

Eastern Kentucky University

Encompass

Online Theses and Dissertations

Student Scholarship

January 2019

Efficient Local Comparison Of Images Using Krawtchouk Descriptors

Julian DeVille

Eastern Kentucky University

Follow this and additional works at: <https://encompass.eku.edu/etd>



Part of the [Computer Sciences Commons](#), and the [Partial Differential Equations Commons](#)

Recommended Citation

DeVilLe, Julian, "Efficient Local Comparison Of Images Using Krawtchouk Descriptors" (2019). *Online Theses and Dissertations*. 605.

<https://encompass.eku.edu/etd/605>

This Open Access Thesis is brought to you for free and open access by the Student Scholarship at Encompass. It has been accepted for inclusion in Online Theses and Dissertations by an authorized administrator of Encompass. For more information, please contact Linda.Sizemore@eku.edu.

EFFICIENT LOCAL COMPARISON OF IMAGES USING KRAWTCHOUK
DESCRIPTORS

By

Julian DeVille

Thesis Approved:



Chair, Advisory Committee



Member, Advisory Committee



Member, Advisory Committee



Dean, Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a M.S. degree at Eastern Kentucky University, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made. Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or, in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature

Juan J. Soltero

Date

4/8/2019

EFFICIENT LOCAL COMPARISON OF IMAGES USING KRAWTCHOUK
DESCRIPTORS

BY

JULIAN DEVILLE

Submitted to the Faculty of the Graduate School of
Eastern Kentucky University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

2019

© Copyright by Julian DeVille, 2019
All Rights Reserved.

DEDICATION

This thesis is dedicated to all who have enabled to explore mathematics both in and out of the classroom.

ACKNOWLEDGEMENTS

I would like to thank Dr. Atilla Sit for advising, and guidance on this project, my committee members Dr. Steve Szabo, and Dr. Eugene Styer, as well as Dr. Ka-Wing Wong for their support in this endeavor. Also, I would like to thank Jacob Patrick for hardware assistance, and Dr. Jeffrey Neugebauer for LaTeX assistance.

ABSTRACT

It is known that image comparison can prove cumbersome in both computational complexity and runtime, due to factors such as the rotation, scaling, and translation of the object in question. Due to the locality of Krawtchouk polynomials, relatively few descriptors are necessary to describe a given image, and this can be achieved with minimal memory usage. Using this method, not only can images be described efficiently as a whole, but specific regions of images can be described as well without cropping. Due to this property, queries can be found within a single large image, or collection of large images, which serve as a database for search. Krawtchouk descriptors can also describe collections of patches of 3D objects, which is explored in this paper, as well as a theoretical methodology of describing nD hyperobjects. Test results for an implementation of 3D Krawtchouk descriptors in GNU Octave, as well as statistics regarding effectiveness and runtime, are included, and the code used for testing will be published open source in the near future.

TABLE OF CONTENTS

| CHAPTER | PAGE |
|--|------|
| 1 INTRODUCTION: MOMENT-BASED PATTERN RECOGNITION . . . | 1 |
| 2 TWO-DIMENSIONAL KRAWTCHOUK DESCRIPTORS | 4 |
| 2.1 Krawtchouk Polynomials and Other Definitions | 4 |
| 2.2 Invariance | 9 |
| 2.3 2D Krawtchouk Descriptors | 11 |
| 2.4 Workflow | 11 |
| 2.5 Example | 14 |
| 3 THREE-DIMENSIONAL KRAWTCHOUK DESCRIPTORS | 15 |
| 3.1 3D Krawtchouk Moments | 15 |
| 3.2 3D Geometric Moments | 16 |
| 3.3 3D Krawtchouk Descriptors | 18 |
| 3.4 Workflow | 19 |
| 3.5 Rotation Test | 22 |
| 4 n-DIMENSIONAL KRAWTCHOUK DESCRIPTORS | 24 |
| 4.1 n-Dimensional Krawtchouk Moments | 24 |
| 4.2 nD Geometric Moments | 25 |
| 4.3 nD Krawtchouk Descriptors | 28 |
| BIBLIOGRAPHY | 30 |

LIST OF FIGURES

| FIGURE | | PAGE |
|--------|--|------|
| 1 | The 2D weight function plotted at various points. | 7 |
| 2 | Examples of 2D gray-scale images and their reconstructions using 2D weighted Krawtchouk polynomials for $\hat{S} = 5, 20, 50,$ and $100,$ and different (p_x, p_y) pairs | 8 |
| 3 | Flowchart of workflow for 2D Krawtchouk Descriptor implementation | 12 |
| 4 | A section of a projection image of GroEL protein complexes in vitreous ice captured using Cryo-EM | 13 |
| 5 | An example query of the top view of GroEL and top 15 retrieval results using 2DKD | 14 |
| 6 | Query protein surface (1gco.pdb, left) and three target surfaces obtained from the query protein by rotating it using different rotation matrices | 23 |

LIST OF TABLES

| TABLE | | PAGE |
|-------|--|------|
| 1 | Rotation Test: Points Correctly Identified by Minimum Euclidean Distance of Krawtchouk Descriptors | 23 |
| 2 | Length of Krawtchouk Descriptors in Varying Dimension and Order . | 29 |

1 INTRODUCTION: MOMENT-BASED PATTERN RECOGNITION

Computer vision and pattern recognition are growing fields of research within both mathematics and computer science, with a plethora of applications including astronomy, robotics, automotive engineering, medical research, and even Google's visual search engine, Google Vision. Automation of visual inspection and comparison of objects is invaluable to research in numerous fields, and as technology progresses, will continue to remain in high demand[10].

Among the more popular methodologies for pattern recognition is the use of moments of various polynomials. Although most mathematics used in pattern recognition were devised in the early to mid 20th century, recent advancements in technology have enabled to use of these concepts in computer vision and machine learning. This process can be summarized as the interpretation of an image or object as a discrete function, which is then integrated alongside an orthogonal polynomial, and results in an image descriptor, which is later compared to other image descriptors to assess the similarity of images. It is crucial that the polynomials used are orthogonal so that the process can be reversed, thus rendering the descriptor meaningful. The particular nuances of this process are discussed in chapters 2-4, each pertaining to the dimension of the object being described. This general methodology was developed by Hu[7], although instead of incorporating polynomials, Hu descriptors were composed of Hu invariants, which were simply sums and products of high order geometric moments of the original function.

With this general methodology in mind, there are a number of viable polynomials suited to the task, each with different properties that impact their potential. For instance, Zernike polynomials are orthogonal on the unit disc, and thus can be used for moment-based pattern recognition; however, since they are continuous, they must first be discretized for compatibility with the data at hand, causing a degree of machine

error, and the image or object function must also be mapped to the unit disk or ball in \mathbb{R}^2 or \mathbb{R}^3 respectively, which is both cumbersome and prone to further machine error. Nonetheless, Zernike polynomials have resulted in function pattern recognition software due their rotational invariance by design[14]. Most orthogonal polynomials will have similar caveats, hence their diversity in the literature.

Some other popular polynomials for moment-based pattern recognition include Legendre polynomials, Gaussian-Hermite polynomials, Chebyshev (sometimes spelled Tchebichef, see [17]) polynomials, and Krawtchouk polynomials.

Legendre polynomials, developed in the late 18th century, are orthogonal over the real interval $[-1, 1]$, and have been used quite successfully in moment-based pattern recognition. However, like Zernike polynomials, they are continuous, and are orthogonal in a specific, continuous domain, and thus will need to be discretized prior to use, and the image will need to be translated, resulting in machine error[1]. However, Legendre moments seem proficient in discerning patterns within blurred and distorted images [19].

Gaussian-Hermite polynomials, while still continuous by definition, are minimally impacted by discretization in comparison to other continuous orthogonal polynomials, and perform better on images with noise due to the smoothness of their basis functions [15]. Since they are orthogonal over \mathbb{R} in continuous form once weighted by a Gaussian function, and orthogonal over a square $[0 \leq i, j \leq K - 1]$ when discretized, the image need not be translated, which further reduces potential for machine error. Hermite polynomials were initially developed by Laplace in the 19th century. Also see [16]. It is worth noting that once this weight function is applied, they are no longer polynomials, and are merely functions.

