Combinatorial methods for approximate pattern matching under rotations and translations in 3D arrays[1]

Kimmo Fredriksson and Esko Ukkonen
Department of Computer Science, University of Helsinki
PO Box 26, FIN–00014 Helsinki, Finland
kfredrik@cs.Helsinki.FI, ukkonen@cs.Helsinki.FI

Abstract

We consider the problem of defining and evaluating the distance between three–dimensional pattern $P[1..m, 1..m, 1..m]$ of voxels and three–dimensional volume $V[1..n, 1..n, 1..n]$ of voxels when also rotations of P are allowed. In particular, we are interested in finding the orientation and location of P with respect of V that gives the minimum distance. We consider several distance measures. Our basic method works for all distance measures such that the voxels affect the distance between P and V only locally, that is, the distance between two voxels can be computed in unit time. The number of different orientations that P can have is analyzed. We give incremental algorithms to compute the distance, and several filtering algorithms to compute the upper and lower bounds for the distance. We conclude with experimental results on real data (three dimensional reconstruction of a biological virus).

1. Introduction

String matching, and more generally combinatorial pattern matching, is one of the most successful special areas of algorithmics, with a wide spectrum of important applications ranging from text processing to information retrieval and genome analysis. The theory and potential applications of string matching techniques in the case of one–dimensional data, that is, linear strings and sequences, is well understood. However, the string matching approach still has considerable unexplored potential in the treatment of pattern matching problems when the data is more complicated than just linear string. Two dimensional digital images and three dimensional arrays of voxels are examples of such data.

In this paper we consider combinatorial pattern matching in 3D arrays of voxels. We give a precise combinatorial formulation of the approximate pattern matching problem, analyze its inherent complexity and develop several algorithms for solving the problem.

Before going into more details, let us consider a motivating application of the algorithms to be developed. In computational structural biology, 3D arrays of voxels are used as the data structure for representing three–dimensional models of biological viruses (Fig. 1) and other macromolecular assemblies. The array is typically an $n \times n \times n$ cube of voxels, where $100 < n < 300$. It represents the spatial distribution of the mass of the virus in the three–dimensional real space: each of the n^3 voxels of the array has a density value which typically is a small integer in the range $0..255$. Such models are produced, e.g. from two–dimensional electron microscopy images (projections) of the virus. The model construction itself is a very challenging problem in computer tomography (see e.g. [3]).

Once a model has been constructed, it is of great interest to compare different models against each other and especially to search for known substructures inside the new model. Such a substructure can be, e.g., a protein that is known to occur in the shell of the virus. The substructure can again be represented as a 3D array of voxels, say an $m \times m \times m$ array. Here m typically is clearly smaller than n. Hence we obtain a pattern matching problem that is analogous to the classical string matching problem of finding the occurrences of the key–word in a long text. Now we want to find the occurrences of a small 3D array inside a larger one. Here, however, the situation is complicated by the fact that, in addition to translating the pattern through all positions of the larger array, all possible spatial rotations of the pattern should be allowed in the matching. This is because the interesting substructure can be inside the virus model in any orientation. Moreover, the pattern matching should be approximate such that small differences in the mass values of the matching voxels are tolerated. Therefore we need a distance function that gives the distance between the pattern and the volume of the model that is covered by the pattern, for any translation and rotation of the pattern with respect

[1] A work supported by ComBi and the Academy of Finland.

to the model.

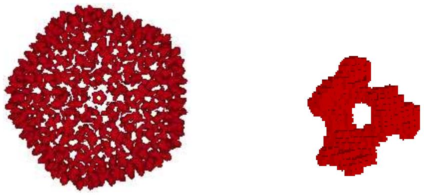

Figure 1. Left: An isosurface visualization of a 3D model ($119 \times 119 \times 119$ voxels) of the *sus1* mutant of the *PRD1* bacteriophage (virus). Right: A voxel map of a substructure extracted from the virus shell (not in the scale).

The Problem. Our problem is, given an $n \times n \times n$ array V of voxels (the model) and an $m \times m \times m$ array P of voxels (the pattern), such that each voxel p has a integer value $color(p)$, to find the translations and rotation of P with respect of V, such that the corresponding distance is smallest possible (the best occurrences problem) or, given a positive κ, to find all translations and rotations such that the distance is $\leq \kappa$ (the κ–threshold problem).

To fully define the problem we have still to fix the distance function. This is complicated by the translations and rotations: in a given superposition of the pattern and the model, the voxels of P and V do not necessarily overlap exactly. Therefore it is not clear what pairs of voxels one should compare. We proceed as follows: V and P are interpreted as arrays of small cubes in the three–dimensional Euclidean space, and each voxel p of P is compared with the voxel $M(p)$ of V that is closest to p. Here the distance between voxels is measured as the Euclidean distance between the center points of the voxels. This is also technically convenient. Voxel $M(p)$ is simply the voxel of the model into which the center of p belongs in the given superposition. (Note that the voxel boundaries form the Voronoi diagram for the centers of the voxels.) In this way we get a matching M between the voxels of pattern P and model V. The distance itself is a value of the form

$$\sum_{p \in P} d(p), \tag{1}$$

where $d(p)$ gives the contribution of p, based on the values of voxels p and $M(p)$. The function d may be e.g.

- $d(p) = 0$, if $color(p) = color(M(p))$, and 1 otherwise;
- $d(p) = |color(p) - color(M(p))|$.

The first one leads to a Hamming distance type of distance function, while the second measures the cumulative differences in the voxel values to be matched under M.

