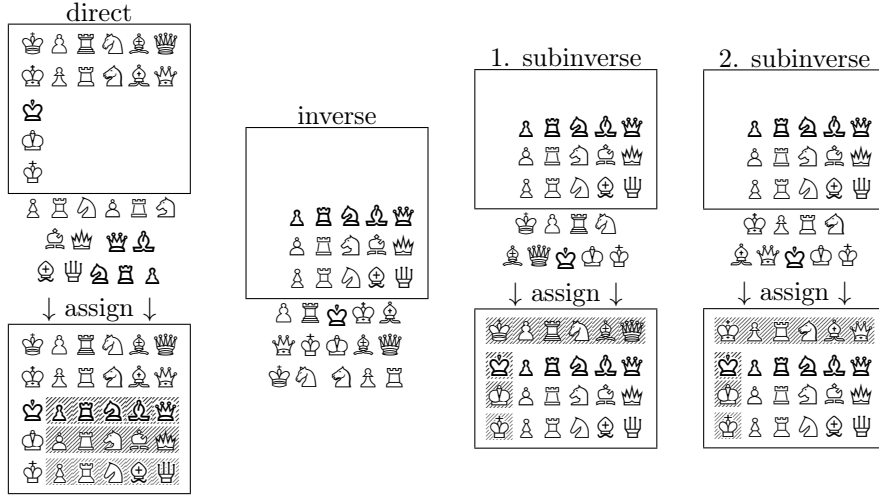


Figure 8: The inverse problem is made from a solution of a direct problem by exchanging the known information with the unknown. Because the solution of the inverse problem is not always unique, we instead solve the subproblems and then calculate their diagonal cost, which then is used to calculate the final inverse diagonal cost.

It might happen, for example if we know all the shapes of two styles, which become fully unknown in the inverse problem (fig. 8), that we are not able

to find the inverse solution. In such case we take small subsets of data, by excluding from the problem all but one of the shapes of the style (function), which for the inverse problem have the style (function) cost undetermined. We solve inverse subproblems and take the sum of the diagonal costs for all of the given subtasks, divided by the number of all elements sharing the style and the function. The smaller the inverse diagonal cost, the more reliable is the hypothetical assignment of the unsorted shape to its label.

We also added other sanity checks of the shapes and consider the following properties:

swapping minimum : swap the hypothetical shape label with all other shape labels in an initial assignment. If the diagonal cost of some of the swapped assignments is smaller than the initial one, this piece is unreliably assigned.

perturbation persistence : solve two assignment problems as initial but instead of original cost use

$$l_{f=10i}^{s=j}(x) = 10 * l_{f=i}(x) + l^{s=j}(x)$$

$$l_{f=i}^{s=10j}(x) = l_{f=i}(x) + 10 * l^{s=j}(x).$$

Then if the chess piece is not assigned to the same label as the initial problem assignment of this piece is unreliable.

In order to minimize bad choices we always take the piece having minimum inverse diagonal cost and which is reliable according to the above reliability criteria. We add it to the training dataset and repeat the assignment and addition of the most reliable pieces until there is no reliable piece to be added. Then we use the assignment from the last step as the final assignment.

3 Consistency Learning

In this chapter we want to solve a problem which can be seen as a reverse to the one solved in the previous section. We have an incomplete set of shapes of some style and we want to find the missing one. Our task is to search in the database of available shapes for the one which is most likely to be of a given style (figure 10).

We assume here that the function is known or can be easily detected. In many contexts function can be given explicitly: for example the type of a tooth is usually associated with its position in the mouth. Also, this is a sound approach when function is much more distinctive and we can determine it easily by using standard shape retrieval methods, as for example it is not so difficult to distinguish between a table and a chair.

We treat the ‘style space’ in a continuous way. We expect some styles to be very close to other styles, such that a shape of a given function can be replaced with a similar style quite well. This approach is especially suitable in a domain of shapes with some kind of biological variation.

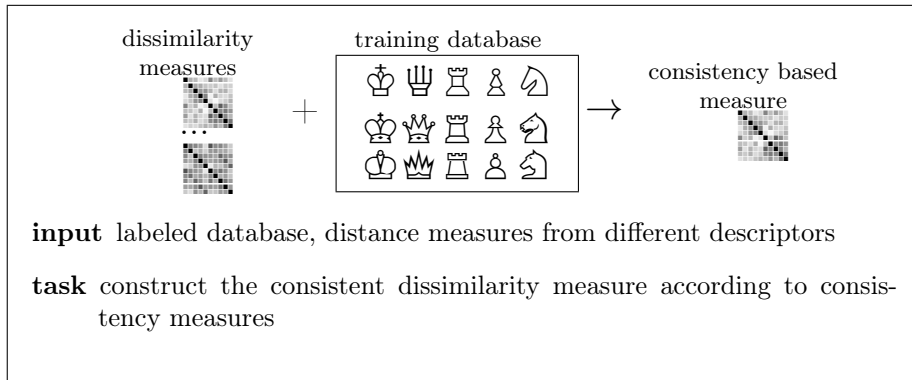


Figure 9: The training phase of the replacement finding task. The consistency dissimilarity measure is constructed based on consistency performance of different descriptors within the context of the training database.

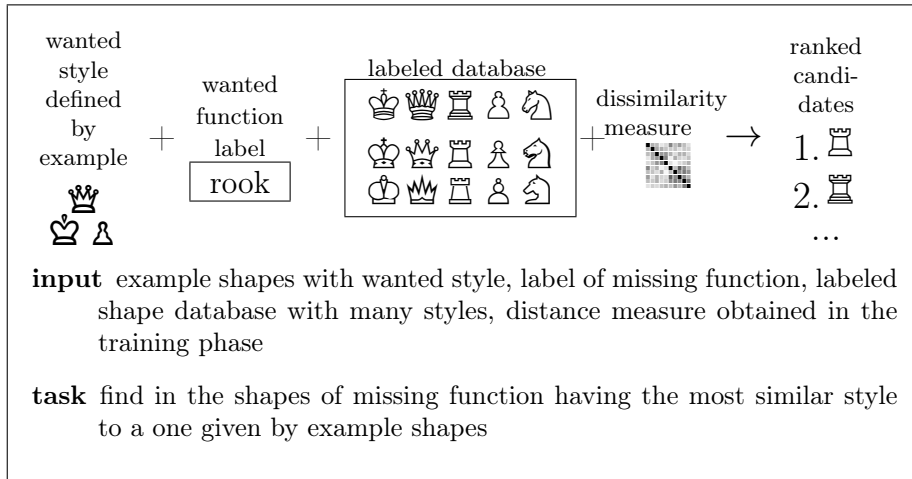


Figure 10: The second step of the replacement finding task. User comes with examples of a style and asks for a shape of that style and a function that was not present in the example shapes. The candidates are selected from the database and ranked according to estimated similarity to the queried style.

In the previous section we took a good dissimilarity measure for granted. In general, we might not know it advance. Instead, we have many proposals of dissimilarity measures $d_i(\cdot)$ which can be obtained through different kinds of shape descriptors D_i .

The task is to choose such a measure d_i or a combination of measures with

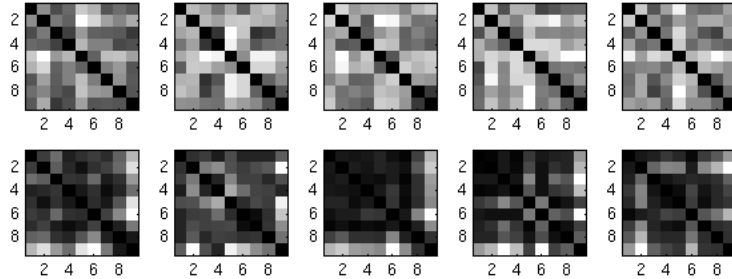


Figure 11: Distances between shapes of the same function and different styles. 5 functions are displayed (columns) and two different descriptors (rows). The top descriptor has consistency 19.35 and the bottom one has consistency 69.27. Note that the smaller the consistency measure, the more similar should the plots of different functions be. Data was taken from calculation for three dimensional descriptors of the chess pieces.

which we can distinguish between different styles. This task is related to metric learning approaches like LDA [9]. But the aim of LDA would be to have objects of the same style close and those of different style far away. We also require that the dissimilarity measures should be consistent across different functions.

We can illustrate the problem with the tooth shapes. Suppose a patient has one tooth destroyed. In order to be able to reproduce its shape, we want to find in a database a tooth which is mostly similar to the existing tooth he has. We have a molar missing but because a premolar is still in the patient’s mouth, we wish to search in our database for a mouth which has the most similar premolar to the patient’s. From that mouth we take a molar as a template for our new tooth. This approach assumes that similarity for premolars induces a similarity between molars.

The tooth replacement example shows that the consistency requirement is necessary as it aids in many concrete tasks – like searching for the best replacement for some missing data. Here we do not know directly what ‘close’ means, as we have many metrics but don’t know which one is a correct. Usually a correct metric combination in such a case can be found by giving example pairs of shapes which are similar and pairs which are dissimilar [5]. In our case we do not have such information. Instead we can impose the metric consistency requirement: the distances between shapes having different styles and function A should be close to the distances of the shapes of the same styles and function B .

3.1 Consistency

In this section we introduce consistency factors which measure the similarities of distances between styles across different functions. We also create a final dissim-

ilarity measure from different descriptors with the weights assigned according to the descriptors' consistency.

Assume we have a set of training shapes S_i^j , where $i = 1..n_i$ indicates the function and $j = 1..n_j$ the style. We also have k_n potential dissimilarity measures $d_k(\cdot)$

Let us take all distances $d_k(S_{j_a=1..n_j}^{i_1}, S_{j_b=1..n_j, j_b \neq j_a}^{i_1})$ between different shapes of the function i_1 . In order to be comparable those distances need to be normalized, which we do by dividing them by the median of obtained distances. This results in a $\binom{n_j}{2}$ dimensional vector of k-distances between all possible pairs of shapes with different styles and the function i_1 , which we will denote $v(d_k, f_{i_1})$.

For each pair $i_1 \neq i_2$ of two different functions we can establish the **consistency score** $cs_{d_k}^{i_1, i_2}$ with respect to a distance k and function i_1 and i_2 as the norm of difference of distance vectors:

$$cs_{d_k}^{i_1, i_2} = \sqrt{\sum_{l=1..(\binom{n_j}{2})} (v(d_k, f_{i_1})_l - v(d_k, f_{i_2})_l)^2} \quad (4)$$

In order to calculate the total consistency factor (TCF_k) for a dissimilarity measure k , we take the sum of the differences for all function pairs. Note that the smaller TCF_k is, the more consistent d_k is with respect to style.