Weighted Krawtchouk polynomials are discrete, local, and orthogonal by design over $\{0, \dots, S\}$ for any finite S , similar to discretized Gaussian-Hermite polynomials, and have been growing in popularity for moment-based pattern recognition. Pio-

neered by Ukrainian mathematician M. P. Krawtchouk (sometimes spelled Kravchuk) in the early 20th century, they emerged in part from the discrete, orthogonal polynomials developed by Chebyshev for military applications, but now have applications in both computer vision and coding theory, alongside Krawtchouk matrices [5][8][11]. Since they are discrete by design, and orthogonal over the previously specified set, machine error is minimized since discretization and continuation translation are unnecessary. However, the primary interest in Krawtchouk polynomials, in this context, is their natural locality once weighted. Locality allows moment-based Krawtchouk descriptors to describe specific regions of an image or object without cropping or recomputing polynomials, which both accelerates computation via precomputing these values, and minimizes the loss of information along edges when cropping. Like Gaussian-Hermite polynomials, they cease to be polynomials once weighted.

Say, for instance, one needs to locate a specific cell formation within an image of a tissue sample- by cropping the image, the cell formation may never be found if it appears along an edge where the image is cropped, and a method is needed which can locate it without the potentially removing the target. Local Krawtchouk descriptors could be computed for the query subimage, and then for a grid of points across the image, and so long as the grid sufficiently covers the image, the query will be found in the main image, since no cropping or manipulation of the main image will take place. This concept has been implemented in [3] and [12].

Although this feature is not unique, Krawtchouk polynomials can also be computed in n dimensions, and thus this methodology can be extended to for regional pattern recognition in n dimensional data, which is explored throughout this thesis. Since 2D and 3D examples exist in literature, these are explored first, with example code for 3D along with testing and results, followed by a theoretical outline of an n D extension of the same methodology.

2 TWO-DIMENSIONAL KRAWTCHOUK DESCRIPTORS

2.1 Krawtchouk Polynomials and Other Definitions

Krawtchouk polynomials are first defined in a single dimension.

Definition 1. Krawtchouk polynomials of degree n are defined as

$$K_n(x; p, S) = \sum_{k=0}^n a_{k,n,p,S} x^k = {}_2F_1(-n, -x; -S, \frac{1}{p}), \quad (2.1)$$

where $x, n = 0, \dots, S$, $S > 0$, $p \in (0, 1)$, and ${}_2F_1$ is the hypergeometric function defined as

$${}_2F_1(a, b; c, z) = \sum_{k=0}^{\infty} \frac{(a)_k (b)_k z^k}{(c)_k k!}, \quad (2.2)$$

and $(a)_k$ is the Pochhammer symbol defined as

$$(a)_k = a(a+1) \dots (a+k-1) = \frac{\Gamma(a+k)}{\Gamma(a)}. \quad (2.3)$$

It is important to note that the domain of K_n is both discrete, and subject to S . In order to get our desired properties of locality and orthogonality, we supplement these with a weight function:

Definition 2. The weight function is given by

$$w(x; p, S) = \binom{S}{x} p^x (1-p)^{S-x}. \quad (2.4)$$

This gives us the desired result

$$\sum_{x=0}^S w(x; p, S) K_n(x; p, S) K_m(x; p, S) = \rho(n; p, S) \delta_{nm}, \quad (2.5)$$

where $n, m = 0, \dots, S$, δ is the Kronecker Delta function, and ρ is the norm. For ρ ,

we have

$$\rho(n; p, S) = (-1)^n \left(\frac{1-p}{p} \right)^n \frac{n!}{(-S)_n} \quad (2.6)$$

We can compute Krawtchouk polynomials using the following recursive formula:

$$\begin{aligned} K_n(x; p, S) &= \frac{Sp - 2np + n - x}{p(S - n)} K_{n-1}(x; p, S) \\ &\quad - \frac{1-p}{p} \frac{n}{S-n} K_{n-2}(x; p, S). \end{aligned} \quad (2.7)$$

Krawtchouk polynomials up to K_2 which are easily computed by hand:

$$K_0(x; p, S) = 1$$

$$K_1(x; p, S) = 1 - \left(\frac{1}{Sp} \right) x$$

and

$$K_2(x; p, S) = 1 - \left(\frac{2}{Sp} + \frac{1}{S(S-1)p^2} \right) x + \left(\frac{1}{S(S-1)p^2} \right) x^2$$

However, as mentioned previously, we need these weighted in order to achieve orthonormality, and thus we create weighted Krawtchouk polynomials using the aforementioned w and ρ functions, and denote them \bar{K}_n :

Definition 3. The weighted Krawtchouk function is defined as

$$\bar{K}_n(x; p, N) = K_n(x; p, N) \sqrt{\frac{w(x; p, N)}{\rho(n; p, N)}}. \quad (2.8)$$

These weighted Krawtchouk polynomials give us the necessary local property needed for regional pattern recognition. The weight function produces a Gaussian smear centered at p , which enables local description and orthogonality, and, like the Krawtchouk polynomials themselves, can be extended into n dimensions as needed

[12]. With \bar{K}_n , we also now have a simpler expression of orthonormality than before:

$$\sum_{x=0}^S \bar{K}_n(x; p, S) \bar{K}_{n'}(x; p, S) = \delta_{nn'}. \quad (2.9)$$

In Section 2.4, these definitions are drastically simplified- due to the precomputation of w , we will fix $p = 0.5$, S will always be the length or width of the square region of 2D image, and due to the later discrete integration of these functions to produce Krawtchouk moments, we are only interested in the coefficients of K_n . With w and S fixed, the computation of K_n is the only symbolic calculation necessary, and all else is easily automated. Specifics on computation are addressed later in Section 2.4.

Let $A = \{0, \dots, S\} \times \{0, \dots, S\}$, and define a function $f : A \rightarrow \mathbb{N} \cup \{0\}$, which we will call our image. In practice, the image to be described is the image of f . This, now in two dimensions, preserves our orthonormality condition like so for $n, m, n', m' \in \{0, \dots, S\}$:

$$\sum_{x=0}^S \sum_{y=0}^S \bar{K}_n(x; p_x, S) \bar{K}_m(y; p_y, S) \cdot \bar{K}_{n'}(x; p_x, S) \bar{K}_{m'}(y; p_y, S) = \delta_{nn'} \delta_{mm'}. \quad (2.10)$$

It is worth noting that the methodology up to this point will work for rectangular images of size $N \times M$ with

$$\sum_{x=0}^N \sum_{y=0}^M \bar{K}_n(x; p_x, N) \bar{K}_m(y; p_y, M) \cdot \bar{K}_{n'}(x; p_x, N) \bar{K}_{m'}(y; p_y, M) = \delta_{nn'} \delta_{mm'}, \quad (2.11)$$

but accuracy of results is suboptimal due to the consequent distortion of w , shown in Fig. 1. Implementation on non-symmetric images is discussed in Section 2.4.

Definition 4. The 2D weight function is given by

$$w(x, y; p_x, p_y) = \sqrt{w(x; p_x, S) w(y; p_y, S)}. \quad (2.12)$$

The 2D weight function, while a circular Gaussian smear at $(0.5, 0.5)$, begins to distort as it is moved near the boundaries of the domain. Methods for circumventing this phenomenon are discussed in Section 2.4.

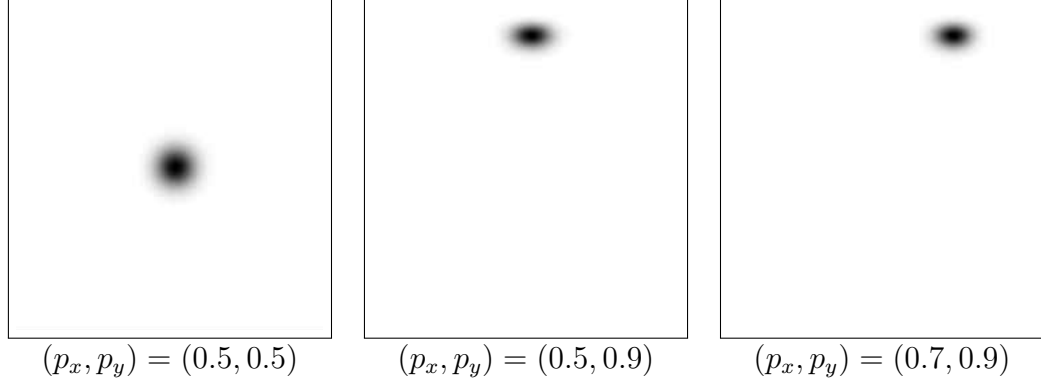


Figure 1: The 2D weight function plotted at various points.