The Results. The problem of finding the best occurrences of P in V under distance function (1) leads to the problem of finding the matchings M that give the best distances. We show that the number of different M is $\mathcal{O}(m^{16}) = \mathcal{O}(|P|^{5\frac{1}{3}})$ when the center of the whole P is allowed to move inside a fixed voxel of V. Hence the total number of relevant matching functions is as high as $\mathcal{O}(|V||P|^{5\frac{1}{3}})$. Going through all of them would lead to impractically slow algorithms, expect for very small $|P|$. Even if we restrict the possible translations such that the center of P is required to overlap some voxel center of V (and hence P is allowed only to rotate around each voxel center of V, the "center–to–center" assumption), the number of possible M is still $\mathcal{O}(|V||P|^{3\frac{2}{3}})$. This gives $\mathcal{O}(|V||P|^{4})$ pattern matching algorithm to compute everywhere an upper bound for distance (1). To get practical algorithms we consider sparser subsets of matching functions M. In this way we obtain an $\mathcal{O}(|V||P|^{2})$ method to get upper bounds. This is further generalized to yield lower bounds, too.

For the κ–threshold problem we modify the above algorithms such that their expected running time depends on κ. This gives an $\mathcal{O}(|V|\kappa^{4})$ expected time algorithm to obtain upper bounds and an $\mathcal{O}(|V|\kappa^{2})$ algorithm to obtain (somewhat looser) upper bounds and lower bounds.

The so–called filtration is in practice a very useful approach to solving the κ–threshold problem. The method works in two phases. In the first phase (filtration) one computes with a fast algorithm lower bound values for the distance in the entire search space of matching functions. In the second phase (checking) the areas of the search space that got a lower bound $\leq \kappa$ are examined using a more accurate distance evaluation method. The hopefully large part of the search space that got lower bound $> \kappa$ can be skipped in the second phase.

The above $\mathcal{O}(|V|\kappa^{2})$ algorithm for lower bounds can be used as a filter for the $\mathcal{O}(|V|\kappa^{4})$ algorithm. Furthermore, we develop another filter algorithm that is based on comparing the histograms of the voxel values of a spherical subset of P and its image under M. This eliminates rotations because the histogram of the sphere is rotation invariant. We obtain very fast filter algorithm which is also very effective when κ is relatively small. The running time of this filter is $\mathcal{O}(|V|\kappa^{2/3})$ for the Hamming distance. The method can be generalized for other distance functions also.

We have implemented the $\mathcal{O}(|V|\kappa^{2})$ and $\mathcal{O}(|V|\kappa^{2/3})$ filters and report some results of their running time at the end of the paper. The results show that filtration is extremely efficient for small κ but looses its power when κ grows.

Significance and comparison. Our analysis of pattern matching in digital volumes is based on the novel approach in which the pattern matching is defined directly on the voxel level. The traditional approach in the field typically defines pattern matching using concepts from continuous mathematics and uses fast Fourier transform to implement translations (as a convolution) and polar transformations (or sampling on an evenly spaced grid) to implement rotations; see e.g. [2]. These methods are inherently approximate and hence introduce another layer of inaccuracy on top of the noise in the data that can be very high in typical applications. Therefore we believe that our exact combinatorial analysis and methods can be useful. We are not aware of earlier similar results (expect for our own [6, 7, 5] that deals with image processing). For example, our analysis of the number and the structure of the relevant rotations gives a basis for comparing different heuristics that use a sparse set of rotations. Our filtration algorithms are very fast and as such of practical value. Also our slow algorithms become more interesting with increasing computing power.

2. Definitions

Let the volume $V = V[1..n, 1..n, 1..n]$ and the pattern $P = P[1..m, 1..m, 1..m]$ be three dimensional arrays of *point samples*. Each sample has a *color* in a finite ordered *alphabet* Σ. The size $|\Sigma|$ of Σ is denoted by σ. The arrays P and V can be thought as point samples of colors taken from a regular cubic grid of sample points of some "natural" volumes in the real space $\mathbf{R}^3$. For simplicity we restrict the consideration on cubic arrays V and P only.

There are several possibilities to define a mapping from P to V, that is how to compare the colors of P to colors of V. Our approach to the problem is combinatorial. We will compare each color sample of P against the color of the closest sample of V. The distance between the samples is simply the Euclidean distance. This is also technically convenient. The Voronoi diagram for the samples is a regular array of unit cubes in $\mathbf{R}^3$, which we call *voxels*.

To define a possibly rotated approximate occurrence of P in V we use a geometric interpretation of P and V. In the real space $\mathbf{R}^3$, the eight corners of the voxel $V[i, j, k]$ are $(i-1, j-1, k-1), (i, j-1, k-1), (i-1, j, k-1), (i, j, k-1)$, $(i-1, j-1, k), (i, j-1, k), (i-1, j, k)$, and (i, j, k). Hence the voxels for V form a regular $n \times n \times n$ array that in $\mathbf{R}^3$ covers the space between $(0, 0, 0), (n, 0, 0), (0, n, 0)$, $(n, n, 0), (0, 0, n), (n, 0, n), (0, n, n)$ and (n, n, n). Each voxel has a *center* which is the geometric center point of the voxel, i.e., the center of the voxel for $V[i, j, k]$ is $(i - \frac{1}{2}, j - \frac{1}{2}, k - \frac{1}{2}) \in \mathbf{R}^3$. The array of voxels for pattern P is defined similarly.

The *center* of the whole pattern P is the center of the voxel in the middle of P. Precisely, assuming for simplicity that m is odd, the center of P is the center of voxel $P[\frac{m+1}{2}, \frac{m+1}{2}, \frac{m+1}{2}]$, that is, point $(\frac{m}{2}, \frac{m}{2}, \frac{m}{2}) \in \mathbf{R}^3$.

Assume now that P has been moved inside of V using a rigid motion (translation and rotation). The location of P with respect to V can be uniquely given as $((u, v, w), (\alpha, \beta, \gamma))$, where (u, v, w) is the location of the center of P in V, and (α, β, γ) is the rotation of P in respect of the z, y and x axes of V (and applied in this order). Now P is said to be at *location* $((u, v, w), (\alpha, \beta, \gamma))$ inside of V.