We construct the final measure by summing the dissimilarities obtained through different shape descriptors with weights that promote consistency.

$$D_f(x, y) = \sum_k e^{(-2 \frac{TCF_k}{\text{mean}(TCF)})} \frac{d_k(x, y)}{\sigma_{d_k}} \quad (5)$$

where σ_{d_k} is the median distance from distances $d_k(\cdot)$ between all training shapes.

3.2 Query Based on Consistency Learning

In the training phase, given the database labeled with styles and functions, we compute the consistency measures from different descriptors and construct a final metric according to equation 5.

In the user phase, when asking for a specific function q we also provide samples of the style with shapes labeled by functions: $Q_1, ..Q_n$. For each such shape, we measure the distances between Q_i and all the shapes from the database sharing this function. The distances reflect the similarity of the queried style to known styles and they might be slightly different if the consistency of the final metric is not perfect. The measure of similarity of the queried style with styles in the database is obtained by summing D_f for different functions.

$$D^a(s = j, Q) = \sum_i D_f(S_i^j, Q_i)$$

We take from the database the closest style as the one with the smallest $D(s = j, Q)$, and take the shape S_q^j as a replacement of the unknown shape Q_q .

We can also go a step further and not take all D_f with the same weights. Having the final dissimilarity measure we compute consistency scores $cs_{D_f}^{i_1, i_2}$ between different functions. These consistency measures can be used in order to asses what pairs of functions are better correlated. For example two neighbor upper molars can be more correlated than a molar and incisor. So if a molar is missing and we have the neighbor molar and incisor, we should give higher weight for query of the closest mouth with respect to a molar than with respect to an incisor. We could either use only the distances with respect to a function most correlated with the query function, or use the weights according to the consistency scores:

$$D^w(s = j, Q) = \sum_i e^{\left(-2 \frac{cs^{q,i}(D_f)}{mean_i(cs^{q,i}(D_f))}\right)} D_f(S_i^j, Q_i)$$

Both in the training and user phase we need a dataset, with labeled functions and styles. This can be the same dataset. The queried style needs to be excluded from the training phase as we do not know what kind of data will be provided by the user.

4 Computing Distances Between Shapes

The methods presented in the previous sections are quite general and are independent of the descriptors we use. On the other hand the performance of the method relies on their choice, as if we use descriptors that are totally unable to capture similarities then the results we obtain will also be of poor quality. For example using only global descriptors for a dataset with the style being mostly expressed in local details would not be a good idea.

Note that in this paper we do not require for the dissimilarity measure produced by our descriptors to have properties of a metric space.

4.1 Curves Comparison with Dynamic Time Warping

In this section we explain how distances between oriented curves can be obtained through Dynamic Time Warping. This is a good example that there are cases when we have a way of establishing the dissimilarities between shapes without a general feature space. On the other hand there is always a way to go from a feature space with coordinates to a dissimilarity measure, by using one of many available vector norms. So the use of similarity is more general than the use of the feature space (see also Giorgi et al. [5]).

The curve \mathbf{X} is represented as a polygonal chain $\mathbf{x}_{i=0..n}$. The standard dynamic time warping problem for two curves \mathbf{A} and \mathbf{B} is to find a sequence of correspondences between their vertices $\mathbf{a}_{i=0..n}$, $\mathbf{b}_{i=0..m}$, denoted as $\mathbf{C} = \mathbf{c}_{i_k j_k}$ where $\mathbf{i}_k \in \{0..n\}$ and $\mathbf{j}_k \in \{0..m\}$ and satisfying the conditions of:

monotonicity if $\mathbf{c}_{i_k j_k}, \mathbf{c}_{i_l j_l} \in \mathbf{C}$ and $\mathbf{i}_k \leq \mathbf{i}_l$ then $\mathbf{j}_k \leq \mathbf{j}_l$

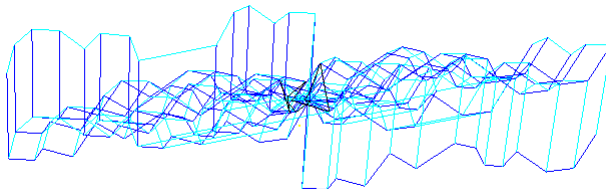


Figure 12: A manifold $V(A, B)$ when comparing two similar curves. Geodesic between the \mathbf{v}_{00} and \mathbf{v}_{nm} is marked with black. Although the embedding of this manifold is complicated it has a simple square topology

continuity for incident correspondences $\mathbf{c}_{i_k j_k}$ and $\mathbf{c}_{i_{k+1} j_{k+1}}$ we have:

$$\mathbf{i}_{k+1} - \mathbf{i}_k \leq \mathbf{1} \text{ and } \mathbf{j}_{k+1} - \mathbf{j}_k \leq \mathbf{1}$$

Classical DTW searches for the correspondence with minimal sum of lengths of vectors $\mathbf{v}_{ij} = \mathbf{a}_i - \mathbf{b}_j$ whose endpoints are defined as vertices indicated by the correspondence.

$$\mathbf{d}_{\text{DTW}}(\mathbf{A}, \mathbf{B}) = \min_{\mathbf{C}(\mathbf{A}, \mathbf{B})} \sum_{\mathbf{c}_{i_k j_k} \in \mathbf{C}(\mathbf{A}, \mathbf{B})} \|\mathbf{v}_{i_k j_k}\|$$

The translation invariant version of Dynamic Time Warping, instead of minimizing the sum of lengths of \mathbf{v}_{ij} , minimizes the sum of lengths of the difference of vectors \mathbf{v}_{ij} for two incident correspondences:

$$\mathbf{d}_{\text{TIW}}(\mathbf{A}, \mathbf{B}) = \min_{\mathbf{C}(\mathbf{A}, \mathbf{B})} \sum_{\mathbf{c}_{i_k j_k} \in \mathbf{C}(\mathbf{A}, \mathbf{B}), k > 0} \|\mathbf{v}_{i_k j_k} - \mathbf{v}_{i_{k-1} j_{k-1}}\|$$

The discrete version of DTW depends heavily on how the vertices are positioned on the curve, since for a given vertex the corresponding point must be chosen from the vertices of the second curve.

We use the method of Efrat et al. [3] and transform the translation invariant DTW into a continuous setting. In such a case we want to represent as \mathbf{a}_i any point on a polygonal chain \mathbf{A} and for this purpose we extended linearly the index $\mathbf{i} \in \{0 \dots \mathbf{n}\}$ to a domain of real numbers $0 \leq \mathbf{i} \leq \mathbf{n}$. This is done using the interpolation of values known at vertices: $\mathbf{a}_i = (\lambda - 1)\mathbf{a}_{\lfloor i \rfloor} + \lambda\mathbf{a}_{\lceil i \rceil}$ where $\lambda = \frac{i - \lfloor i \rfloor}{\lceil i \rceil - \lfloor i \rfloor}$. As a result vectors $\mathbf{v}_{ij} = \mathbf{a}_i - \mathbf{b}_j$ are also extended to a two dimensional surface defined as the combinatorial manifold $\mathbf{V}(\mathbf{A}, \mathbf{B}) = \mathbf{A} \oplus -\mathbf{B}$. After this modification the translation invariant problem can be defined as finding the shortest monotonous path $\mathbf{P}(\mathbf{A}, \mathbf{B})$ on this manifold which connects the endpoints \mathbf{v}_{00} and \mathbf{v}_{nm} . The minimized function \mathbf{d}_{CTIW} is the length of this path and this value is used to establish the dissimilarity between the curves.

4.2 Three Dimensional Descriptors

Recently there has been a lot of work within the content based shape retrieval field and a huge amount of different shape descriptors exist [15] and many new

methods are proposed each year. The performance of such methods is measured through the ability to retrieve objects of the same class as the query shape, across some given benchmark of shapes (SHREC retrieval contest). As mentioned by Godil et al. [6] in the field of general shape retrieval, usually hybrid methods, which combine many shape descriptors, perform better as they can capture many local and global characteristics.

Our main contribution was not to introduce any kind of a new descriptor which will perform well within our style-function problem. Instead we propose a method to assess the usability of existing descriptors, to combine and use them so that our style function discrimination tasks can be achieved.

Besides our general approach, we briefly present the descriptors which we have used as input for our methods. We have chosen to use local shape descriptors which rely on neighborhood at some distance from a given position. This way, by changing the neighborhood size both local and global features can be captured. We used three types of such descriptors:

curvatures : minimum and maximum curvature obtained by fitting primitives through points sampled from the neighborhood area [16] (2x4 descriptors),

covariance : eigenvalues of the covariance matrix of points sampled from the neighborhood area (3x4 descriptors),

slippage coefficient which are 6 eigenvalues of the slippage covariance matrix [10] of points sampled within the neighborhood area; we have also included six values being a translational contribution to eigenvectors (12x3 descriptors).

We uniformly sampled the surface of the shapes and computed local descriptors out of those samples. As neighborhood size we have taken 0.01, 0.04, 0.16 and 0.64 of the radius of a bounding sphere of a tooth. For slippage we used 0.01, 0.04 and 0.16.

For each shape we gathered local information coming from any descriptor into a soft histogram, which means that the values are convolved with a Gaussian kernel of a fixed width before being discretized in a histogram. A smooth histogram has the advantage that it induces the smoothness property of a descriptor: if a shape varies smoothly under continuous deformations, the descriptor will also vary continuously [12].

In order to reduce sampling bias we took 2 samplings of 1000 points each, and computed a soft histogram for each of them. Histograms from the two independent samplings were compared. The mean across all training shapes of their difference was taken in order to estimate the measure error coming from the different samplings. Then the mean of the 2 histograms is taken. However in order to compare two histograms for shapes S_i and S_j the distance between two bins is reduced by the previously computed measure error. Then the sum of those values is taken across all bins as our distance $d_k(S_i, S_j)$.

Note that besides those descriptors any descriptors which can compute pairwise distances between shapes can be used here.