The orthonormality is crucial to this methodology, as it can be reversed like so:

$$\bar{Q}_{nm} = \sum_{x=0}^S \sum_{y=0}^S \bar{K}_n(x; p_x, S) \bar{K}_m(y; p_y, S) f(x, y) \quad (2.13)$$

$$f(x, y) = \sum_{n=0}^S \sum_{m=0}^S \bar{K}_n(x; p_x, S) \bar{K}_m(y; p_y, S) \bar{Q}_{nm} \quad (2.14)$$

and thus one can produce Krawtchouk moments of f through this methodology, as well as reconstruct the original image f , so we can later denote these moments as “descriptors” of the image. We will call the approximate reconstruction $\hat{f}(x, y)$ if insufficient moments are produced for a full reconstruction, which will often be the case.

$$\hat{f}(x, y) = \sum_{n=0}^{\hat{S}} \sum_{m=0}^{\hat{S}} \bar{K}_n(x; p_x, S) \bar{K}_m(y; p_y, S) \bar{Q}_{nm} \quad (2.15)$$

Unlike other similar methods, these weighted Krawtchouk polynomials are local, and thus specific regions can be both described and reconstructed independent of the rest of the domain. Visually, this weight function is a Gaussian smear, as shown in

Fig. 1, but in practice, it darkens all but the region of interest, and produces increasing clarity approaching the precise p_x, p_y values with maximum clarity at precisely p_x, p_y .

Using this methodology, along with Krawtchouk moments of various orders, the classic cameraman.jpg image can be reconstructed like so, as demonstrated in [12]. See Fig. 2.










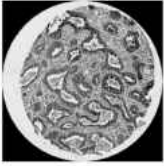
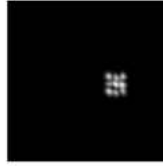
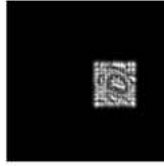
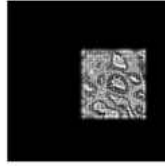
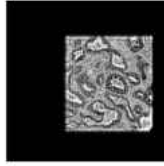
| Original image | (p_x, p_y) | Reconstructions | | | |
|---|----------------|---|---|---|---|
| | | $\hat{S} = 5$ | $\hat{S} = 20$ | $\hat{S} = 50$ | $\hat{S} = 100$ |
|  | $(0.45, 0.75)$ |  |  |  |  |
| | $(0.10, 0.10)$ |  |  |  |  |
|  | $(0.68, 0.48)$ |  |  |  |  |

Figure 2: Examples of 2D gray-scale images and their reconstructions using 2D weighted Krawtchouk polynomials for $\hat{S} = 5, 20, 50,$ and $100,$ and different (p_x, p_y) pairs. The pixel size for the cameraman image (first row) is $300 \times 300.$ The pixel size for the tissue microarray image (second row) is $1024 \times 1024.$ (p_x, p_y) pair here plays the critical role in determining the center of local region-of-interest in an image. (p_x, p_y) is set to $(0.45, 0.75)$ and $(0.68, 0.48)$ for the cameraman and the tissue microarray image, respectively. Image credits – cameraman: Massachusetts Institute of Technology, tissue image: Stanford Tissue Microarray Database [9].

Our intention, however, is not to compress the image into descriptors for later reconstruction, but to produce a minimal set of regional descriptors for an image to compare, and assess the similarity of specific regions of matrices via their corresponding descriptors. Perfect image reconstruction is both unfeasible and unnecessary to this process, so long as the order of moments used sufficient for the intended application.

As demonstrated visually, locality is lost with high orders of moments, rendering

them unfeasible for regional image comparison, as well as computationally inefficient, as the descriptors will consist of more values than the original image. The number of descriptors for a given region is given by the order of Krawtchouk moments used, resulting in a vector of length Υ such that

$$\Upsilon = \frac{(v+1)(v+2)}{2!} \quad (2.16)$$

where v is the order of moments used. More detail on moments is provided in Section 2.2.

2.2 Invariance

The mere comparison of Krawtchouk moments is insufficient for true pattern recognition, as this is simply an efficient pixel by pixel comparison of the images. Queried patterns may not appear in another image precisely as they appear in the query, and hence these weighted Krawtchouk moments must be further modified to account for unforeseen circumstances. For instance, a given pattern may differ in scale, position, rotation, or a combination thereof. Using geometric moments, and the rotation matrix of the region of interest, we produce descriptors which are scale, translation, and rotation invariant as follows.

To begin, we define the weighted 2D function \tilde{f} , then geometric moments and invariants, beginning in a manner similar to that of [7]. Note that $\tilde{\cdot}$ denotes calculations impacted by the weighted f function.

Definition 5. The 2D weighted image function is defined as

$$\tilde{f}(x, y) = w(x, y; p_x, p_y) f(x, y) \quad (2.17)$$

The geometric moments of $\tilde{f}(x, y)$ are defined as follows.

Definition 6. 2D Geometric Moments:

$$\tilde{M}_{ij} = \sum_{x=0}^S \sum_{y=0}^S x^i y^j \tilde{f}(x, y) \quad (2.18)$$

Note that \tilde{M}_{00} is merely the mass of the function, and thus of the range of f from the previous Section 2.1, and so we can use this to create central moments which are translation invariant:

$$\tilde{\mu}_{ij} = \sum_{x=0}^S \sum_{y=0}^S (x - \tilde{x})^i (y - \tilde{y})^j \tilde{f}(x, y) \quad (2.19)$$

where $\tilde{x} = \tilde{M}_{10}/\tilde{M}_{00}$, and $\tilde{y} = \tilde{M}_{01}/\tilde{M}_{00}$, which are called the centroids of the density.

We can further use the mass of the function to produce scale invariance:

$$\tilde{\eta}_{ij} = \frac{\tilde{\mu}_{ij}}{(\tilde{M}_{00})^{\frac{i+j}{2}+1}} \quad (2.20)$$

For rotational invariance, we will introduce trigonometry. Note that this methodology will change drastically in higher dimensions, instead relying on the eigenvectors of a rotation matrix.

Definition 7. 2D Rotation, Scale, and Translation Invariant Geometric Moments:

$$\begin{aligned} \tilde{\nu}_{ij} = (\tilde{M}_{00})^{-\frac{i+j}{2}-1} \sum_{x=0}^S \sum_{y=0}^S \{ \tilde{f}(x, y) \cdot [(x - \tilde{x}) \cos \tilde{\theta} + (y - \tilde{y}) \sin \tilde{\theta}]^i \\ \cdot [-(x - \tilde{x}) \sin \tilde{\theta} + (y - \tilde{y}) \cos \tilde{\theta}]^j \} \end{aligned} \quad (2.21)$$

where

$$\tan 2\tilde{\theta} = \frac{2\tilde{\mu}_{11}}{\tilde{\mu}_{20} - \tilde{\mu}_{02}} \quad (2.22)$$

with $-\frac{\pi}{4} \leq \tilde{\theta} \leq \frac{\pi}{4}$.

Instead of applying $\tilde{\nu}_{ij}$ directly, these invariants will be incorporated into the later

defined $\tilde{\lambda}_{ij}$, which will be integrated into the final descriptors.

2.3 2D Krawtchouk Descriptors

Having established scale, translation, and rotational invariance in 2D, we can now form 2D Krawtchouk descriptors. We will encompass these invariants in $\tilde{\lambda}_{ij}$, and using $\tilde{\lambda}_{ij}$, our descriptors.

$$\begin{aligned} \tilde{\lambda}_{ij} = (\tilde{M}_{00})^{-\frac{n+m}{2}-1} \sum_{x=0}^S \sum_{y=0}^S \{ & \tilde{f}(x, y) \\ & \cdot [[(x - \tilde{x}) \cos \tilde{\theta} + (y - \tilde{y}) \sin \tilde{\theta}] / (\tilde{M}_{00})^{\frac{1}{2}} + Sp_x]^i \\ & \cdot [[-(x - \tilde{x}) \sin \tilde{\theta} + (y - \tilde{y}) \cos \tilde{\theta}] / (\tilde{M}_{00})^{\frac{1}{2}} + Sp_y]^j \} \end{aligned} \quad (2.23)$$

Definition 8. 2D Krawtchouk descriptors are then given by

$$\tilde{Q}_{nm} = [\rho(n; p_x, S)\rho(m; p_y, S)]^{-\frac{1}{2}} \sum_{i=0}^n \sum_{j=0}^m a_{i,n,p_x,S} a_{j,m,p_y,S} \tilde{\lambda}_{ij} \quad (2.24)$$

where $a_{i,n,p,S}$ denotes the coefficient of the i^{th} degree term in $K_n(x; p, S)$.