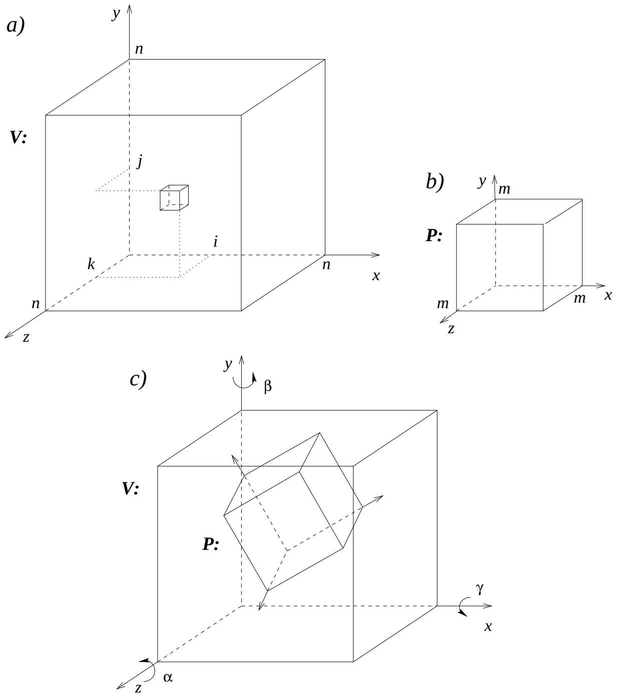

Figure 2. a) An $n \times n \times n$ volume V with voxel $V[i, j, k]$ shown. b) An $m \times m \times m$ pattern P. c) A possible location of P in V.

The distance between V and P located at $((u, v, w), (\alpha, \beta, \gamma))$ is defined by comparing the colors of the voxels of V and P that overlap. Because of the translation and rotation, the voxels do not necessarily overlap each other exactly; a voxel of V can intersect several voxels of P. Hence, there are several possibilities to define a match between the overlapping voxels. We will use the centers of the voxels of P for selecting the comparison points.

More precisely, let us define the *matching function* M when P is at location $((u, v, w), (\alpha, \beta, \gamma))$ as follows.

Definition 1 *For each voxel $P[r, s, t]$ of P, let $V[r', s', t']$ be the voxel of V such that the center of $P[r, s, t]$ belongs to the area covered by $V[r', s', t']$. Then $M(P[r, s, t]) = V[r', s', t']$.*

Hence M is a function from the voxels of P to the voxels of V. We may assume that M is uniquely defined; otherwise adjust (α, β, γ) "infinitesimally" such that no center of P hits the voxel boundaries of V.

We use distance function $d(p)$ to compute the distance between voxels p and $M(p)$. Our algorithms take the distance function as a "black box", and assume that it can be computed in a unit time per voxel. However, one of our filtering algorithms (see Sec. 5) makes further assumptions of the distance function to allow fast computation.

Definition 2 *The distance D between V and P located at position $((u,v,w),(\alpha,\beta,\gamma))$ in V is $D(P,V,((u,v,w),(\alpha,\beta,\gamma))) = \sum_{r,s,t} d(P[r,s,t], M(P[r,s,t])) = \sum_{r,s,t} d(P[r,s,t])$.*

Our problem is to evaluate $D(P,V,((u,v,w),(\alpha,\beta,\gamma)))$ when the location $((u,v,w),(\alpha,\beta,\gamma))$ varies over V. In particular, we want to find $((u,v,w),(\alpha,\beta,\gamma))$ such that distance $D(P,V,((u,v,w),(\alpha,\beta,\gamma)))$ is the smallest possible. We have also a threshold versions of our algorithms, that is we require that $D \leq \kappa$, for some κ. In Definition 2, D was given in terms of $d(p) = d(d, M(p))$. In typical applications function $d(p)$ can be, e.g.,

1. $d(p) = 0$, if $\mathit{color}(p) = \mathit{color}(M(p))$, and 1 otherwise (the Hamming distance),
2. $d(p) = |\mathit{color}(p) - \mathit{color}(M(p))|$,
3. $d(p) = (\mathit{color}(p) - \mathit{color}(M(p)))^2$.

Our algorithms will be based on the structure of the matching functions M when $((u,v,w),(\alpha,\beta,\gamma))$ changes. Therefore we first analyze M in the next section.

3. Counting the matching functions

We make first the so–called *center–to–center assumption*, which considerably simplifies the situation. We assume that the center of P coincides the center of some voxel of V, say voxel $V[i,j,k]$. Hence location $((u,v,w),(\alpha,\beta,\gamma))$ equals $((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}),(\alpha,\beta,\gamma))$.

Consider what happens to M when angle α grows continuously, starting from $\alpha = 0$. Function M changes only at the values of α such that some voxel center of P hits some voxel boundary of V. The same happens for growing β, for each α, and for growing γ, for each α and β.