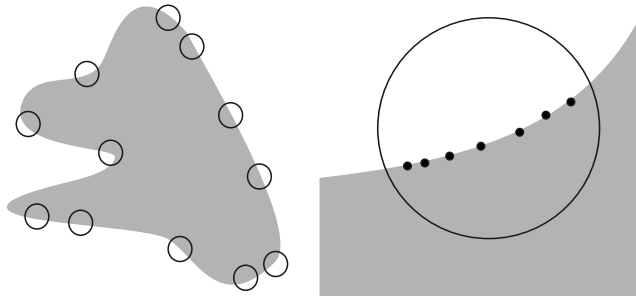


Figure 13: Descriptors we use require two levels of sampling. The first level is to take points uniformly sampled on the whole surface (left). Local descriptors computed at those points are collected in the form of a histogram. The second sampling occurs at the level of computing local shape descriptors, when new sample points are taken within a required distance from the base point (right), those points are used to establish measures of that surface, such as curvature which is estimated by fitting primitives into the sampled points.

5 Results and Applications

5.1 Chess Piece Classification

In order to obtain the first dataset we scanned 9 different chess sets. The type of the chess set is the style and the function is a chess type. In a standard chess set we have 6 different functions, however we excluded the knight as this piece was not rotationally symmetric. This resulted in 5 different functions. Since there is clear rotational symmetry in the chess pieces, their three dimensional representation was reduced to the space of plane curves by taking the outline curve obtained through rotating the chess piece by its rotational symmetry axis (figure 2). In order to eliminate the scaling factor each piece was rescaled to the same height. Continuous Translation Invariant Dynamic Time Warping was used in order to establish a similarity metric $d(.,.)$ between the curves (figure 14).

In the first experiment we have taken one style and one function as training shapes (refer to section 2.2). For the rest of the chess pieces the assignment problem was solved. Table 3 contains the result of such assignment where for each style and function we have three values: the first indicates how many test shapes had wrong label, the second how many shapes had wrong function label and third how many had wrong style label. We have also calculated the average performance for all styles and functions. The results depend a lot on the type of the set and function imposed as an example shape. Some of the sets contain a lot of function information but some others do not. The sets 024 and 008 perform the worst. Also the results for the rooks are always worse than for other functions. Note that if we provide a given style as the training set it is used as a definition for the functions and if we give some function it is used

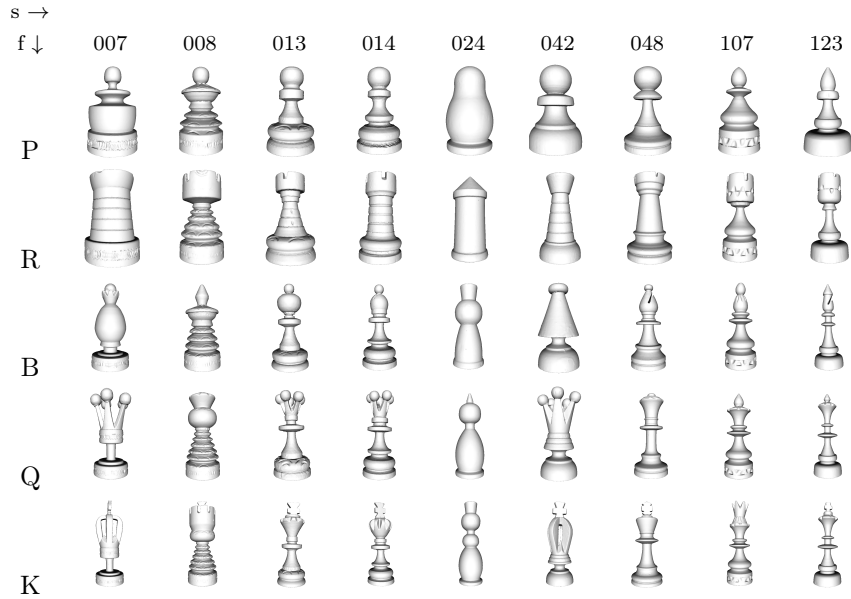


Table 1: The chess piece dataset. Our dataset has 45 chess pieces, which are the scans taken from 9 existing chess sets. The function is the type of the chess piece (pawn, rook, bishop, king, queen) and the style is the set the chess piece belongs to.

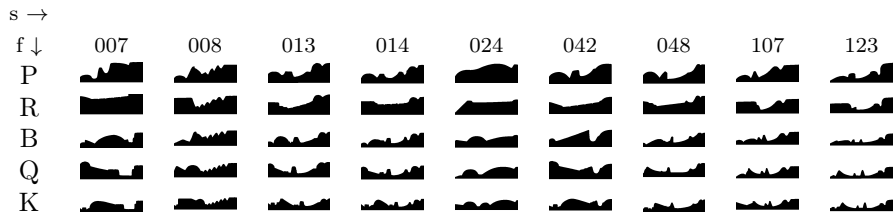


Table 2: Outline curves of the chess pieces generated from the three dimensional chess pieces displayed in table 1.

as a definition of style. Sometimes we also had a situation when introducing a difficult set into the training data improved the results, because then the labeling of such style was not interfering with the labeling of other styles. So by putting into the training data a set which has a difficult to distinguish style but a clear function or putting a difficult to distinguish function but a clear style distinction usually improved the results.

It is also interesting to see how the mismatches looked like. For that reason we have displayed the sorting results for two cases. In table 5 we have a situation that swaps were done within style 008 within pawns and one 3-cycle within

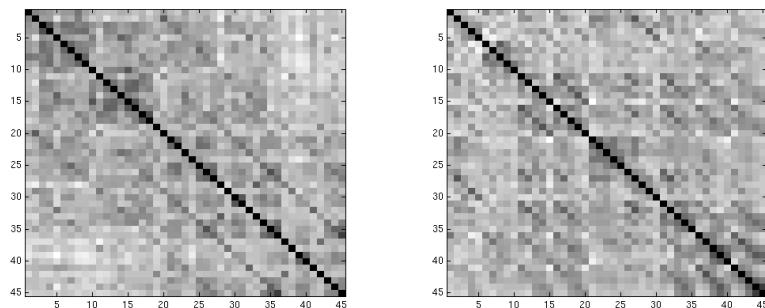


Figure 14: Similarities between the curves. Each block has the same function or style, the diagonals of blocks are darker which reflects the smaller distance when the function or the style is the same.

rooks, however in the table 6 a more complicated long cycle running through many styles and functions is present. Note also that some swaps might have a very logical explanation - for example very often something else of a style 008 is labeled as a king instead of the K008. This is because 008 pieces have a kind of collared shape at the top, which might be confused with the upper collared shape of the training king. Note also that bishop 008 as a training shape has attracted shapes P024 and K042, this might be due to the fact that all those three shapes have a rounded barrel shape.

For the multiple step assignment (results: table 4, method: refer to section 2.2.1) we observe an average improvement of the assignment tasks by approximately 3 chess pieces. Usually if the initial guess is quite good but not perfect then correctness of the matching may be improved quite well. If there are too many mismatches the improvement does not occur: as then we also take as reliable the matchings which are not correct. Usually it does not make the solution worse but keeps it at a similar level as it was with the initial problem.

5.2 3D Tooth Consistency

A dataset we use for this problems contains teeth shapes (table 7) from 6 different mouths. We treat the type of mouth as style and the tooth type as a function. In order to make the number of styles larger, we assume that the left side of a mouth will be treated separately from the right side. This assumption is correct as long as we don't use any descriptors related to symmetry orientation. Thus we have 12 styles which we will label as $A, B, C, D, E, F, a, b, c, d, e, f$, where big letter means one left part of a mouth and small the other one. We have taken 10 tooth types: 2 upper molars, lower molar, 2 upper premolars, lower premolar, upper canine, upper incisor, 2 down incisors. They are labeled and placed in the following order: $7M, 6M, 6m, 5P, 4P, 4p, 3C, 1I, 1i, 2i$, where upper case means the upper tooth.

| | P | R | B | Q | K | mean |
|------|------|------|------|------|------|------|
| 007 | 15 | 23 | 8 | 10 | 21 | 15.4 |
| f | 8 | 12 | 6 | 7 | 12 | 9 |
| s | 14 | 15 | 4 | 6 | 16 | 11 |
| 008 | 22 | 24 | 19 | 16 | 24 | 21 |
| f | 18 | 21 | 15 | 10 | 17 | 16.2 |
| s | 14 | 18 | 9 | 7 | 11 | 11.8 |
| 013 | 14 | 24 | 6 | 6 | 19 | 13.8 |
| f | 7 | 11 | 2 | 2 | 8 | 6 |
| s | 8 | 15 | 4 | 4 | 13 | 8.8 |
| 014 | 20 | 20 | 10 | 9 | 14 | 14.6 |
| f | 6 | 11 | 2 | 2 | 6 | 5.4 |
| s | 17 | 13 | 8 | 7 | 10 | 11 |
| 024 | 28 | 27 | 19 | 25 | 28 | 25.4 |
| f | 20 | 22 | 15 | 20 | 22 | 19.8 |
| s | 16 | 12 | 8 | 12 | 14 | 12.4 |
| 042 | 20 | 24 | 21 | 14 | 18 | 19.4 |
| f | 14 | 14 | 13 | 9 | 10 | 12 |
| s | 11 | 16 | 9 | 7 | 12 | 11 |
| 048 | 16 | 17 | 11 | 10 | 17 | 14.2 |
| f | 7 | 6 | 4 | 2 | 8 | 5.4 |
| s | 14 | 12 | 8 | 8 | 12 | 10.8 |
| 107 | 17 | 20 | 16 | 12 | 17 | 16.4 |
| f | 13 | 12 | 10 | 5 | 9 | 9.8 |
| s | 12 | 15 | 11 | 7 | 10 | 11 |
| 123 | 9 | 24 | 12 | 17 | 18 | 18 |
| f | 13 | 11 | 5 | 9 | 10 | 9.6 |
| s | 13 | 21 | 9 | 10 | 10 | 12.6 |
| mean | 19 | 22.6 | 13.6 | 13.2 | 19.6 | 17.6 |
| f | 11.8 | 13.3 | 8 | 7.3 | 11.3 | 10.4 |
| s | 13.2 | 15.2 | 7.8 | 7.6 | 12 | 11.2 |

Table 3: Mismatches of the single assignment problem with one style and one function given. The table contains the general number of pieces with mismatched total label, mismatched function and the mismatched style. The performance depends on the choice of shapes that were used for the definition of the style and the function.

The descriptors from the section 4.2 were used for which the total consistency factors were computed as mentioned in section 3.1 and the final dissimilarity measure is created from the measures.