2.4 Workflow

In practice, implementation of this methodology flows similarly to the outline of this paper, although several minor adjustments are made to facilitate computation. Due to high values in the choose function, computing w for every ordered pair p_x, p_y can prove quite strenuous, and if this methodology is implemented verbatim from [12], will account for the majority of runtime. Alternatively, w can be precomputed at $p_x, p_y = 0.5, 0.5$ and then translated about its center to the p_x, p_y value for every region of interest in an image. This also mitigates the distortion of the Gaussian smear when computing at extreme points, which further enhances accuracy. Due to the locality of weighted Krawtchouk functions, and the separation of ρ and w in

the final form of descriptors given in the previous Section 2.3, the coefficients of the Krawtchouk polynomials, as well as ρ , can be precomputed as well. In languages capable of symbolic computation, the coefficients of the Krawtchouk polynomials can be computed quickly using the recursive formula from 2.1 [18].

Following the precomputations outlined above, in some languages, particularly Matlab/Octave and Python’s NumPy/SciPy, reshaping the image matrix from $S \times S$ to $1 \times S^2$, then using a single summation in place of a double summation can reduce runtime. See Fig. 3 for a general outline of workflow.

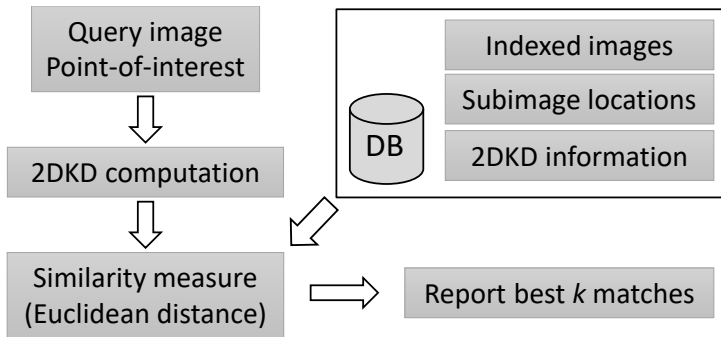


Figure 3: A flowchart of the workflow of 2D Krawtchouk descriptors in implementation. 2DKD denotes 2D Krawtchouk Descriptors, and DB denotes the database in which to search for the queried subimage [3].

To search for a subimage within a large image, or series of large images, one can simply compute the Krawtchouk descriptor of the query, then create a grid of points covering each of the larger images within the database, compute the Krawtchouk descriptors for each point in the grid, and store them as an array which describes the image as a whole. As mentioned following equation 2.9, the larger images need not be square, but to avoid the distortion of w , and machine error when computing large choose functions of w , it is ideal to temporarily crop the image into reasonably sized squares about the point of interest. Due to the locality of weighted Krawtchouk polynomials, this will not affect the descriptors, and no information will be lost along the boundary of cropping since the square will be re-cropped for every point on the grid while computing its respective descriptor. In Matlab/Octave, the `n choose k` function does not experience a high degree of error until $n > 600$, so it is recommended

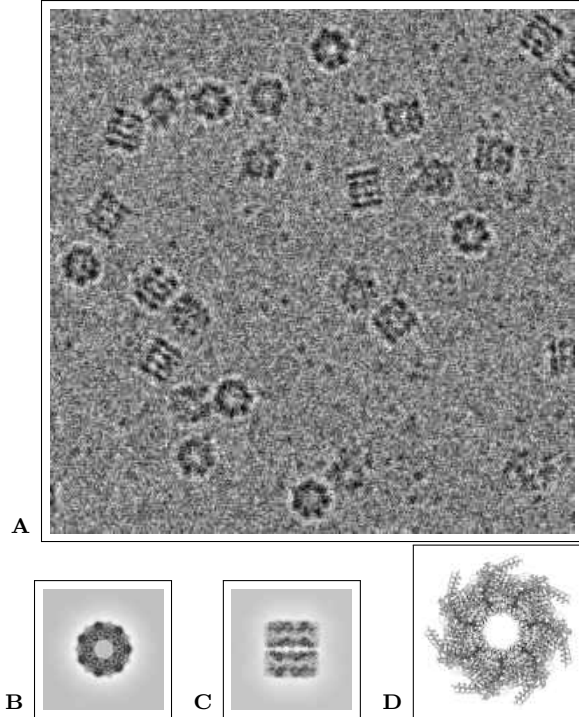


Figure 4: **A.** A section of a projection image of GroEL protein complexes in vitreous ice captured using Cryo-EM. **B.** Averaged top view of GroEL. **C.** Averaged side view of GroEL. **D.** An end-on view of the 3D atomic structure of GroEL complex. Image credits – A: Vossman, <https://commons.wikimedia.org/wiki/File:Cryoem.groel.jpg>, B, C: Electron Microscopy Data Bank (EMD-8750), D: Protein Data Bank (PDB ID: 5W0S).

to make the square roughly this size, and, as previously mentioned, precompute w for this size, then translate within the square, moving it along the image as necessary.

Once this database is created, the likelihood of match, or degree of similarity is quantified by the Euclidean distance between Krawtchouk descriptors, with 0 denoting a theoretical perfect match. Since these are strictly non-negative by definition, it is easy to order them, and produce a set of results ranked by similarity. Depending on the speed of hardware, size of the image, and order of moments used, this can take some time. Using a database of 39 roughly 1000×1000 images with 5th order Krawtchouk moments, promising results returned in a matter of hours with an ordinary laptop [3].

2.5 Example

In [3], a cryo-EM 2D projection of several identical protein structures with severe noise is used as a single-image database, in which the query is one of these structures. In particular, these are several instances of the molecular chaperonin GroEL (see Fig. 4). Since the query was taken directly from the database, the top result is an exact copy of the query, but the following ranked results still appear quite similar, with only the 4th and 14th results appearing questionable to the human eye (see Fig. 5).

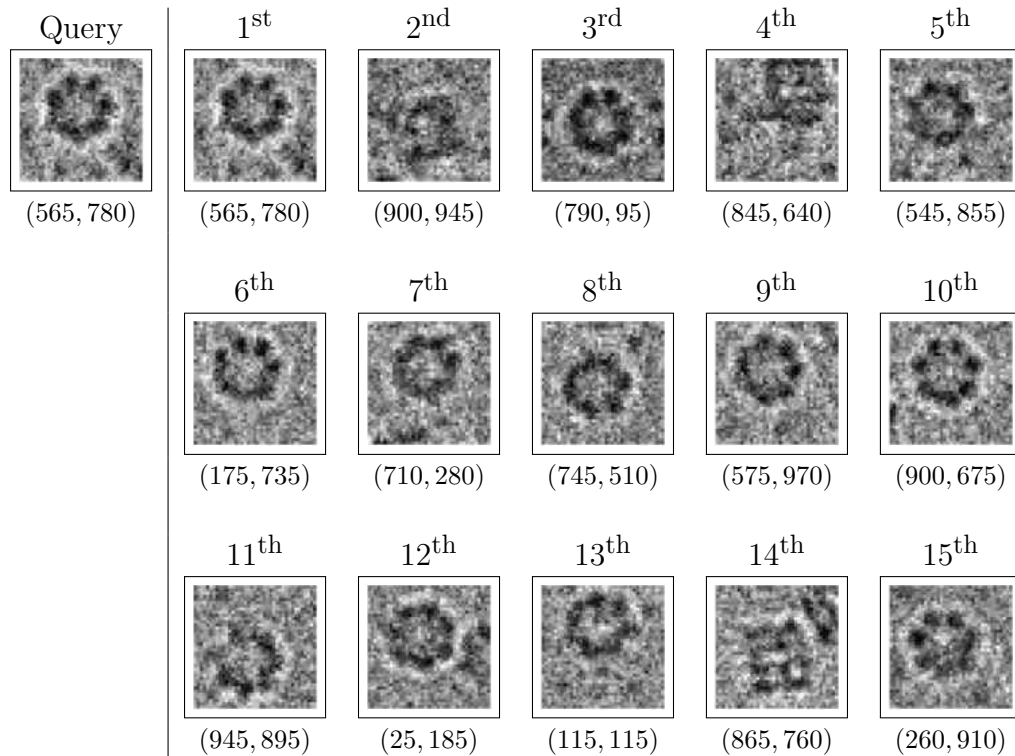


Figure 5: An example query of the top view of GroEL and top 15 retrieval results using 2DKD. The pixel size for the local subimages is 40×40 . The (x, y) centers of the query and retrieval results in the 1024×1024 global image are provided under each subimage.