Some elementary analysis shows that the set of rotation angles $\mathcal{A}$, at which M changes is obtained as follows; here $i', j', k', h = -\lfloor m/2 \rfloor, \ldots, \lfloor m/2 \rfloor$, such that the formulas are defined, and $R_z(\alpha)$ is the rotation matrix [9] in respect of z–axis by angle α:

$$\begin{aligned} A_\alpha &= \{\arcsin \frac{h+\frac{1}{2}}{\sqrt{i^2+j^2}} - \arcsin \frac{j}{\sqrt{i^2+j^2}}, \\ &\quad \arccos \frac{h+\frac{1}{2}}{\sqrt{i^2+j^2}} - \arcsin \frac{j}{\sqrt{i^2+j^2}} \mid \\ &\quad (i,j,k)^T = (i',j',k')^T \}. \end{aligned}$$

$$\begin{aligned} A_\beta(\alpha) &= \{\arcsin \frac{h+\frac{1}{2}}{\sqrt{i^2+k^2}} - \arcsin \frac{k}{\sqrt{i^2+k^2}}, \\ &\quad \arccos \frac{h+i-\lfloor i \rfloor}{\sqrt{i^2+k^2}} - \arcsin \frac{k}{\sqrt{i^2+k^2}} \mid \\ &\quad (i,j,k)^T = R_z(\alpha)(i',j',k')^T \}. \\ A_\gamma(\alpha,\beta) &= \{\arcsin \frac{h+j-\lfloor j \rfloor}{\sqrt{j^2+k^2}} - \arcsin \frac{j}{\sqrt{j^2+k^2}}, \\ &\quad \arccos \frac{h+k-\lfloor k \rfloor}{\sqrt{j^2+k^2}} - \arcsin \frac{j}{\sqrt{j^2+k^2}} \mid \\ &\quad (i,j,k)^T = R_z(\alpha)R_y(\beta)(i',j',k')^T \}. \end{aligned}$$

$$\mathcal{A} = \{(a,b,c) \mid a \in A_\alpha;\ b \in A_\beta(a);\ c \in A_\gamma(a,b)\}.$$

The size of $\mathcal{A}$, that is, the number of orientations of P with different M is $\mathcal{O}(m^4 m^4 m^4) = \mathcal{O}(m^{12}) = \mathcal{O}(|P|^4)$. Notice, however, that A_α is a multi–set, and contains only $\mathcal{O}(m^3)$ different angles, so there are actually only $\mathcal{O}(m^{11})$ different orientations for P. We have obtained the following.

Theorem 1 *Assuming the center–to–center translation, there are $\mathcal{O}(|P|^{3\frac{2}{3}})$ different matching functions for each fixed voxel $V[i,j,k]$ of V.* □

For the sequel, we assume that the angles in $\mathcal{A}$ are listed in the lexicographic order. Calculating and sorting the angles takes $\mathcal{O}(m^{11} \log m)$ time. Each angle in $\mathcal{A}$ is also associated with the information of which voxels of P hit the boundaries of V for that particular angle. This information can be obtained in $\mathcal{O}(|P|^4)$ time.

We now consider removing the center–to–center restriction. We use a certain heuristic to approximately achieve this. Let the set $\mathcal{M}_{27}(p)$ denote the voxel $M(p)$ and its 26 neighboring voxels of V. Define $\delta(P[r,s,t]) = \min\{d(P[r,s,t], V[r',s',t']) \mid V[r',s',t'] \in \mathcal{M}_{27}(P[r,s,t])\}$. Assume that P is at location $((u,v,w),(\alpha,\beta,\gamma))$ such that (u,v,w) is inside the voxel $V[i,j,k]$. Now P must have at location $((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}),(\alpha,\beta,\gamma))$ at least as good an occurrence as at $((u,v,w),(\alpha,\beta,\gamma))$, in the sense of the following theorem.

Theorem 2 *Let M be the matching function at location $((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}),(\alpha,\beta,\gamma))$. Then the generalized distance*

$$\sum_{r,s,t} \delta(P[r,s,t]) \leq D(P,V,((u,v,w),(\alpha,\beta,\gamma))).$$

Proof. Translate P from $((u,v,w),(\alpha,\beta,\gamma))$ to $((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}),(\alpha,\beta,\gamma))$, that is, center–to–center condition becomes true, while preserving the rotation angle. This translation may move any center of P at most distance

$\sqrt{3}/2$. However, in order to be translated outside the area covered by $M(P[r,s,t])$ and its 26 neighboring voxels, any center of P must be translated at least distance 1. Hence $\delta(P[r,s,t])$ before the translation is $\leq d(P[r,s,t])$ after the translation. This means that $\sum_{r,s,t} \delta(P[r,s,t])$ must be $\leq D(P,V,((u,v,w),(\alpha,\beta,\gamma)))$. □

The distance $\sum_{r,s,t} \delta(P[r,s,t])$ can be evaluated using rotation $(\alpha_i,\beta_j,\gamma_k)$ in $\mathcal{A}$, instead of (α,β,γ), such that $\alpha_{i-1} \leq \alpha \leq \alpha_{i+1}$, $\beta_{j-1} \leq \beta \leq \beta_{j+1}$, and $\gamma_{k-1} \leq \gamma \leq \gamma_{k+1}$, because the matching M for those two rotations is the same, by our construction.

Theorem 3 *Without the center–to–center assumption, there are $\mathcal{O}(|P|^{5\frac{1}{3}})$ different matching functions for each fixed voxel $V[i,j,k]$ of V.*

Proof. (Sketch.) Assume that P is at location $((u,v,w),(\alpha,\beta,\gamma))$. Translate P in the direction of x–axis, such that some center of P hits some voxel boundary of V. There are $\mathcal{O}(m^3)$ voxel centers, that can hit $\mathcal{O}(m)$ voxel boundaries, which gives total $\mathcal{O}(m^4)$ different translations. Each such translation brings some center of, say voxel p_0, to coincide some voxel boundary of V. Now for each translation, fix the voxel p_0, and begin another translation. Rotate P about the y axis, around the center of p_0, such that the center of another voxel p_1 hits some voxel boundary of V. There are again $\mathcal{O}(m^4)$ such rotations. Fix also p_1, and rotate P again, now around the line defined by p_0 and p_1, until the center of a voxel p_2 hits a voxel boundary. This gives again $\mathcal{O}(m^4)$ possible rotations. Now allow p_0, p_1, and p_2 move along the voxel boundaries they touch, until the center of a voxel p_3 hits a voxel boundary of V, giving additional $\mathcal{O}(m^4)$ translations. Now the location of P is fixed, if we fix the voxel boundaries that p_0, p_1, p_2, and p_3 touch. That is, P cannot be translated in any direction without disconnecting p_0, p_1, p_2, and p_3 from their corresponding boundaries of V. We have $\mathcal{O}(m^4)$ different choices for each of the four voxels, hence the total number of possible translations and rotations is $\mathcal{O}(m^{16}) = \mathcal{O}(|P|^{5\frac{1}{3}})$. □