In the first experiment we analyze the metric obtained through the consistency learning process. Because of the size of our dataset we included all of the teeth data. Figure 15 contains the resulting dissimilarity measure. What is worth noting is that the similarity between corresponding styles coming from

| | P | R | B | Q | K | mean |
|------|------|------|-----|-----|------|------|
| 007 | 17 | 25 | 6 | 8 | 11 | 13.4 |
| f | 8 | 9 | 2 | 2 | 6 | 5.4 |
| s | 13 | 19 | 4 | 6 | 5 | 9.4 |
| 008 | 22 | 28 | 23 | 9 | 8 | 18 |
| f | 18 | 22 | 14 | 4 | 7 | 13 |
| s | 13 | 25 | 14 | 6 | 5 | 12.6 |
| 013 | 15 | 21 | 0 | 0 | 19 | 11 |
| f | 8 | 11 | 0 | 0 | 11 | 6 |
| s | 9 | 14 | 0 | 0 | 12 | 7 |
| 014 | 14 | 12 | 5 | 4 | 17 | 10.4 |
| f | 8 | 8 | 0 | 0 | 9 | 5 |
| s | 9 | 9 | 5 | 4 | 11 | 7.6 |
| 024 | 24 | 25 | 18 | 26 | 27 | 24 |
| f | 20 | 18 | 14 | 20 | 20 | 18.4 |
| s | 14 | 18 | 7 | 12 | 11 | 12.4 |
| 042 | 22 | 21 | 15 | 17 | 23 | 19.6 |
| f | 13 | 14 | 10 | 15 | 13 | 13 |
| s | 14 | 15 | 7 | 2 | 16 | 10.8 |
| 048 | 19 | 19 | 2 | 4 | 12 | 11.2 |
| f | 11 | 9 | 2 | 0 | 5 | 5.4 |
| s | 14 | 14 | 0 | 4 | 9 | 8.2 |
| 107 | 15 | 14 | 10 | 5 | 11 | 11 |
| f | 6 | 7 | 6 | 0 | 5 | 4.8 |
| s | 13 | 13 | 5 | 5 | 8 | 8.8 |
| 123 | 19 | 23 | 9 | 7 | 15 | 14.6 |
| f | 11 | 11 | 4 | 2 | 9 | 7.4 |
| s | 10 | 18 | 5 | 5 | 10 | 9.6 |
| mean | 18.6 | 20.9 | 9.8 | 8.9 | 15.9 | 14.8 |
| f | 11.4 | 12.1 | 5.8 | 4.8 | 9.4 | 8.7 |
| s | 12.1 | 16.1 | 5.2 | 4.9 | 9.7 | 9.6 |

Table 4: Mismatches of the multiple assignment problem with one style and one function given. The table contains the general number of pieces with the mismatched total label, mismatched function and mismatched style. The improvement occurs usually for tasks where the single assignment solution didn't have too many mismatches, otherwise the performance stays similar to the performance in the case of the one step assignment.

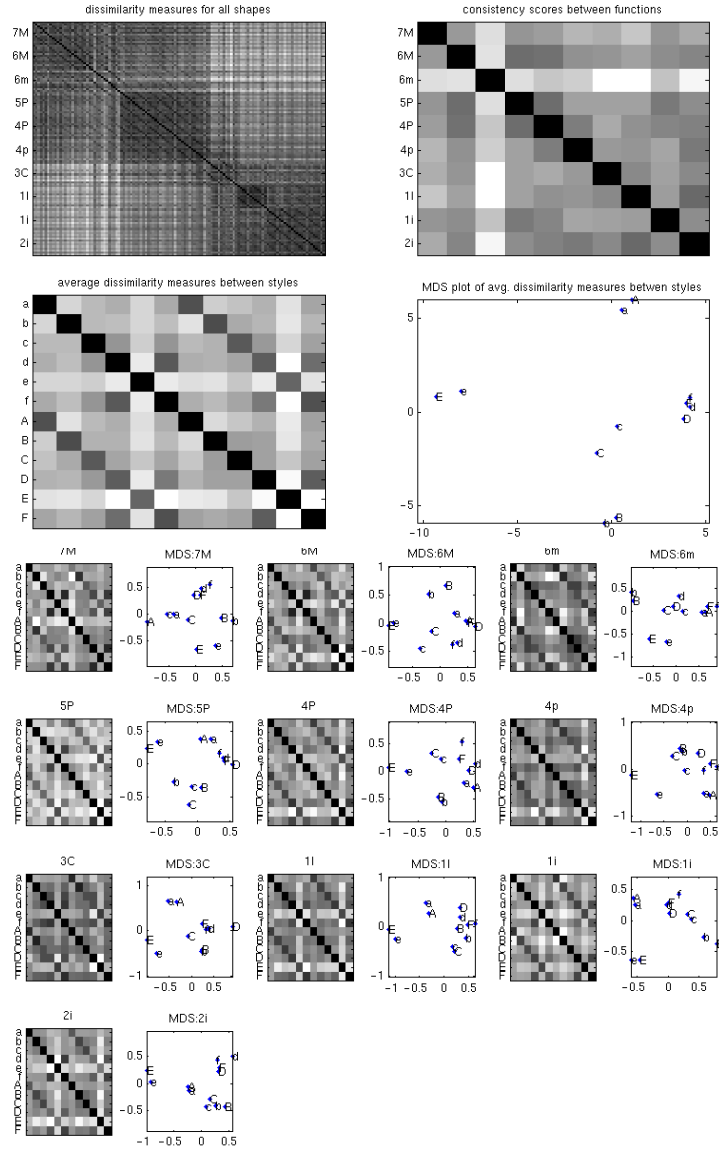


Figure 15: Dissimilarity measures obtained with the consistency weights. Note similarities between styles coming from the left and right sides of the mouths (second upper left plot) and low consistency scores (upper right plot) calculated between similar teeth: for example upper neighbors: 6M,5P,4P or upper and lower first premolar (4P and 4p) or the incisors (1I and 2i).

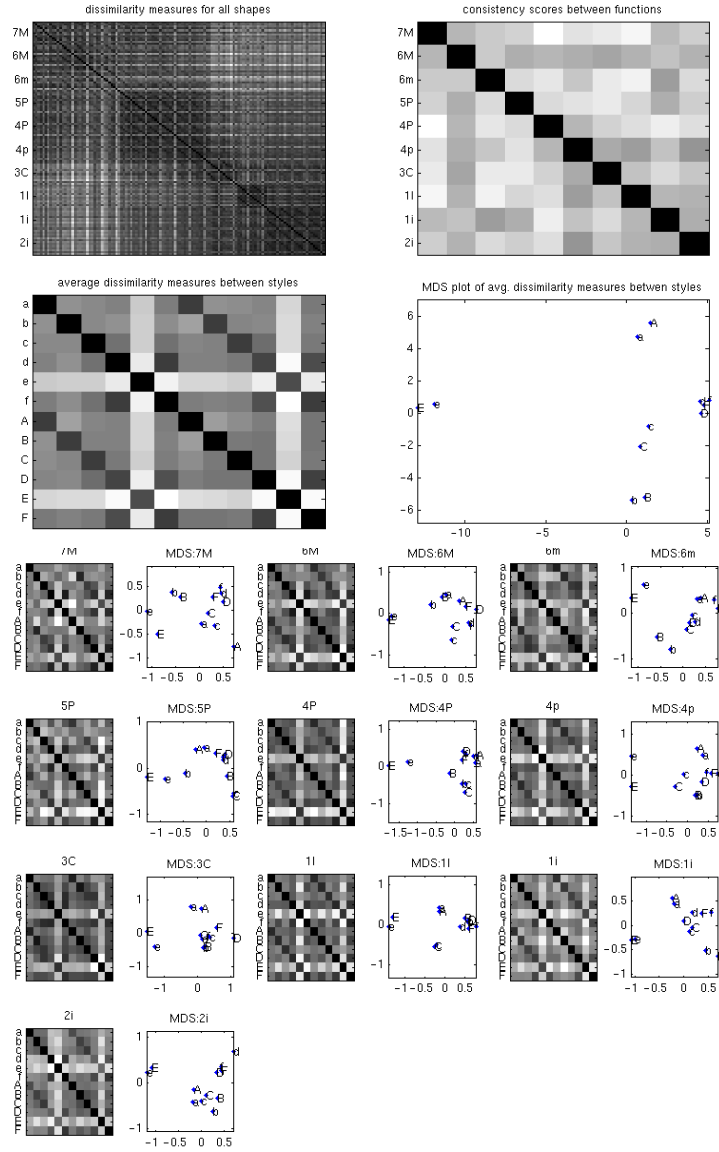


Figure 16: Dissimilarity measure obtained with the equal weights. The average structure of differences between styles seem to be similar to the consistency customized measure, besides the tendency of styles e, E to become even more distant than the rest of the shapes. Note however that the consistency scores are very different than the previous example. Also there are higher differences between the lower plots when compared to figure 15

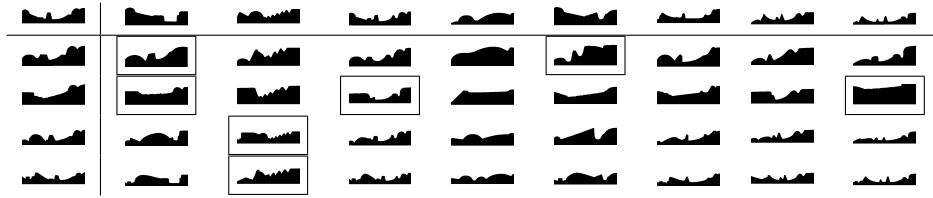


Table 5: Example where queens define the styles and 013 define functions. We have one 3 cycle within rooks, a swap between pawns and between pieces from 008 style.

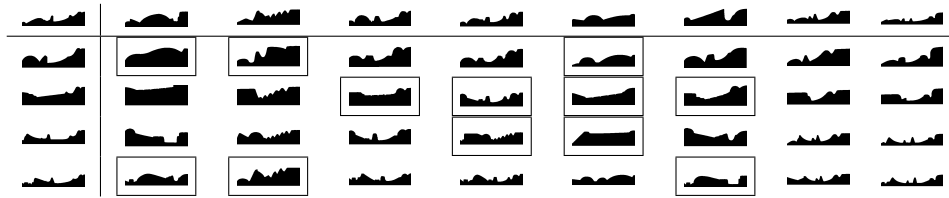


Table 6: Example where bishops define the styles and 048 define functions. Note that mismatched assignments can be explained by geometric similarities to the training shapes.