3 THREE-DIMENSIONAL KRAWTCHOUK DESCRIPTORS

3.1 3D Krawtchouk Moments

Using the same initial definition of weighted Krawtchouk polynomials, orthonormal 3D Krawtchouk moments can be created as well. Let A now denote $\{0, \dots, S\} \times \{0, \dots, S\} \times \{0, \dots, S\}$, with function $f : A \rightarrow \mathbb{N} \cup \{0\}$, where we wish to describe the image of f .

Definition 9. 3D Krawtchouk Moments are defined by

$$\bar{Q}_{nml} = \sum_{x=0}^S \sum_{y=0}^S \sum_{z=0}^S \bar{K}_n(x; p_x, S) \bar{K}_m(y; p_y, S) \bar{K}_l(z; p_z, S) f(x, y, z) \quad (3.1)$$

Like before, these give us the orthonormality condition

$$\begin{aligned} \sum_{x=0}^S \sum_{y=0}^S \sum_{z=0}^S \bar{K}_n(x; p_x, S) \bar{K}_m(y; p_y, S) \bar{K}_l(z; p_z, S) \\ \cdot \bar{K}_{n'}(x; p_x, S) \bar{K}_{m'}(y; p_y, S) \bar{K}_{l'}(z; p_z, S) = \delta_{nn'} \delta_{mm'} \delta_{ll'} \end{aligned} \quad (3.2)$$

and can be reversed via

$$f(x, y, z) = \sum_{n=0}^S \sum_{m=0}^S \sum_{l=0}^S \bar{Q}_{nml} \bar{K}_n(x; p_x, S) \bar{K}_m(y; p_y, S) \bar{K}_l(z; p_z, S) \quad (3.3)$$

or reversed approximately via

$$\hat{f}(x, y, z) = \sum_{n=0}^{\hat{S}} \sum_{m=0}^{\hat{S}} \sum_{l=0}^{\hat{S}} \bar{Q}_{nml} \bar{K}_n(x; p_x, S) \bar{K}_m(y; p_y, S) \bar{K}_l(z; p_z, S) \quad (3.4)$$

As was the case in 2D, these moments can be transformed into Krawtchouk descriptors which are scale, translation, and rotation invariant. The specifics of incorporating these invariants are discussed in Section 3.2.

3.2 3D Geometric Moments

To produce invariants for Krawtchouk descriptors in three dimensions, the geometric moments discussed in Section 2.2 must be redefined to account for the z axis [13]. We begin with the 3D weighted image function \tilde{f} .

Definition 10. Using a now 3D function, \tilde{f} is defined as

$$\tilde{f}(x, y, z) = f(x, y, z) \sqrt{w(x; p_x, S)w(y; p_y, S)w(z; p_z, S)}. \quad (3.5)$$

Like before, $\tilde{\cdot}$ denotes values computed using the weighted function \tilde{f} .

Definition 11. 3D geometric moments are now defined as

$$\tilde{M}_{ijk} = \sum_{x=0}^S \sum_{y=0}^S \sum_{z=0}^S x^i y^j z^k \tilde{f}(x, y, z) \quad (3.6)$$

As before, \tilde{M}_{000} yields the mass of the function $\tilde{f}(x, y, z)$, and can be used to produce the centroids $\tilde{x} = \tilde{M}_{100}/\tilde{M}_{000}$, $\tilde{y} = \tilde{M}_{010}/\tilde{M}_{000}$, $\tilde{z} = \tilde{M}_{001}/\tilde{M}_{000}$, which can be used to produce the translation invariant moments

$$\tilde{\mu}_{ijk} = \sum_{x=0}^S \sum_{y=0}^S \sum_{z=0}^S (x - \tilde{x})^i (y - \tilde{y})^j (z - \tilde{z})^k \tilde{f}(x, y, z) \quad (3.7)$$

and we obtain the scale and translation invariant value $\tilde{\eta}_{ijk}$ via

$$\tilde{\eta}_{ijk} = \frac{\tilde{\mu}_{ijk}}{(\tilde{M}_{000})^{\frac{i+j+k}{3}} + 1} \quad (3.8)$$

Rotational invariance is where the our methodology will differ from the 2D case. In lieu of straightforward trigonometry, we will rotate the image of $\tilde{f}(x, y, z)$ to align its principle axes with the x , y , and z axes respectively. With our function positioned at the origin, the principle axes of $\tilde{f}(x, y, z)$ are defined as the eigenvectors

of the inertia matrix [6].

Definition 12. Inertia Matrix

$$\tilde{I} = \begin{bmatrix} \tilde{I}_{xx} & \tilde{I}_{xy} & \tilde{I}_{xz} \\ \tilde{I}_{yx} & \tilde{I}_{yy} & \tilde{I}_{yz} \\ \tilde{I}_{zx} & \tilde{I}_{zy} & \tilde{I}_{zz} \end{bmatrix} \quad (3.9)$$

where

$$\tilde{I}_{xx} = \tilde{\mu}_{020} + \tilde{\mu}_{002}$$

$$\tilde{I}_{yy} = \tilde{\mu}_{200} + \tilde{\mu}_{002}$$

$$\tilde{I}_{zz} = \tilde{\mu}_{200} + \tilde{\mu}_{020}$$

$$\tilde{I}_{xy} = \tilde{I}_{yx} = -\tilde{\mu}_{110}$$

$$\tilde{I}_{xz} = \tilde{I}_{zx} = -\tilde{\mu}_{101}$$

$$\tilde{I}_{yz} = \tilde{I}_{zy} = -\tilde{\mu}_{011}$$

Note that \tilde{I} is a symmetric matrix, and has real eigenvalues $\{\lambda_1, \lambda_2, \lambda_3\}$, and orthogonal eigenvectors $\{u_1, u_2, u_3\}$, with the condition

$$\tilde{I}u_i = \lambda_i u_i \quad (3.10)$$

for $i = 1, 2, 3$. These eigenvectors will ultimately serve as the columns of the rotation matrix which rotates $\tilde{f}(x, y, z)$ to the x , y , and z axes as described previously; however, for all i , u_i and $-u_i$ both remain valid eigenvectors, and thus there are 8 possible rotation matrices to choose from, each corresponding to an octant of \mathbb{R}^3 . We will choose the rotation matrix which corresponds to the all-negative octant using the provided surface normal vector to the point (p_x, p_y, p_z) where the Krawtchouk descriptor is being generated.

This should place our region of interest in the all positive octant to facilitate later calculations. For each u_i , if the dot product of u_i and the surface normal vector is negative, then choose $-u_i$ for the i^{th} column of the rotation matrix \tilde{R} . Otherwise, choose u_i . This then produces the rotation matrix \tilde{R} used in Section 3.3 to ensure rotational invariance of the Krawtchouk descriptor.

3.3 3D Krawtchouk Descriptors

Having obtained all necessary invariants, the translation, scale, and rotation invariant 3D Krawtchouk Descriptors can now be computed with the following variables.

$$\begin{aligned} \tilde{\lambda}_{ijk} = & (\tilde{M}_{000})^{-1} \sum_{x=0}^S \sum_{y=0}^S \sum_{z=0}^S \tilde{f}(x, y, z) \\ & \cdot (\tilde{\phi}_1(x, y, z)/(\tilde{M}_{000})^{1/3} + S/2)^i \\ & \cdot (\tilde{\phi}_2(x, y, z)/(\tilde{M}_{000})^{1/3} + S/2)^j \\ & \cdot (\tilde{\phi}_3(x, y, z)/(\tilde{M}_{000})^{1/3} + S/2)^k \end{aligned} \quad (3.11)$$

where

$$\begin{aligned} \tilde{\phi}_1(x, y, z) &= \tilde{R}_{11}(x - \tilde{x}) + \tilde{R}_{12}(y - \tilde{y}) + \tilde{R}_{13}(z - \tilde{z}) \\ \tilde{\phi}_2(x, y, z) &= \tilde{R}_{21}(x - \tilde{x}) + \tilde{R}_{22}(y - \tilde{y}) + \tilde{R}_{23}(z - \tilde{z}) \\ \tilde{\phi}_3(x, y, z) &= \tilde{R}_{31}(x - \tilde{x}) + \tilde{R}_{32}(y - \tilde{y}) + \tilde{R}_{33}(z - \tilde{z}) \end{aligned}$$

We will use this as a factor to encompass these invariants in the final descriptor.