4. Evaluating the distance

For finding the best occurrence of P in V, it suffices to evaluate the distance D using the matching functions for each $V[i,j,k]$ (that is, the center of P is located somewhere in voxel $V[i,j,k]$ of V). Precisely, for each voxel $V[i,j,k]$ of V, let

$$\begin{aligned} D'(i,j,k) &= \min\{\, D((u,v,w),(\alpha,\beta,\gamma)) \mid \\ &\quad (u,v,w) \in V[i,j,k], 0 \leq \alpha,\beta,\gamma \leq 2\pi \,\} \end{aligned}$$

and

$$\begin{aligned} D'_c(i,j,k) &= \min\{\, D((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}),(\alpha,\beta,\gamma)) \\ &\quad \mid 0 \leq \alpha,\beta,\gamma \leq 2\pi \,\}. \end{aligned}$$

Obviously D'_c is an upper bound for D': $D'_c(i,j,k) \geq D'(i,j,k)$ for all $1 \leq i,j,k \leq n$.

With the center–to–center assumption, Theorem 1 suggests the following algorithm for evaluating $D'_c(i,j,k)$: Evaluate $D(((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}),(\alpha,\beta,\gamma)))$ for each $(\alpha,\beta,\gamma) \in \mathcal{A}$. This would take $\mathcal{O}(|P|^5)$ time because there are $\mathcal{O}(|P|^4)$ angles (α,β,γ), and evaluating the distance for each (α,β,γ) takes $\mathcal{O}(|P|)$ time with the trivial algorithm. This is unnecessarily slow; using incremental evaluation of the distance we obtain $\mathcal{O}(|P|^4)$ algorithm as follows.

When preprocessing P to obtain the angles $\mathcal{A} = ((\alpha,\beta,\gamma)_1, (\alpha,\beta,\gamma)_2, \ldots, (\alpha,\beta,\gamma)_K)$, we also associate with each $(\alpha,\beta,\gamma)_s$ the set $\mathcal{C}_s$ containing the corresponding voxel centers that must hit a voxel boundary at $(\alpha,\beta,\gamma)_s$. Hence we can evaluate $D((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}),(\alpha,\beta,\gamma)_{s+1})$ incrementally from $D((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}),(\alpha,\beta,\gamma)_s)$ just by re–evaluating the voxel distances restricted to set $\mathcal{C}_s$. This takes time $\mathcal{O}(|\mathcal{C}_s|)$. Hence the total time of finding the smallest distance at $V[i,j,k]$ when $(\alpha,\beta,\gamma) = (\alpha,\beta,\gamma)_0, \ldots (\alpha,\beta,\gamma)_K$ is $\mathcal{O}(\sum_s |\mathcal{C}_s|)$, which is $\mathcal{O}(|P|^4)$.

Repeating the above method for each voxel of V we obtain the following result.

Theorem 4 *The distance $D'_c(i,j,k)$ for all voxels $V[i,j,k]$ of V can be evaluated in time $\mathcal{O}(|V||P|^4)$.* □

4.1. The threshold version of the problem

In applications one is often interested in finding approximate occurrences that are good enough. Given a parameter κ, one wants to find voxels $V[i,j,k]$ such that $D'(i,j,k)$ or $D'_c(i,j,k)$ is $\leq \kappa$. For the Hamming distance, this is sometimes called the *κ–mismatches problem* in the string matching literature.

We consider first the Hamming distance. We modify the algorithms presented in Section 4 such that its expected running time $\mathcal{O}(|V|\kappa^4)$ for the center–to–center version of the problem. We use the uniform Bernoulli model. Then each voxel of V and P gets its color from Σ such that each color occurs with the same probability, independently of the other voxels. Consider counting the mismatches between P and V at $((u,v,w),(\alpha,\beta,\gamma))$. The number of mismatches has binomial distribution, with success probability $(|\Sigma|-1)/|\Sigma|$. The expected number of mismatches in q tests is $q\frac{|\Sigma|-1}{|\Sigma|}$. Requiring that $q\frac{|\Sigma|-1}{|\Sigma|} \approx \kappa$ gives that about $q = \kappa\frac{|\Sigma|}{|\Sigma|-1}$ tests should be enough in typical cases to find out that the distance must be $\geq \kappa$.

This suggests an improved algorithm for the threshold case. Instead of using the whole P, select the smallest sub-pattern P' of P, with the same center voxel, of size $m' \times m' \times m'$ such that $m' \times m' \times m' \geq \kappa\frac{|\Sigma|}{|\Sigma|-1}$. Use the algorithm of Theorem 4 to find if $D(P',V,((u,v,w),(\alpha,\beta,\gamma))) \leq \kappa$. If it is, then check with gradually growing sub-patterns P'' whether or not $D(P'', V, ((u,v,w),(\alpha,\beta,\gamma))) \leq \kappa$, until $P'' = P$. If not, continue with P' at the next location. The expected running time of the algorithm is $\mathcal{O}(|V||P'|^4)$ which is $\mathcal{O}(|V|\kappa^4)$.