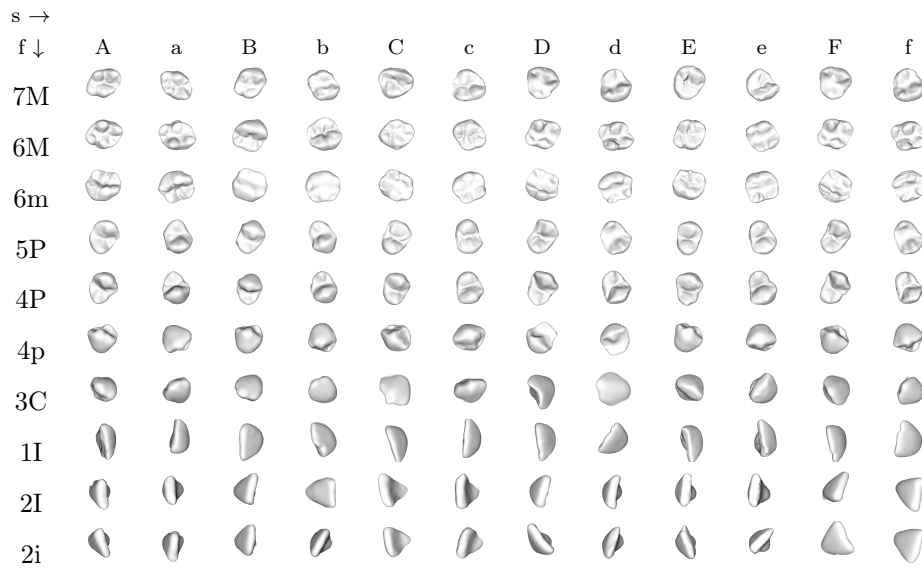


Table 7: Tooth dataset

symmetric teeth was clear. During our tests we have discovered that the styles d, D and f, F are very similar and they probably came from the same mouth but are differently meshed. In order to compare the obtained dissimilarity measure

with the situation without consistency information, we created the other measure as the average of all available measures (figure 16). From the first glance, just when looking at the average distances between styles, the results look very similar no matter if we use the consistency weights or not. However if we look closer and analyze the dissimilarities between the classes functionwise we can see the difference (lower plots of figure 15 and figure 16). Dissimilarities with consistency weights are more coherent than the average dissimilarities. Also note that the consistency matrix of the average looks more random (upper right plot of 15 and 16), while the one with the weights has more intuitive information: we can see that the molars are more consistent with each others than incisors, consistency also grows if we have neighbor teeth.

5.3 Replacing a Missing Tooth

In this section we show an application of the consistency based query method described in section 3.2 to a tooth dataset in the tooth replacing scenario.

We start by choosing a training dataset which has several exemplar styles where each style is complete, which means it contains all functions. Then we use the training dataset in all phases of computation when training is needed: to compute the measure error for histogram based descriptors, to compute the total consistency factors TCF_k for given descriptors and finally as a supply to select the best replacement for a missing shape.

We take some style which was not in the training set, and assume that one shape of a function i is missing. In order to replace it we choose from the training database a shape being of function i , which we estimate to be the closest to the missing one. Because we don't have the missing shapes we cannot measure the distances directly. Instead we make an estimation based on the distances $D_f(S_k^j, Q_k)$ between the shape of the query style and the functions k that we have and the shapes from the database sharing this function.

In our results we used three strategies of combining $D_f(S_k^j, Q_k)$:

average $D^a(s = j, Q)$ distances summed with equal weights (see section 3.2),

weighted $D^w(s = j, Q)$ distances summed with weights according to consistency factors (see section 3.2),

best direct use of $D_f(S_k^j, Q_k)$ for function k having the smallest consistency factor $cs_{D_f}^{ki}$ with the function i of a query object.

Note that the 'dissimilarity measure' D_f is used in all cases. As previously mentioned D_f is computed as consistency weighted according to the TCF score of a given descriptor. What is different here is how we combine the distance if we have more than one function to be checked. The weighted scheme takes distances from all functions with weight according to consistency factors evaluated between the function of an unknown object and the function for which we are computing the weight.

We have also added 'true' which is the distance to the shape we assumed is unknown to the shapes of the same function from the training database. It can

be seen as a ground truth. Note however that the distance D_f we use here was obtained by consistency learning as we don't have a ground truth distance.

| method | missing | 1 | 2 | 3 | 4 | 5 | 6 |
|-----------|---------|---|---|---|---|---|---|
| weighted | | | | | | | |
| | | | | | | | |
| average | | | | | | | |
| | | | | | | | |
| best (1i) | | | | | | | |
| | | | | | | | |
| true | | | | | | | |
| | | | | | | | |

Table 8: Replacing missing 1I b shape. For this case the difference between the candidates chosen by our methods is hard to spot visually. This illustrates that even if we get styles with different labels than the ground truth, this does not mean that the alternative choice is bad. The other candidate can also be a good replacement for the missing object.

| method | missing | 1 | 2 | 3 | 4 | 5 | 6 |
|-----------|---------|---|---|---|---|---|---|
| weighted | | | | | | | |
| average | | | | | | | |
| best (5P) | | | | | | | |
| true | | | | | | | |

Table 9: Replacing missing 6M b. Visual inspection indicates that weights and average strategy seem to choose the most similar tooth - this tooth was also chosen by the ground truth distance.

The results for different queries are shown in the form of labeled plots of distances. We included results obtained with a training dataset *cdfACD*, and with queries made for all of the functions of styles *a*, *b*, *E* and *F* (figures 17, 18, 19 and 20 respectively).

From those examples we can see that for our dataset we have some queries which are very easy as for example *a* (figure 17) and always result in finding the symmetric counterpart. When searching for *F* (figure 20) there is label shuffling at the top of the rankings, but all pieces which are assessed to be the closest come from the very similar styles *f*, *d*, *D*: either from differently meshed teeth or symmetric counterparts. Thus even though there is a change in the

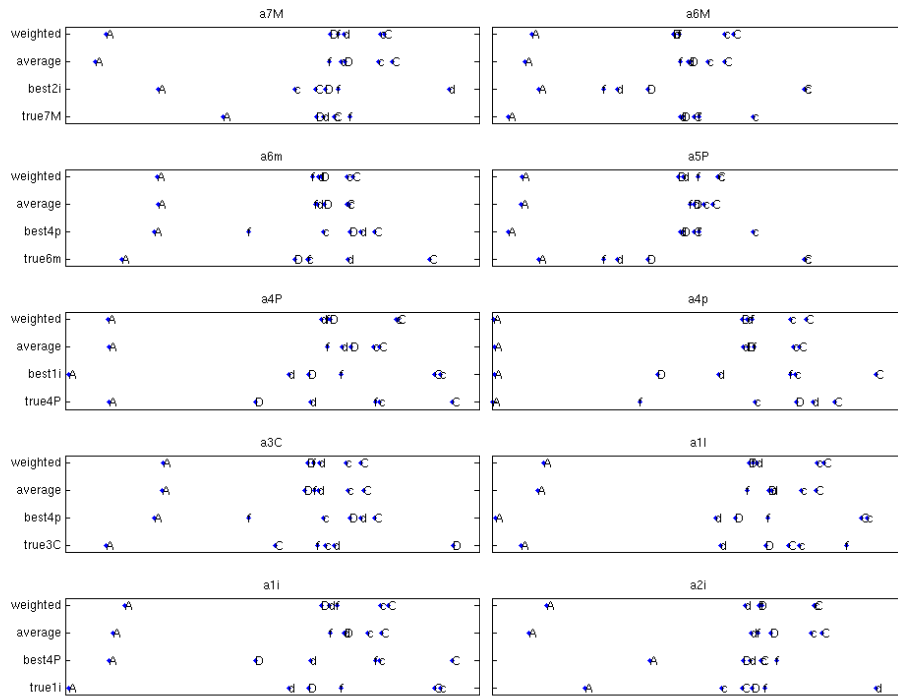


Figure 17: Replacing object of a style a , we assume we have all the functions of style a except one. Training dataset contains $cdfACD$ styles. This is an easy case as the winner (the shape of the smallest distance), comes from a symmetric counterpart of the missing style.

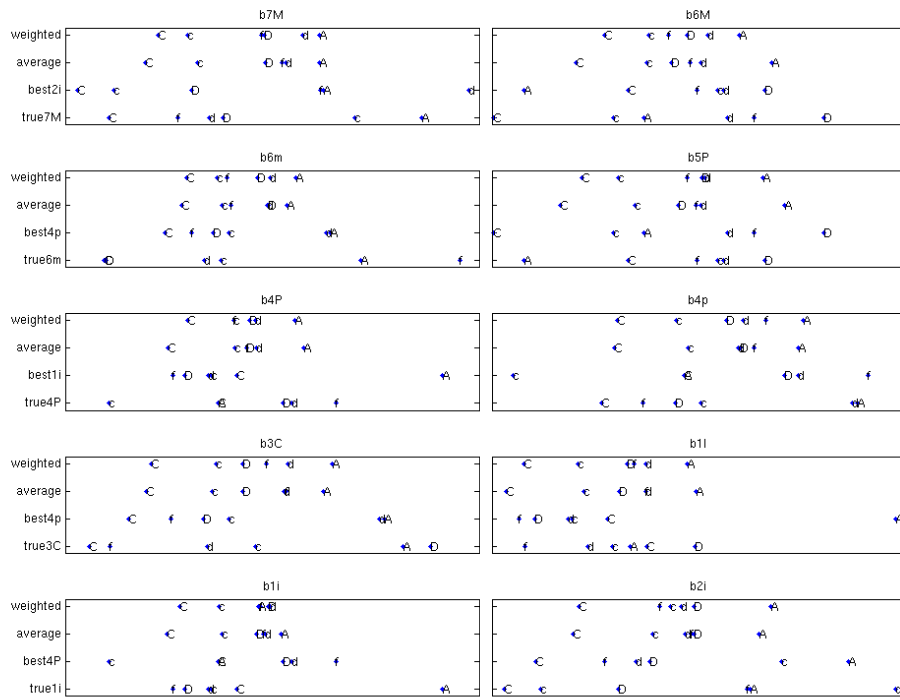


Figure 18: Replacing b , we assume we have all the functions of style a except one. Training dataset contains $cdfACD$ styles. The average replacement style is C but some exceptions occur especially within the domain of incisors.