Definition 13. 3D Krawtchouk Descriptor

$$\tilde{Q}_{nml} = [\Omega(n, m, l; p^*, \mathcal{S})]^{-1/2} \sum_{i=0}^n \sum_{j=0}^m \sum_{k=0}^l a_{i,n,p_x,S} a_{j,m,p_y,S} a_{k,l,p_z,S} \tilde{\lambda}_{ijk} \quad (3.12)$$

where $a_{i,n,p,S}$ denotes the i^{th} coefficient of $K_n(x;p,S)$, and

$$\Omega(n, m, l, p^*, \mathcal{S}) = \rho(n; p_x, S)\rho(m; p_y, S)\rho(l; p_z, S)$$

For simple computation, we assume $p_x = p_y = p_z = 0.5$, which is discussed further in Section 3.4.

3.4 Workflow

Similar to the 2D implementation, the overall process consists of generating descriptors at various points about the object, creating a database of them, and then comparing them to a query descriptor via Euclidean distance; however, due to the more specialized data used in application, the selection of points is more specific than a general grid, and some additional precomputations must take place.

The testing done for 3D Krawtchouk descriptors in [13] assumes the 3D object is a shell with 1s on the surface of the object, and 0s elsewhere, which would make a grid approach impractical because the majority of points would be trivial descriptions of empty space, and instead, a set of surface points were selected by LZerD software at Kihara Lab [4] from a triangulated surface mesh of the shell. The surface normal vectors were computed from the tangent plane to a given point on this triangulated surface, which are necessary to producing rotational invariance, as mentioned in section 3.2. These are further discussed in Section 4.2, when surface normal vectors are computed for a theoretical n D surface.

Also similar to the 2D implementation, the Krawtchouk polynomial coefficients are precomputed via a recurrence relation, and weight function is precomputed at $p_x, p_y, p_z = 0.5$ and translated to the needed p_x, p_y, p_z , and the now 3D object array is reshaped from $S \times S \times S$ to $1 \times S^3$ to facilitate computation. Since runtime is drastically increased for 3D objects, these precomputations are all the more important, and in testing, small arrays were used. In 2D, test images were roughly 1000×1000 , and in

the following test, images were $170 \times 170 \times 170$. Although this methodology could work for objects other than a shell, perhaps accounting for density, the runtimes would be unfeasible with the hardware available for this work.

Lastly, in addition to reshaping the 3D object array, the ϕ values used in λ can also be altered into 2D objects to drastically reduce runtime, using the following methodology from [13]:

First, we rewrite our ε values like so:

$$(\tilde{\phi}_1(x, y, z))^r = \sum_{\varepsilon_1=0}^r \binom{r}{\varepsilon_1} (\tilde{A}_1(x, y))^{r-\varepsilon_1} (\tilde{D}_1(z))^{\varepsilon_1}, \quad (3.13)$$

$$(\tilde{\phi}_2(x, y, z))^s = \sum_{\varepsilon_2=0}^s \binom{s}{\varepsilon_2} (\tilde{A}_2(x, y))^{s-\varepsilon_2} (\tilde{D}_2(z))^{\varepsilon_2}, \quad (3.14)$$

$$(\tilde{\phi}_3(x, y, z))^t = \sum_{\varepsilon_3=0}^t \binom{t}{\varepsilon_3} (\tilde{A}_3(x, y))^{t-\varepsilon_3} (\tilde{D}_3(z))^{\varepsilon_3}, \quad (3.15)$$

where $\tilde{A}_\tau(x, y) = \tilde{R}_{\tau 1}(x - \tilde{x}) + \tilde{R}_{\tau 2}(y - \tilde{y})$ and $\tilde{D}_\tau(z) = \tilde{R}_{\tau 3}(z - \tilde{z})$ with $\tau = 1, 2, 3$.

1. Compute the auxiliary image $\tilde{f}(x, y, z)$, via

$$\tilde{f}(x, y, z) = f(x, y, z) \sqrt{w(x; p_x, S)w(y; p_y, S)w(z; p_z, S)}. \quad (3.16)$$

2. Compute

$$\tilde{T}(x, y; \varepsilon_1, \varepsilon_2, \varepsilon_3) = \sum_{z=0}^S \tilde{f}(x, y, z) (\tilde{D}_1(z))^{\varepsilon_1} (\tilde{D}_2(z))^{\varepsilon_2} (\tilde{D}_3(z))^{\varepsilon_3} \quad (3.17)$$

for $0 \leq \varepsilon_1 + \varepsilon_2 + \varepsilon_3 \leq 5$. With z eliminated, the computation is now in just 2 dimensions.

3. Compute

$$\tilde{T}_3(x, y; \varepsilon_1, \varepsilon_2, t) = \sum_{\varepsilon_3=0}^t \binom{t}{\varepsilon_3} (\tilde{A}_3(x, y))^{t-\varepsilon_3} \tilde{T}(x, y; \varepsilon_1, \varepsilon_2, \varepsilon_3) \quad (3.18)$$

for $0 \leq \varepsilon_1 + \varepsilon_2 + t \leq 5$.

4. Compute

$$\tilde{T}_2(x, y; \varepsilon_1, s, t) = \sum_{\varepsilon_2=0}^s \binom{s}{\varepsilon_2} (\tilde{A}_2(x, y))^{s-\varepsilon_2} \tilde{T}_3(x, y; \varepsilon_1, \varepsilon_2, t) \quad (3.19)$$

for $0 \leq \varepsilon_1 + s + t \leq 5$.

5. Compute

$$\tilde{T}_1(x, y; r, s, t) = \sum_{\varepsilon_1=0}^r \binom{r}{\varepsilon_1} (\tilde{A}_1(x, y))^{r-\varepsilon_1} \tilde{T}_2(x, y; \varepsilon_1, s, t) \quad (3.20)$$

for $0 \leq r + s + t \leq 5$.

6. Compute

$$\tilde{\nu}_{rst} = (\tilde{M}_{000})^{-\frac{r+s+t}{3}-1} \sum_{x=0}^S \sum_{y=0}^S \tilde{T}_1(x, y; r, s, t) \quad (3.21)$$

for $0 \leq r + s + t \leq 5$.

7. Compute $\tilde{\lambda}_{ijk}$ in for $0 \leq i + j + k \leq 5$ like so using our results thus far:

$$\tilde{\lambda}_{ijk} = \sum_{r=0}^i \sum_{s=0}^j \sum_{t=0}^k \binom{i}{r} \binom{j}{s} \binom{k}{t} \cdot \frac{\tilde{\nu}_{rst} S^{i+j+k-r-s-t}}{2} \quad (3.22)$$

8. Compute \tilde{Q}_{nml} for $0 \leq n + m + l \leq 5$ using $\tilde{\lambda}_{ijk}$ as follows:

$$Q_{nml} = [\Omega(n, m, l; p^*, \mathcal{S})]^{-1/2} \sum_{i=0}^n \sum_{j=0}^m \sum_{k=0}^l a_{i,n,0.5,S} a_{j,m,0.5,S} a_{k,l,0.5,S} \tilde{\lambda}_{ijk} \quad (3.23)$$

where $a_{i,n,p,S}$ denotes the i^{th} coefficient of $K_n(x; p, S)$, and

$$\Omega(n, m, l, p^*, \mathcal{S}) = \rho(n; 0.5, S)\rho(m; 0.5, S)\rho(l; 0.5, S)$$

3.5 Rotation Test

Using efficient implementation of 3D Krawtchouk Descriptors with Krawtchouk moments of order 5, and thus descriptors consisting of 56 elements, the same object was described at a selection of 1,608 points before and after a series of 3 rotations to test the translation and rotational invariance of the descriptors, in addition to their general consistency. The particular object is a $170 \times 170 \times 170$ array depicting the protein 1gco, a crystal structure of glucose dehydrogenase (GlcDH), which is represented by values of 1 along the surface points, and 0 elsewhere. This was also supplied with a set of surface normal vectors about these preselected points to enable the computation of the rotation matrix R , as outlined in Section 3.2. All computation beyond the supplied data was done in GNU Octave 4.2.2 on a PC running Linux Fedora 29 virtualized in Windows 10 on an Intel core i7-7700k overclocked to 4.6GHz, with 16GB DDR4 RAM at 2400MHz, 8GB of which was allotted to the virtual machine. Also note that the computations were not parallelized or multithreaded on this machine. The RCSB Protein Data Bank currently has 46,602 distinct protein sequences, which, at this rate on this machine, would take about 15 minutes to search with this methodology, assuming the sorting process runs in a similar time-frame, and the proteins have been described with the same methodology with a similar number of surface points [2].