Theorem 5 *The κ–threshold problem can be solved in expected time $\mathcal{O}(|V|\kappa^4)$ for the center–to-center Hamming distance.* □

For the other distances similar algorithms can be derived. Consider e.g. the distance $d(p) = |color(p) - color(M(p))|$. In the Bernoulli model, the expected value for $d(p)$ is $|\Sigma|/3$. This gives that $q \approx 3\kappa/|\Sigma|$ voxels are needed to find out that $D > \kappa$.

5. Rotation invariant filtration

In this section, we give filtering algorithms that can find the upper and lower bounds for the distance fast. We scan V using fast rotation invariant filtering algorithm to find candidate positions that may have an occurrence. These candidate positions are then verified using the algorithm of Theorem 5. We take two approaches. In Sec. 5.1 we use a reduced set of matching functions of size $\mathcal{O}(|P|)$, and calculate upper and lower bounds for D in time $\mathcal{O}(|P|^2)$ and in time $\mathcal{O}(\kappa^2)$ for the threshold problem. In Sec. 5.3 we reduce the number of rotations to 1, by using color histograms. This gives fast $\mathcal{O}(\kappa^{2/3})$ time algorithm.

5.1. Reducing the number of the matching functions

With or without the center–to–center assumption, the number of matching functions can be reduced to just $\mathcal{O}(|P|)$ with the heuristic we used in Theorem 2, if the set of angles is chosen properly. The trick is to choose such angles that the voxel that is farthest away from the center of rotation moves at most distance 1 (or $\frac{1}{2}$) at each step, which guarantees that the center of that voxel cannot move outside of its 26 voxel neighborhood. It then follows that no center of any other voxel of P moves outside its corresponding 26 voxel neighborhood.

The distance between the rotation center and the voxel $p = P[m,m,m]$ farthest away from it is $r = \sqrt{3}(m+1)/2$. Assume that P is at location $((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}), (\alpha,\beta,\gamma))$. We are interested in a (large) angle $\hat{\alpha}$ such that if P is rotated to location $((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}), \alpha+\hat{\alpha}, \beta, \gamma)$, then $M(P[m,m,m])$ and its 26 neighbors still cover the center of $P[m,m,m]$. One such angle is

$$\hat{\alpha} = 2 \arcsin \frac{1}{2r} > \frac{1}{r}$$

If p cannot rotate out of its 26 voxel neighborhood, then clearly no other voxel of P can. If the center–to–center assumption is eliminated, then p is allowed to move only distance $\frac{1}{2}$ for angle $\hat{\alpha}$. This gives $\hat{\alpha} = \frac{1}{2r}$.

The new set of angles $\hat{\mathcal{A}}$ for the orientations of P is

$$\hat{\mathcal{A}} = \{(i\hat{\alpha}, j\hat{\alpha}, k\hat{\alpha}) \mid 0 \leq i\hat{\alpha}, j\hat{\alpha}, k\hat{\alpha} < 2\pi; i,j,k \in \mathbf{N}\}$$

The size of $\hat{\mathcal{A}}$ is approximately $(2\pi/\hat{\alpha})^3$, which is $\mathcal{O}(|P|)$.

Theorem 6 *Assume that P is at location $((u,v,w),(\alpha,\beta,\gamma))$. Let M be the matching function at location $((u,v,w),(i\hat{\alpha}, j\hat{\alpha}, k\hat{\alpha}))$, where $\alpha-\hat{\alpha} < i\hat{\alpha} < \alpha+\hat{\alpha}$, $\beta-\hat{\alpha} < j\hat{\alpha} < \beta+\hat{\alpha}$, and $\gamma-\hat{\alpha} < k\hat{\alpha} < \gamma+\hat{\alpha}$. Then the generalized distance*

$$\sum_{r,s,t} \delta(P[r,s,t])$$

for M is $\leq D(P,V,((u,v,w),(\alpha,\beta,\gamma)))$. □

As there are $\mathcal{O}(|P|)$ angles and the distance evaluation takes at most $\mathcal{O}(|P|)$ time for each angle, the lower bound of Theorem 6 for D can be evaluated in time $\mathcal{O}(|P|^2)$.

Theorem 7 *The lower bound distance of $D'(i,j,k)$ for all voxels $V[i,j,k]$ of V can be evaluated in time $\mathcal{O}(|V||P|^2)$ and in expected time $\mathcal{O}(|V|\kappa^2)$ for the threshold problem.* □

Note that using $\hat{\mathcal{A}}$ and the distance function d in the evaluation of the distance gives an upper bound of $D'(i,j,k)$ (as well as $D'_c(i,j,k)$).

5.2. Improving the lower bound

Theorems 2 and 6 suggest a matching method that ignores some of the matching functions, but looks also the neighbors of $M(P[r,s,t])$. This heuristic obviously results in lower bound of the true distance. Because of the nature of the heuristic, a uniformly colored $3 \times 3 \times 3$ pattern may be successfully matched against a single voxel of V. This is not possible without the heuristic. The matching function is a many–to–one mapping, but at most four (neighboring) voxels of P may map to the same voxel of V. This information can be utilized as follows. Assume that P is at location $((i-\frac{1}{2}, j-\frac{1}{2}, k-\frac{1}{2}), (\alpha,\beta,\gamma))$. Associate with each voxel $V[i,j,k]$ a counter $counter(V[i,j,k])$, initialized to zero, that tells how many voxels of P are successfully matched against it. Now consider voxel $p = P[r,s,t]$. If p matches with only one of the voxels defined by $M(p)$, and the corresponding counter is less than four, then increase the counter by one for a successful match. If the counter is already four, then p mismatches. If there are several possible matches for p, then do not increase any counter, as the match is not uniquely defined.