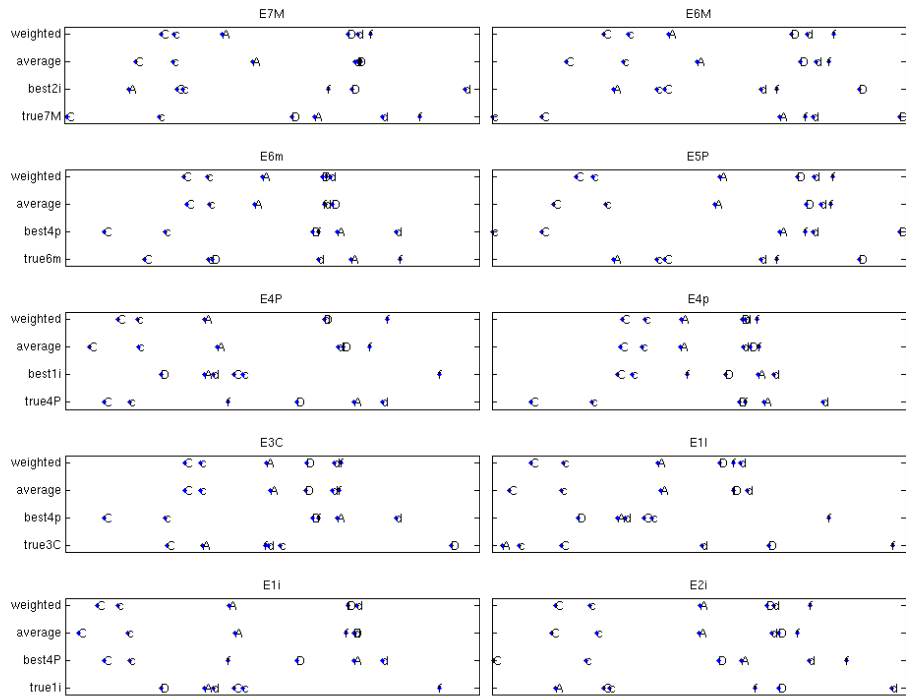


Figure 19: Replacing E , we assume we have all the functions of style a except one. Training dataset contains $cdfACD$ styles. As in the previous case we have styles C at the top of most rankings with minor exceptions for incisors and function $5p$.

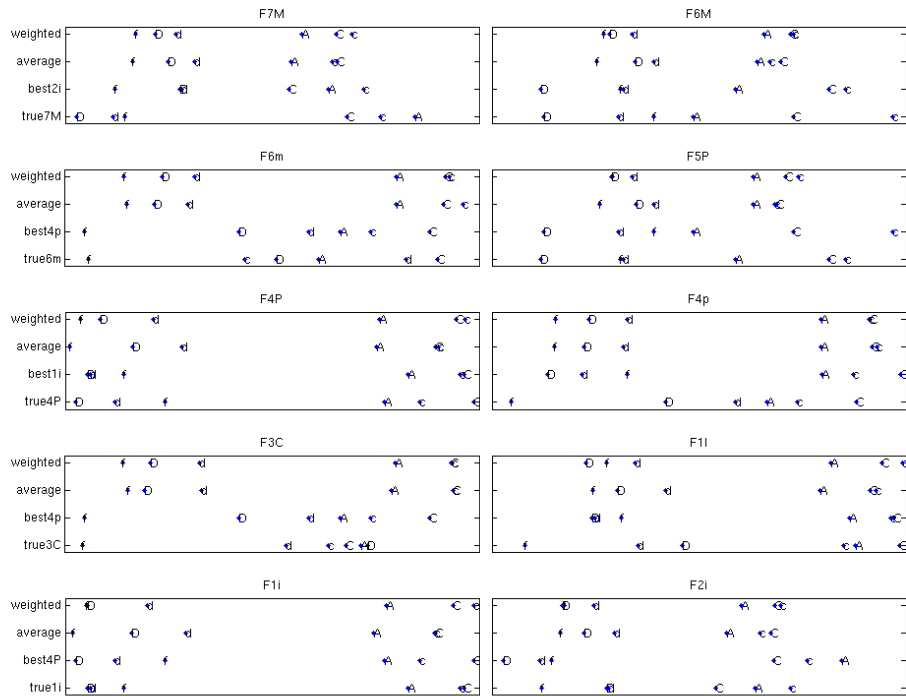


Figure 20: Replacing F , we assume we have all the functions of style a except one. Training dataset contains $cdfACD$ styles. Although there are some differences at the top of the replacement rankings we can see that all the top candidates come from styles very similar to the query style being f, d or D .

ranking, any of those candidates will do a good job as a replacement. More ambiguous are the queries for which we do not have a symmetric counterpart. As our dataset is not so big this corresponds to the case when we don't have a very close replacement. For such cases we see a tendency that for most of the functions there is a clear solution which is the average replacement that works - as the style C , but for some of the tooth types other candidates appear as top on the list: for example A or f when asking for b (figure 18) and A or D when asking for E (figure 19). Usually the ambiguous cases are among incisors - this might be a good hint to treat the front teeth separately from molars and premolars in order to reach better performance. For the ambiguous cases sometimes the 'best' strategy of replacement is a correct one, but it might also cause overfitting for other cases. The 'average' and 'weighted' strategies have a very similar results. We do not know however if the swaps are a method failure or if this is also the case when one shape might be as good as the other one. This can be left to subjective judgment of the user. To illustrate such case, we have picked two example queries from the figure 18: first molar of style (6M) and first incisor (1I) of style to be shown with the images of the ranked replacement candidates (table 9 and 8). For better insight we have added the front view of incisors. We have also reflected symmetric counterparts of the second part of the mouth (see dataset description in section 5.2) for easier spotting of the differences.

We also tested how the consistency properties of the dissimilarity measure change when different subsets of styles were used as the training dataset. We generated a dissimilarity measure from the training data and evaluated the results on all of the data. Usually removing only a small number of mouths does not increase or even slightly decreases the consistency scores. Only when using 3 or 4 mouths, the results seemed to be different. This might come from the fact that there was always some symmetric tooth left in the set which was able to set the consistency scores in a correct way. The increase was mostly noticeable when styles which are close to each other are used as the training set (table 10).

| training | TCS | training | TCS | training | TCS |
|-----------|--------|----------|---------|----------|--------|
| all | 97.94 | EFb | 102.088 | cEF | 98.17 |
| none | 117.22 | ABd | 99.138 | $ADEF e$ | 97.35 |
| $ABCDFaf$ | 95.929 | Eef | 117.107 | $cdfACD$ | 99.25 |
| $AFade$ | 98.494 | $DdFf$ | 111.007 | $deADEF$ | 115.04 |

Table 10: Total consistency factors when using different mouth subsets as training data. First the descriptors were trained by using training data, than all the data was used in order to evaluate TCF. One can see the impact of the choice of training shapes on the quality of the resulting measure. For example using data from very similar styles results in a measure which does not have good consistency for other styles.

5.4 Chess Pieces Consistency with Comparison To The Warping Distance

As we have 3D representation of chess pieces at our disposal we have applied 3d descriptor computation and the consistency framework, to see the performance of the consistency methods on the chess dataset. We compared the obtained results with the warping distance obtained when using the outline curves (figures 21 and 22). Some differences occur, however this might be due to the fact that when taking an outline curve a lot of information was lost, so the curve dataset and the style dataset are not totally equivalent. For example the set 008 was very distinguishable in the curve set. However this is not so true for the 3d set, which might be due to the fact that local details at the base are very similar to the details at the base of 007 and 107. Also the upper part has all function characteristic details while when having a curve representation the most dominant was the Christmas-tree-like middle part.

5.5 Empirical Connection Between Consistency Measure and Dissimilarities Between Styles

As it is much easier to make visual judgments for the dissimilarities between different chess sets than for teeth, we used the chess set example to show the experimentally based connection between the consistency score and dissimilarity measure between different styles. From the 3d images presented in Table 1 we assume that some sets can be seen as having standard shapes: 013, 014, 048, 123, 107, while 007 and 042 have the middle part missing and the upper part magnified, 008 is unusual but has a base similar to 007 and 024 - the 'babushka set' can be treated as an outlier.

From all of our descriptors we have chosen to plot dissimilarity measures for those descriptors which have the best (Figure 23), and the worst (Figure 24) consistency. We can see that the most consistent descriptor was much better in capturing our judgments, with having standard sets nearby and the non standard 'christmass tree' and 'babushka' as outliers. The least consistent descriptor does not preserve the structure of standard chess pieces being close. For example it treats the set 123 as the most unusual one.

It is worth noting that the TCF value when we used all descriptors combined was 15.19 while the TCF of the best descriptor was 19.35, and of the worst it was 69.27. We can clearly see that combining many descriptors (with weights according to TCF) is better than using just the most consistent descriptor.

5.6 Chess Sorting Based on Consistency Measures

In this experiment we sorted the chess pieces with the metric obtained from 3d descriptors from section 5.4. Tables 11 and 12 contain the results for both single and multistep assignment. The results are comparable with the sorting based on consistency measure.