Table 1: Rotation Test: Points Correctly Identified by Minimum Euclidean Distance of Krawtchouk Descriptors

| Correct Region Found: | Rotation 1: | Rotation 2: | Rotation 3: |
|--------------------------------------|--------------------|--------------------|--------------------|
| As Top Result | 92.70% | 88.37% | 88.50% |
| Within Top 5 | 97.76% | 96.39% | 97.01% |
| Within Top 10 | 97.75% | 97.02% | 97.26% |
| Average Runtime per Descriptor | 0.422s | 0.426s | 0.426s |
| Total Runtime for Object Description | 678.02s | 685.03s | 684.92s |
| Average Runtime per Search | .018s | .018s | .018s |

These results are aimed to compare to those of [13], although accuracy came out lower due to a difference in data. In [13], numerous points for removed to ensure clear distinction between regions, whereas in the results shown in Fig. 1, no points were removed. Given the high accuracy and low runtimes with more points, many of which were near each other and thus described similar regions, these results are promising for later implementation.

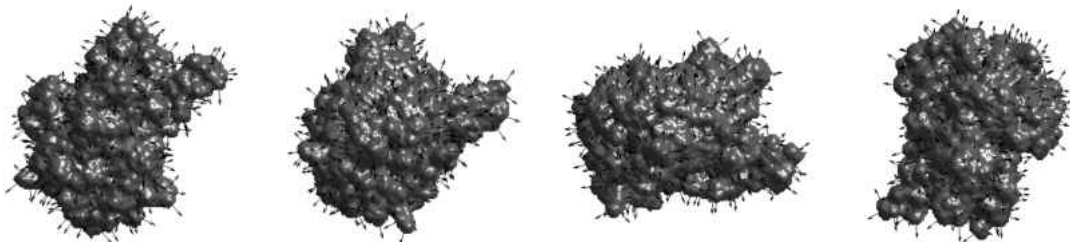


Figure 6: Query protein surface (1gco.pdb, left) and three target surfaces obtained from the query protein by rotating it using different rotation matrices. The vertex normals on each surface are also demonstrated. Each protein surface is placed in a voxel grid of size 170^3 .

4 n-DIMENSIONAL KRAWTCHOUK DESCRIPTORS

4.1 n-Dimensional Krawtchouk Moments

Hypothetically, a slightly modified implementation of this Krawtchouk descriptor methodology could be implemented in n dimensions. Recall the definition of Krawtchouk moments from Section 2.1. $\bar{K}_s(x; p, S)$ is computed for a single variable, and thus single dimension at a time, and thus the Krawtchouk polynomials need not change for n dimensional Krawtchouk moments, and we can simply compute $\bar{K}_s(x_i; p, S_i)$ for $i = 0, \dots, n$. Considering $A = \{0, \dots, S\}$, we then have $A^n = A \times A \times \dots \times A$. We then define $f(\vec{x}) : A^n \rightarrow \mathbb{N} \cup \{0\}$ as an n dimensional hyperobject, this will produce Krawtchouk moments in n dimensions.

Definition 14. nD Krawtchouk polynomial can be constructed like so

$$\mathcal{K}_{\vec{s}}(\vec{x}; \vec{p}, S) = \bar{K}_{s_1}(x_1; p_1, S) \cdot \dots \cdot \bar{K}_{s_n}(x_n; p_n, S) = \prod_{i=1}^n \bar{K}_{s_i}(x_i; p_i, S), \quad (4.1)$$

where $\vec{x}, \vec{s} \in A^n$ and $\vec{p} = (p_1, p_2, \dots, p_n)$.

The moments of these polynomials will then give us weighted Krawtchouk moments $\bar{Q}_{\vec{s}}$.

Definition 15. nD Krawtchouk moments are created from nD as follows

$$\bar{Q}_{\vec{s}} = \sum_{\vec{x} \in A^n} f(\vec{x}) \mathcal{K}_{\vec{s}}(\vec{x}; \vec{p}, S)$$

These descriptors, like in the 2D and 3D cases, can be used to reconstruct f generally like so:

$$f(\vec{x}) = \sum_{\vec{s} \in A^n} \bar{Q}_{\vec{s}} \mathcal{K}_{\vec{s}}(\vec{x}; \vec{p}, S)$$

Alternatively, these weighted Krawtchouk moments can also reconstruct a local approximation via

$$\hat{f}(\vec{x}) = \sum_{\vec{s} \in \hat{A}^n} \bar{Q}_{\vec{s}} \mathcal{K}_{\vec{s}}(\vec{x}; \vec{p}, S),$$

where $\hat{A} = \{0, \dots, \hat{S}\}$ with $0 \leq \hat{S} \leq S$, and $\hat{A}^n = \hat{A} \times \hat{A} \times \dots \times \hat{A}$. Up to this point, the dimension of A has little impact on the process of generating general Krawtchouk descriptors, and reconstructing f .

4.2 nD Geometric Moments

Invariance, on the other hand, becomes arduous in n dimensions. To begin, since scale invariance is simply a scalar constant appended to all Krawtchouk moments via multiplication, we will establish scale invariance using the same method as before, adjusted for n dimensions. However, we must first redefine geometric moments more generally.

Definition 16. Geometric moments in n dimensions are as follows:

$$M_{\vec{s}} = \sum_{\vec{x} \in A^n} \left(\prod_{i=1}^n x_i^{s_i} \right) f(\vec{x}) \quad (4.2)$$

$$\mu_{\vec{s}} = \sum_{\vec{x} \in A^n} \left(\prod_{i=1}^n (x_i - \bar{x}_i)^{s_i} \right) f(\vec{x}) \quad (4.3)$$

where $\vec{e}_1 = (1, 0, \dots, 0)$, $\vec{e}_2 = (0, 1, 0, \dots, 0)$, \dots , $\vec{e}_n = (0, 0, \dots, 1)$, and the center of mass is $(\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)$ where $\bar{x}_i = \frac{M_{\vec{e}_i}}{M_{\vec{0}}}$

With these, we can then define the scale and translation invariant geometric moments like so:

$$\eta_{\vec{s}} = \frac{\mu_{\vec{s}}}{(M_{\vec{0}})^{\frac{1}{n}(\sum_{i=0}^n s_i)+1}}, \quad (4.4)$$

Unfortunately, rotational invariance proves far more difficult to obtain in n dimensions. Setting aside the practical concerns such as what a rotation in n dimensions denotes, the methodology implemented thus far for obtaining the rotation matrix of f relies on surface normal vectors distributed across the surface of the object in question, which are difficult to obtain.

In 3D, we utilize outside software, as mentioned, which utilizes a triangulated surface mesh to obtain the surface normal vector. In n D, in lieu of an approximate tangent plane to a given point on the surface, or the cross product of vectors in the triangulated closed surface mesh, we choose the nearest 3 vectors, extend them through the chosen vertex, then average the set of them to obtain an approximate surface normal vector.

Choose point $x^{(0)}$ from the simplex mesh of an n D hypersurface. Choose the nearest n vertices in the mesh, and call them $x^{(1)}, x^{(2)}, \dots, x^{(n)}$. We can then average them, and obtain our surface normal vector approximation via

$$v^{(0)} = \sum_{j=1}^n \frac{x^{(0)} - x^{(j)}}{n}. \quad (4.5)$$

However, $v^{(0)}$ may not point outward from the hyperobject, as this portion of the simplex mesh could be concave or convex, and this remains an open problem. We compute $-v^{(0)}$ as a vector in the opposite direction of $v^{(0)}$ with respect to $x^{(0)}$ as a second option. Here, we will assume the hypersurface is convex, so we can assume the proper vector choice is $v^{(0)}$.

Upon acquiring this normal vector, rotational invariance can be achieved in n D using the same method used in 3D, now generalized. To begin, form the inertia matrix I from the aforementioned geometric moments.