Heuristic 1 *Match the voxel p according to one of the cases:*

1. *$color(p) \notin \{\ color(v) \mid v \in \mathcal{M}_{27}(p)\ \} \Rightarrow p$ mismatches.*

2. *$color(p) \in \{\ color(v) \mid v \in \mathcal{M}_{27}(p), counter(v) < 4\ \} \Rightarrow p$ matches. If v is uniquely defined, then increase $counter(v)$ by one.*

Assume that the voxels of V are divided to eight small equally sized cubes, such that they all have one common corner, that is the center of the voxel. Calculate now the set of angles $\mathcal{A}$, using this finer grid for V (the grid for P is not altered). This means that the size of $\mathcal{A}$ will grow eight times larger. The matching function now maps the voxel centers of P to *octants* of V. Now it is possible to use $3 \times 3 \times 3$ neighborhood of octants of $M(p)$ to compute the lower bounds. These 27 octants cover only 8 voxels of V. This observation gives a method to compute better lower bound than Theorem 2 suggests. However, the method is more complex, as the eight voxel neighborhood must be chosen according to which octant $M(p)$ belongs to.

The method is very similar to that of Theorem 2. For a proof why this works, assume that P is at location $((u,v,w),(\alpha,\beta,\gamma))$. The center of $p = P[r,s,t]$ is now inside of the octant $M(p)$. If the center of P is allowed to move to $(i - \frac{1}{2}, j - \frac{1}{2}, k - \frac{1}{2})$, then the center of p may be translated outside of the corresponding octant. Because the center of P, and hence the center of p may move only a distance $\frac{1}{2}$ along any coordinate axis direction, the center of p must remain inside its original octant, or its 26 neighboring octants. These octants are covered by 8 original voxels of V, and one of those covers also $M(p)$.

5.3. Histogram filtering

For κ small enough, the efficiency of our algorithm can still be improved by *histogram filtering*. The filter is based on the ideas of [11] and [8]. Consider each voxel p of P such that the distance from the center of p to the center of P is at most $R \leq (m-1)/2$ (assuming odd m). These voxels fall inside a discrete sphere whose radius is R. Now calculate a *histogram* of the color distribution of those voxels. This histogram is (almost) rotation invariant, and together with corresponding histograms from V it can effectively filter unpromising positions of V. The histogram $\mathcal{H}$ has one bin for each color in Σ. Let $color(v)$ denote the color of pixel v. The histogram for position (i,j,k) of V is defined as follows.

Definition 3 $\mathcal{H}_{V[i,j,k]}(c) = \sum_{i',j',k'}\{1 \mid color(V[i+i', j+j', k+k']) = c;\ \sqrt{i'^2 + j'^2 + k'^2} \leq R\}$.

Note that it is easy to obtain $\mathcal{H}_{V[i+1,j,k]}$ from $\mathcal{H}_{V[i,j,k]}$ incrementally in time $\mathcal{O}(R^2)$. To see this, note that when the $\mathcal{H}_V$ is translated by one voxel along a coordinate axis, about πR^2 new voxels of V will come inside the sphere, and exactly the same number of voxels of V will go out the sphere.

We need also a histogram $\mathcal{H}_P$ for P. This is slightly more complicated to obtain than $\mathcal{H}_V$. Assume that P is at location $((u,v,w),(\alpha,\beta,\gamma))$ inside of V. Consider the matching function at that location. It may happen for e.g. voxels $p_0 = P[r,s,t]$, $p_1 = P[r+1,s,t]$, $p_2 = P[r,s+1,t]$ and $p_4 = P[r+1,s+1,t]$ that $M(p_0) = M(p_1) = M(p_2) = M(p_3)$. In this case, if any of the colors of p_0, p_1, p_2, and p_3 are different, then there cannot be an exact match at location $((u,v,w),(\alpha,\beta,\gamma))$, because the color $M(p_0)$ cannot agree with all the colors of p_0, p_1, p_2, and p_3. This means that κ must be greater than zero for an approximate match. On the other hand, if the colors of p_0, p_1, p_2, p_3, and $M(p_0)$ are the same, then a single voxel of V is counted to match with four voxels of P. This must be taken into account in computation of $\mathcal{H}_P$.

In order to compute the histogram for P, assume that P is (in some orientation) inside a *reference grid of voxels*, called W. The grid W is used to determine at which orientation, and which voxels of P can map to the same voxels of V. That is, there are some voxels (inside a sphere of radius R) in W (and hence in V) that may cover more than one center of voxels of P, for some orientations of P. The contents (voxel colors) of W is irrelevant here.

Let P be at location $((u,v,w),(\alpha,\beta,\gamma))$ in W, such that (u,v,w) belongs to voxel $W[i,j,k]$, and $M(P[r,s,t]) = W[r',s',t']$ for each voxel $P[r,s,t]$. Let $g_{r',s',t'}$ denote the number of voxels of P of color c, that map to the voxel $W[r',s',t']$ when P is in location $((u,v,w),(\alpha,\beta,\gamma))$. Now the histogram bin $\mathcal{H}_{P(\alpha,\beta,\gamma)}(c)$ for color c of P for orientation (α,β,γ) is

$$\mathcal{H}_{P(\alpha,\beta,\gamma)}(c) = \sum_{r,s,t}\{1/g_{r',s',t'} \mid color(P[r,s,t]) = c; \\ \sqrt{(i-r')^2 + (j-s')^2 + (k-t')^2} \leq R\}.$$

As the matching function is many–to–one mapping, the number of colors in each bin of $\mathcal{H}_P$ may change as P rotates. Therefore, as the correct angle of rotation is not known in the filtering phase, and hence the exact number of colors in each bin is unknown, we need to use "minimum–histogram" for P:

Definition 4 $\mathcal{H}_P(c) = \min(\ \mathcal{H}_{P(\alpha,\beta,\gamma)}(c)\ )$.