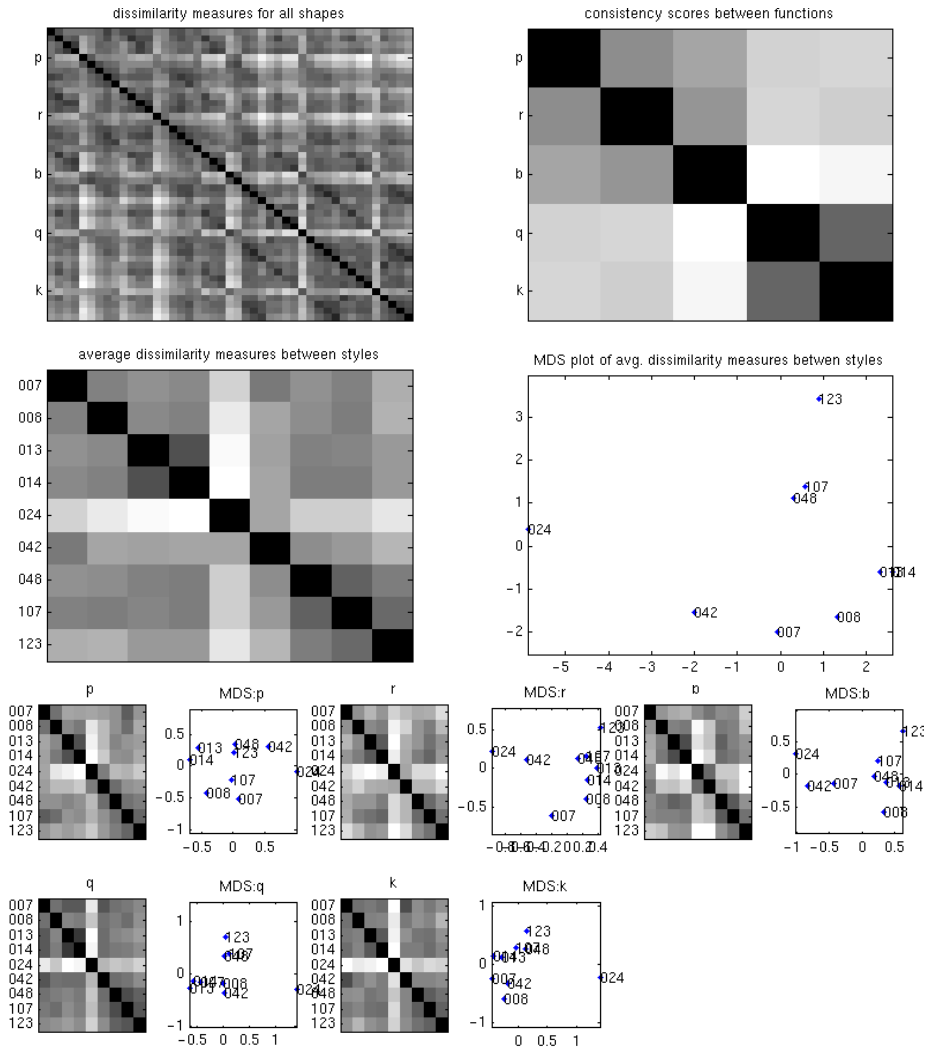


Figure 21: Chess piece dissimilarity measure from the 3d descriptors by using consistency weights. The styles are ordered 007,008,013,014,024,042,048,107,123 and the functions are ordered p,r,b,q,k. Note that 008 is not so distinctive as it was for the DTW curve measure. The main source of inconsistency comes from the style 042 which is more similar to the outlier set 024 for pawn, rook and bishop but has more standard shapes for king and queen.

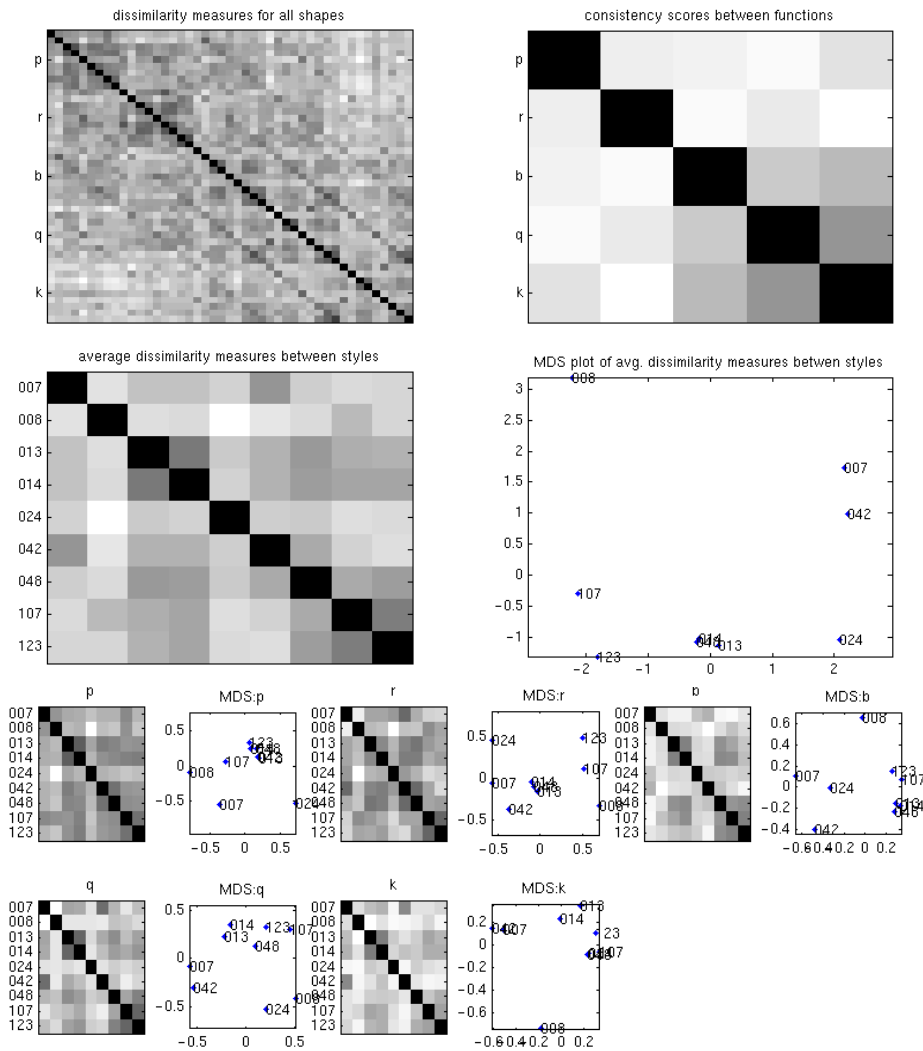


Figure 22: Curve warping distance displayed in the same form as consistency based distances. The structure of dissimilarities between styles is similar (with some exceptions like the style 008) to the structure of the dissimilarity coming from consistency weighted 3d descriptors but the general dissimilarity looks slightly different.

| | P | R | B | Q | K | mean |
|------|-------|-------|-------|-------|-------|-------|
| 007 | 24 | 17 | 12 | 9 | 20 | 16.4 |
| f | 16 | 10 | 4 | 6 | 8 | 8.8 |
| s | 23 | 15 | 12 | 8 | 18 | 15.2 |
| 008 | 21 | 23 | 20 | 15 | 24 | 20.6 |
| f | 9 | 7 | 10 | 8 | 7 | 8.2 |
| s | 21 | 23 | 19 | 15 | 23 | 20.2 |
| 013 | 19 | 16 | 16 | 3 | 16 | 14 |
| f | 12 | 7 | 6 | 2 | 8 | 7 |
| s | 18 | 16 | 16 | 3 | 14 | 13.4 |
| 014 | 11 | 13 | 12 | 5 | 10 | 10.2 |
| f | 5 | 7 | 2 | 4 | 6 | 4.8 |
| s | 11 | 13 | 12 | 5 | 10 | 10.2 |
| 024 | 25 | 22 | 17 | 12 | 20 | 19.2 |
| f | 10 | 11 | 9 | 6 | 10 | 9.2 |
| s | 25 | 21 | 16 | 12 | 19 | 18.6 |
| 042 | 22 | 23 | 10 | 9 | 17 | 16.2 |
| f | 14 | 13 | 4 | 2 | 6 | 7.8 |
| s | 21 | 22 | 10 | 9 | 16 | 15.6 |
| 048 | 13 | 17 | 16 | 14 | 17 | 15.4 |
| f | 9 | 9 | 7 | 8 | 7 | 8 |
| s | 12 | 17 | 16 | 14 | 16 | 15 |
| 107 | 16 | 13 | 12 | 9 | 15 | 13 |
| f | 11 | 10 | 7 | 7 | 4 | 7.8 |
| s | 15 | 13 | 12 | 9 | 15 | 12.8 |
| 123 | 26 | 17 | 19 | 19 | 19 | 20 |
| f | 15 | 11 | 3 | 11 | 6 | 9.2 |
| s | 25 | 16 | 19 | 19 | 19 | 19.6 |
| mean | 19.67 | 17.89 | 14.89 | 10.56 | 17.56 | 16.11 |
| f | 11.22 | 9.44 | 5.78 | 6 | 6.89 | 7.87 |
| s | 19 | 17.33 | 14.67 | 10.44 | 16.67 | 15.62 |

Table 11: Mismatches of the single assignment problem with one style and one function given. The table contains the general number of pieces with mismatched total label, mismatched function and the mismatched style. Results are comparable with the DTW sorting results presented in table 3.

| | P | R | B | Q | K | mean |
|------|-------|-------|------|------|-------|-------|
| 007 | 25 | 19 | 0 | 0 | 26 | 14 |
| f | 15 | 12 | 0 | 0 | 15 | 8.4 |
| s | 25 | 17 | 0 | 0 | 25 | 13.4 |
| 008 | 10 | 17 | 14 | 4 | 26 | 14.2 |
| f | 4 | 9 | 8 | 4 | 12 | 7.4 |
| s | 10 | 16 | 14 | 3 | 26 | 13.8 |
| 013 | 16 | 13 | 0 | 5 | 12 | 9.2 |
| f | 8 | 2 | 0 | 5 | 4 | 3.8 |
| s | 16 | 13 | 0 | 5 | 12 | 9.2 |
| 014 | 15 | 15 | 0 | 5 | 13 | 9.6 |
| f | 6 | 5 | 0 | 5 | 3 | 3.8 |
| s | 15 | 15 | 0 | 5 | 13 | 9.6 |
| 024 | 22 | 22 | 0 | 5 | 24 | 14.6 |
| f | 14 | 11 | 0 | 5 | 11 | 8.2 |
| s | 22 | 20 | 0 | 4 | 22 | 13.6 |
| 042 | 21 | 19 | 0 | 0 | 14 | 10.8 |
| f | 12 | 13 | 0 | 0 | 5 | 6 |
| s | 19 | 18 | 0 | 0 | 14 | 10.2 |
| 048 | 17 | 15 | 5 | 3 | 18 | 11.6 |
| f | 10 | 7 | 2 | 3 | 9 | 6.2 |
| s | 16 | 14 | 5 | 2 | 17 | 10.8 |
| 107 | 17 | 23 | 0 | 6 | 13 | 11.8 |
| f | 10 | 16 | 0 | 5 | 6 | 7.4 |
| s | 16 | 22 | 0 | 6 | 13 | 11.4 |
| 123 | 18 | 21 | 6 | 2 | 20 | 13.4 |
| f | 10 | 9 | 2 | 2 | 11 | 6.8 |
| s | 18 | 19 | 6 | 2 | 17 | 12.4 |
| mean | 17.89 | 18.22 | 2.78 | 3.33 | 18.44 | 12.13 |
| f | 9.89 | 9.33 | 1.33 | 3.22 | 8.44 | 6.44 |
| s | 17.44 | 17.11 | 2.78 | 3 | 17.67 | 11.6 |

Table 12: Mismatches of the single assignment problem with one style and one function given. The table contains the general number of pieces with mismatched total label, mismatched function and the mismatched style. The results are by approximately 2 pieces better than the DTW distance results presented in table 4.