Definition 17. Inertia Matrix:

$$I = \begin{bmatrix} I_{E_1} & I_{e_1,e_2} & I_{e_1,e_3} & \cdots & I_{e_1,e_n} \\ I_{e_2,e_1} & I_{E_2} & I_{e_2,e_3} & \cdots & I_{e_2,e_n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ I_{e_n,e_1} & I_{e_n,e_2} & I_{e_n,e_3} & \cdots & I_{E_n} \end{bmatrix} \quad (4.6)$$

where $I_{e_i,e_j} = -\mu_{e_i+e_j}$, and $I_{E_i} = \mu_{E_i}$ with $E_i = [\sum_{k=1}^n e_k] - e_i$.

Notice that the matrix is both symmetric and real, and will have real eigenvalues $\{\lambda_1, \dots, \lambda_n\}$ with corresponding orthogonal eigenvectors $\{u_1, \dots, u_n\}$. Upon obtaining this matrix, we then compute these eigenvectors. From these, we obtain 2^n possible rotation matrices with $\{\pm u_1, \dots, \pm u_n\}$ as possible columns. Each of these rotation matrices will send the previously computed normal vector to a corresponding region of \mathbb{R}^n . In \mathbb{R}^2 , there are 4 rotation matrices corresponding to the 4 quadrants, in \mathbb{R}^3 , there are 8 rotation matrices corresponding to 8 quadrants, and generally, \mathbb{R}^n will yield 2^n possible rotations. In order to achieve rotational invariance, we must choose a single rotation matrix corresponding to a single region.

In lieu of assessing the rotation matrix as a whole, we instead assess the eigenvectors individually to determine the proper rotation matrix one column at a time. For every u_i , if the dot product of u_i and the surface normal vector is negative, return $-u_i$ to the rotation matrix. Otherwise, return u_i . Conceptually, this ensures that the surface normal vector is sent to the all-positive region of \mathbb{R}^n , which in turn will place the portion of the surface in question to the all-negative region of \mathbb{R}^n . This ensures we establish a convention, so the surface will always be rotated to the same position.

4.3 nD Krawtchouk Descriptors

Upon computing the rotation matrix, the invariant descriptors can be computed like so:

$$\lambda_{\vec{x}} = (M_{\vec{0}})^{-1} \sum_{\vec{x} \in A^n} \prod_{i=1}^n \phi_i(\vec{x}) f(\vec{x}) \quad (4.7)$$

where

$$\phi_i(\vec{x}) = \sum_{j=1}^n R_{(e_i+e_j)}(x_j - \bar{x}_j) \quad (4.8)$$

With these computed, we obtain our final set of descriptors, \mathcal{Q} :

$$\mathcal{Q}_{\vec{s}} = \left[\prod_{i=0}^n \rho(s_i; p_i, S) \right]^{-\frac{1}{2}} \sum_{\vec{\sigma}=\vec{0}}^{\vec{s}} \left[\prod_{i=0}^n a_{\sigma_i, s_i, p_i, S} \right] \lambda_{\vec{\sigma}}, \quad (4.9)$$

where $a_{\sigma_i, s_i, p_i, S}$ denotes the coefficient of the σ_i degree term in $K_{s_i}(x_i; p_i, S)$.

This n D methodology, while theoretically possible given a convex hyperobject, is not currently aimed and any specific application, and would have daunting runtimes in practice. The concepts of machine vision and image processing are lost when assessing higher dimensional data, and this would instead serve merely as a pattern recognition within datasets with high dimensions. The concept of rotational invariance is less than intuitive in higher dimensions as well, and given the insurance of rotational invariance is solely responsible for the convexity requirement, scale and translation invariance alone may prove more feasible in practice, should demand for higher dimensional pattern recognition in this manner emerge.

While the processing power and memory necessary for this calculation are immense, the length of the descriptors could also grow unfeasible sizes, even with relatively low orders of Krawtchouk moments as the dimension of the data grows large. To see this, we generalize the formula for length of descriptors from Section 2.1 for

nD :

$$\Upsilon = \frac{\prod_{i=0}^n (v + i)}{n!}, \quad (4.10)$$

where v is the order of moments used. Included is a brief table of descriptor lengths for varying v and n (Table 2).

Table 2: Length of Krawtchouk Descriptors in Varying Dimension and Order

| Dimension | $v = 5$ | $v = 6$ | $v = 7$ | $v = 10$ |
|------------------|-------------------|-------------------|----------------------|----------------------|
| 4 | 126 | 200 | 330 | 1001 |
| 5 | 252 | 462 | 792 | 3003 |
| 6 | 462 | 924 | 1716 | 8008 |
| 10 | 3003 | 8008 | 19448 | 184756 |
| 100 | $9.66 \cdot 10^7$ | $1.71 \cdot 10^9$ | $2.61 \cdot 10^{10}$ | $4.69 \cdot 10^{13}$ |

BIBLIOGRAPHY

- [1] R.E. Attar. *Legendre Polynomials and Functions*. CreateSpace Independent Publishing Platform, 2009.
- [2] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.
- [3] Julian S DeVille, Daisuke Kihara, and Atilla Sit. 2DKD: A toolkit for content based local image search. *Source Code for Biology and Medicine*, submitted, 2019.
- [4] Juan Esquivel-Rodriguez, Vianney Filos-Gonzalez, Bin Li, and Daisuke Kihara. Pairwise and multimeric protein–protein docking using the LZerD program suite. In *Protein Structure Prediction*, pages 209–234. Springer, 2014.
- [5] Philip Feinsilver and Jerzy Kocik. Krawtchouk polynomials and Krawtchouk matrices. In *Recent advances in applied probability*, pages 115–141. Springer, 2005.
- [6] JM Galvez and M Canton. Normalization and shape recognition of three-dimensional objects by 3D moments. *Pattern Recognition*, 26(5):667–681, 1993.
- [7] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2):179–187, 1962.
- [8] Vladimir I Levenshtein. Krawtchouk polynomials and universal bounds for codes and designs in Hamming spaces. *IEEE Transactions on Information Theory*, 41(5):1303–1321, 1995.
- [9] Robert J Marinelli, Kelli Montgomery, Chih Long Liu, Nigam H Shah, Wijan Prapong, Michael Nitzberg, Zachariah K Zachariah, Gavin J Sherlock, Yasodha

- Natkunam, Robert B West, et al. The Stanford tissue microarray database. *Nucleic acids research*, 36(suppl_1):D871–D877, 2007.
- [10] Jacek Nowakowski and Dominik Sankowski. *Computer Vision In Robotics And Industrial Applications*. Number vol. 3 in Series in Computer Vision. World Scientific, 2014.
- [11] GY Pryzva. Kravchuk orthogonal polynomials. *Ukrainian Mathematical Journal*, 44(7):792–800, 1992.
- [12] Atilla Sit and Daisuke Kihara. Comparison of image patches using local moment invariants. *IEEE Transactions on Image Processing*, 23(5):2369–2379, 2014.
- [13] Atilla Sit, Woong-Hee Shin, and Daisuke Kihara. Three-dimensional Krawtchouk descriptors for protein local surface shape comparison. *Pattern Recognition*, in revision, 2019.
- [14] Michael Reed Teague. Image analysis via the general theory of moments. *J. Opt. Soc. Am.*, 70(8):920–930, Aug 1980.
- [15] Bo Yang and Mo Dai. Image reconstruction from continuous Gaussian–Hermite moments implemented by discrete algorithm. *Pattern Recognition*, 45(4):1602–1616, 2012.
- [16] Bo Yang, Tomáš Suk, Mo Dai, and Jan Flusser. 2D and 3D image analysis by Gaussian-Hermite moments. *Moments and Moment Invariants-Theory and Applications*, 1:143–173, 2014.
- [17] P-T Yap, Raveendran Paramesran, and Seng-Huat Ong. Image analysis by Krawtchouk moments. *IEEE Transactions on image processing*, 12(11):1367–1377, 2003.

- [18] Guojun Zhang, Zhu Luo, Bo Fu, Bo Li, Jiaping Liao, Xiuxiang Fan, and Zheng Xi. A symmetry and bi-recursive algorithm of accurately computing Krawtchouk moments. *Pattern Recognition Letters*, 31(7):548–554, 2010.
- [19] H. Zhang, H. Shu, G. N. Han, G. Coatrieux, L. Luo, and J. L. Coatrieux. Blurred image recognition by Legendre moment invariants. *IEEE Transactions on Image Processing*, 19(3):596–611, March 2010.