Our filter compares $\mathcal{H}_{V[i,j,k]}$ and $\mathcal{H}_P$ at each position (i,j,k) of V. Consider the Hamming distance. $\mathcal{H}_P$ tells for each color the minimum number of voxels of that color that is at least required for a complete match. The difference $Z(i,j,k)$ between $\mathcal{H}_{V[i,j,k]}$ and $\mathcal{H}_P$ tells the lower bound of the number of mismatches at position (i,j,k) of V. This is computed as the number of colors missing from $\mathcal{H}_P$:

$$Z(i,j,k) = \sum_{c \in \Sigma} \max(\, 0, \mathcal{H}_P(c) - \mathcal{H}_{V[i,j,k]}(c)\,).$$

If $Z \leq \kappa$, then there may be an approximate occurrence of P in V with at most κ mismatches. Note that Z can be calculated incrementally also, in the same way as histograms $\mathcal{H}_{V[i,j,k]}$, in time $\mathcal{O}(R^2)$.

To get a good filtration capacity, the radius R must be chosen such that the expected $Z(i,j,k)$ is greater than κ. At minimum, the number of voxels inside the sphere must be greater than κ. This gives approximately $\frac{4}{3}\pi R^3 > \kappa \Rightarrow R > \sqrt[3]{\frac{3\kappa}{4\pi}}$. If $R = \mathcal{O}(\kappa^{1/3})$, the filtering takes time $\mathcal{O}(|V|\kappa^{2/3})$. Exact analysis of the expected value of Z is very difficult. However, the (somewhat pessimistic) analysis of the "counting filter" [10] could be adapted for our filter.

Theorem 8 *For small κ the κ–threshold Hamming distance problem can be solved in expected time $\mathcal{O}(|V|\kappa^{2/3})$.* □

It is possible to generalize the histogram filter for many other distance functions, see [4].

6. Experimental Results (Preliminary)

The $\mathcal{O}(|V|\kappa^2)$ and $\mathcal{O}(|V|\kappa^{2/3})$ expected time versions of the algorithm were implemented for the Hamming distance. The algorithms were implemented in C–programming language, compiled by gcc version 2.7.2.3 running on 450MHz PentiumII with Linux operating system. The reference histogram $\mathcal{H}_P$ was taken from a single orientation (that is, $(\alpha, \beta, \gamma) = (0,0,0)$), and histograms $\mathcal{H}_V$ were computed using the 26 voxel neighborhood for each voxel, allowing each voxel in the neighborhood contribute to $\mathcal{H}_V$. This heuristic allows fast preprocessing of the pattern, and in practice works very well, the filtering capability being about 99% for all our experiments.

The test data was a reconstruction of *sus1* mutant of the *PRD1–bacteriophage* [1]. The size of the density map was 119 × 119 × 119 voxels, and the size of the alphabet is 256. The patterns were extracted from the virus shell in random positions and orientations, and then searched from the reconstruction.

Table 1 shows running times for the Hamming distance. The method first uses the incremental histogram filter for κ. If this filter fails for some location of V, then the maximum sized histogram is extracted from V in that location. If that fails to filter, the method of Theorem 7 is used.

κ	0	1	2	4	8	16
total time	0.54	17	19	61	165	3574
filtering time	0.13	16	16	42	42	42
succ / fail	670298	121286	16445	2287	471	57

Table 1. Experimental results for the Hamming distance, for $|P| = 5 \times 5 \times 5$ for different κ. Times are given in seconds. Succ / fail tells the ratio of the number of successful filtrations versus the number of failures (if the histogram filter indicates that there cannot be a match in the current position, then the filtration is said successful, otherwise it is a failure.)

7. Discussion and Conclusions

We have presented fast combinatorial algorithms for computing 3–D Hamming distance under rotations and translations. Ours is the first combinatorial approach for this problem. The basic algorithms also work as given for many other distance functions. The histogram filtering algorithm can be generalized for other distance functions, and it can be made faster, see [4].

References

[1] S. J. Butcher, D. H. Bamford, and S. D. Fuller. DNA packaging orders the membrane of bacteriophage PRD1. *EMBO Journal*, 14:6078–6086, 1995.

[2] E. D. Castro and C. Morandi. Registration of translated and rotated images using finite fourier transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):700–703, Sept. 1987.

[3] J. Frank. *Three–dimensional electron microscopy of macromolecular assemblies*. Academic Press, 1996.

[4] K. Fredriksson. Rotation invariant histogram filters for multidimensional similarity and distance measures. In *SPIRE'2000*, 2000. (These proceedings).

[5] K. Fredriksson, G. Navarro, and E. Ukkonen. An index for two dimensional string matching allowing rotations. In *IFIP TCS´2000*, 2000. To appear.

[6] K. Fredriksson and E. Ukkonen. A rotation invariant filter for two–dimensional string matching. In *Proceedings of the 9th Annual Symposium on Combinatorial Pattern Matching*, Lecture Notes in Computer Science, pages 118–125. Springer-Verlag, Berlin, 1998.

[7] K. Fredriksson and E. Ukkonen. Combinatorial methods for approximate image matching under translations and rotations. *Pattern Recognition Letters*, 20(11–13):1249–1258, 1999.

[8] R. Grossi and F. Luccio. Simple and efficient string matching with k mismatches. *Inf. Process. Lett.*, 33(3):113–120, 1989.

[9] D. Hearn and M. P. Baker. *Computer Graphics*. Prentice-Hall, 1994.

[10] G. Navarro. *Approximate Text Searching*. PhD thesis, Department of Computer Science, University of Chile, Dec. 1998. `ftp://ftp.dcc.uchile.cl/pub/users/gnavarro/thesis98.ps.gz`.

[11] M. Werman, S. Peleg, and A. Rosenfeld. A distance metric for multidimensional histograms. Technical Report CAR-TR-90, University of Maryland, Center for Automation Research, 1984.