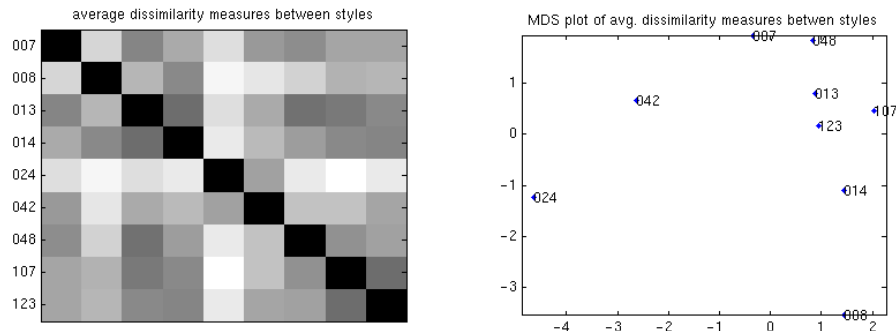


Figure 23: Average distances between styles for the descriptor with the best consistency: $TCS = 19.35$. This descriptor is the histogram of the second eigenvalue of the slippage matrix at scale 0.16.

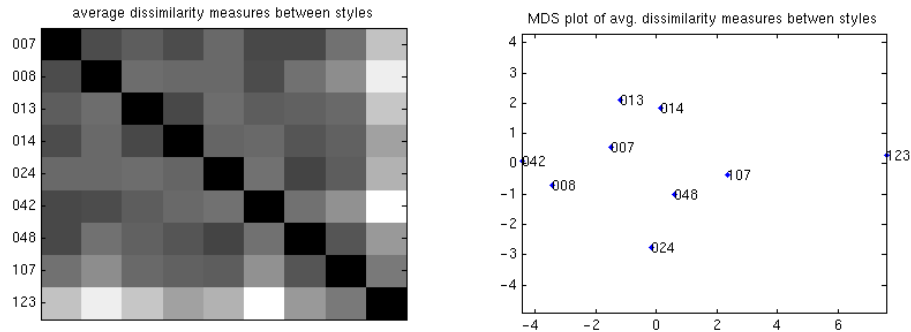


Figure 24: Average distances between styles for the descriptor with the worst consistency: $TCS=69.27$. This descriptor is the histogram of the translational impact of the last eigenvalue of the slippage matrix at scale 0.01

Note that in order to obtain the consistency measures we trained the dataset with all labels known, then we forgot the labels and remembered only the consistency scores. With dissimilarities computed according to the consistency we try to solve the reverse problem - finding the labels back. The sorting and consistency are two different problems, for the consistency we optimize something else - so one does not have a guarantee how well the sorting task will work with the dissimilarity measure obtained by using consistency scores. However, trying to retrieve the labels is an interesting test which measures if consistency based dissimilarity measures are able to store the style and function information that was provided in the training labels. From the sorting results we can see that at least we are not worse than curve matching distance and that the multilevel step outperformed the DTW based distance.

5.7 Descriptors

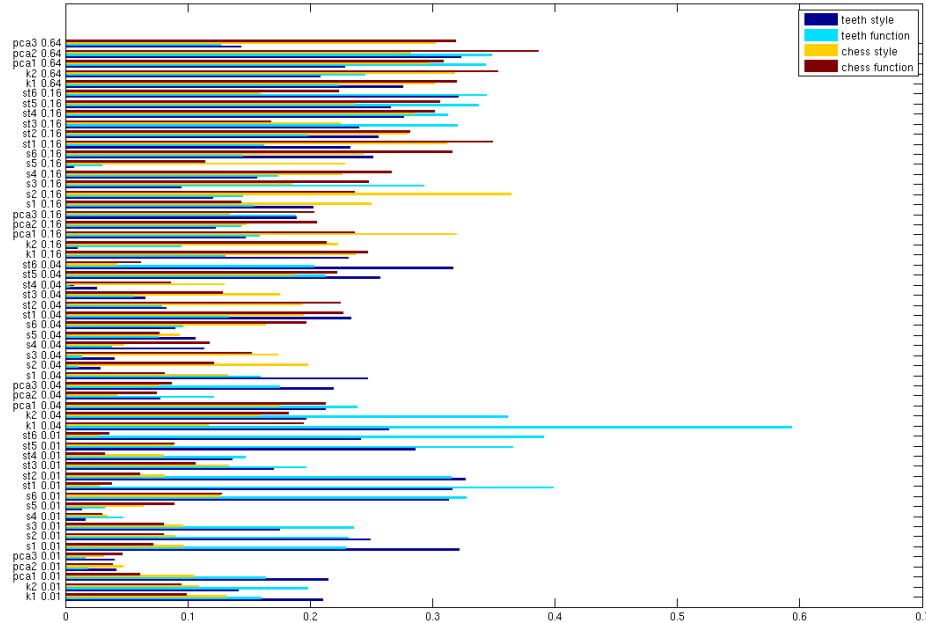


Figure 25: Consistency weights used to create the style and function consistent descriptors for tooth and chess datasets. Note that there is larger difference of weights between the datasets than between styles and functions within one dataset.

The consistency measure process can be seen as a black box where we throw the data with a bunch of descriptors and we obtain the consistency customized measure as an output. However it is very interesting to look closer and study the consistency weights in order to see which descriptors had higher impact. For this reason we displayed the style consistency weights for chess and tooth datasets. For both datasets we also exchanged the style and function labels and obtained function consistency weights. All those weights are displayed in figure 25. From this plot we see that different descriptors are distinctive for different datasets. Note that for the tooth dataset we have higher weights for smaller scale, for the medium scale the weights get smaller and for bigger scale they grow slightly again. This might be due to the fact that we have organic shapes where their local roughness has importance, but also in the higher scale the general shape counts especially when it comes to determine a function. For the chess pieces the global properties seem to be the more important than the local ones, as for man made objects the global structure of the shape has much importance (such as the existence of big flat or rounded regions or the proportions of such regions).

We can also observe that within one dataset if a descriptor has a high impact

on style (or function) consistency it very often has higher impact on function (or style) of that dataset. This indicates that in such cases we can find descriptors which are appropriate for style and function classification for a given dataset but those descriptors are not purely responsible for style or function but are distinctive for both.

6 Discussions

In this paper we presented a general framework, which avoids defining style related features explicitly. Instead we introduced a method of finding valuable style related descriptors by applying consistency criteria to the example dataset. We have also shown that even without pointing out the descriptors containing pure style related properties, some practical tasks can be achieved by factoring out the main property of an object, its function. Therefore we do not claim to have discovered a single descriptor or a set of descriptors which are responsible for style or for function. But we claim that if we have descriptors which at least contain some of the style information, and possibly other stuff, we can still perform well in at least some of the tasks related to the object that have both style and the function. We also confirm the need for treating shapes in broader context than just one shape [11, 12] and its geometric descriptors, by analyzing whole sets of shapes from the specific domain.

References

- [1] Vedrana Andersen. Height and tilt geometric texture, 2009. Presented at: Danish Conference on Pattern Recognition and Image Analysis ; 17 : Ven, 2009.
- [2] Rainer Burkard, Mauro Dell’Amico, and Silvano Martello. *Assignment Problems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009.
- [3] Alon Efrat, Quanfu Fan, and Suresh Venkatasubramanian. Curve matching, time warping, and light fields: New algorithms for computing similarity between curves. *J. Math. Imaging Vis.*, 27(3):203–216, 2007.
- [4] Ran Gal, Ariel Shamir, Tal Hassner, Mark Pauly, and Daniel Cohen-Or. Surface reconstruction using local shape priors. In *SGP ’07: Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 253–262, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.
- [5] D. Giorgi, P. Frosini, M. Spagnuolo, and B. Falcidieno. 3d relevance feedback via multilevel relevance judgements. *Vis. Comput.*, 26:1321–1338, October 2010.

- [6] Afzal Godil, Helin Dutagaci, Ceyhun Burak Akgül, Apostolos Axenopoulos, Benjamin Bustos, Mohamed Chaouch, Petros Daras, Takahiko Furuya, Sebastian Kreft, Zhouhui Lian, Thibault Napoleon, Athanasios Mademlis, Ryutarou Ohbuchi, Paul L. Rosin, Bülent Sankur, Tobias Schreck, Xianfang Sun, Masaki Tezuka, Anne Verroust-Blondet, M. Walter, and Yücel Yemez. Shrec'09 track: Generic shape retrieval. In Michela Spagnuolo, Ioannis Pratikakis, Remco C. Veltkamp, and Theoharis Theoharis, editors, *3DOR*, pages 61–68. Eurographics Association, 2009.
- [7] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 327–340, New York, NY, USA, 2001. ACM.
- [8] Aaron Hertzmann, Nuria Oliver, Brian Curless, and Steven M. Seitz. Curve analogies. In *EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering*, pages 233–246, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association.
- [9] G.J. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. Wiley, 1992.
- [10] Gelfand N. and L. Guibas. Shape segmentation using local slippage analysis. In *Proc. Symp. Geom. Processing*, 2004.
- [11] Andy Nguyen, Mirela Ben-Chen, Katarzyna Welnicka, Yinyu Ye, and Leonidas Guibas. An optimization approach to improving collections of shape maps. *Computer Graphics Forum*, 30(5):1481–1491, 2011.
- [12] Maks Ovsjanikov, Wilmot Li, Leonidas Guibas, and Niloy J. Mitra. Exploration of continuous variability in collections of 3d shapes. *ACM Transactions on Graphics*, 30(4):33, 2011.
- [13] Mark Pauly, Niloy J. Mitra, Joachim Giesen, Markus Gross, and Leonidas J. Guibas. Example-based 3d scan completion. In *SGP '05: Proceedings of the third Eurographics symposium on Geometry processing*, page 23, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.
- [14] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 399–405, New York, NY, USA, 2004. ACM.
- [15] Johan W. H. Tangelder and Remco C. Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia Tools Appl.*, 39(3):441–471, 2008.
- [16] G. Taubin. Estimating the tensor of curvature of a surface from a polyhedral approximation, 1995.

- [17] Joshua B. Tenenbaum and William T. Freeman. Separating style and content with bilinear models. *Neural Comput.*, 12:1247–1283, June 2000.
- [18] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*, volume 15, pages 505–512, 2002.
- [19] Kai Xu, Honghua Li, Hao Zhang, Daniel Cohen-Or, Yueshan Xiong, and Zhiqian Cheng. Style-content separation by anisotropic part scales. *ACM Transactions on Graphics, (Proc. of SIGGRAPH Asia 2010)*, 29(5):to appear, 2010.
- [20] Christopher Zach, Manfred Klopschitz, and Manfred Pollefeys. Disambiguating visual relations using loop constraints. In *CVPR'10*, pages 1426–1433, 2010.