GEOMETRIC CONTEXT DRIVEN INFERENCE FOR HIGH THROUGHPUT CRYOGENIC ELECTRON TOMOGRAPHY

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL
ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Farshid Moussavi

August 2010

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Mark A. Horowitz) Principal Co-Advisor

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Daphne Koller) Principal Co-Advisor

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Harley McAdams)

Approved for the University Committee on Graduate Studies.

Abstract

Cryogenic Electron Tomography (Cryo-ET) has gained increasing interest in recent years due to its ability to image whole cells and subcellular structures in 3D at nanometer resolution in their native environment. However, due to dose restrictions and the inability to acquire high tilt angle images, the reconstructed volumes are noisy and have missing information. In order to overcome these limitations and fulfill the promise of this method, it is necessary to image numerous instances of the same underlying object and average them, requiring a high throughput pipeline. Currently, the bottlenecks in the electron tomography pipeline are a set of image inference tasks which require manual intervention by an expert due to weak and unreliable local image features. In this thesis we propose the use of geometric context in a structured probabilistic models framework to overcome the low reliability of local features and achieve automation and high throughput for two of the bottleneck tasks- precision registration of 2D images and 3D segmentation of whole cells. The central idea in our approach is to overcome the uncertainty from unreliable features by exploiting their mutual geometric and spatial relationships in varying degrees of locality to classify them more accurately. Structured probabilistic models provide a framework for encoding a diverse set of geometric relationships, as well as a substantial body of efficient yet effective approximate inference algorithms.

space

– space

In the first problem of precision registration of 2D images, the features are a set of gold markers which can be difficult to distinguish at high tilt angles. Precision alignment of the images requires the successful tracking of these markers throughout the series of images. We track markers jointly as a group, using the geometric relationship of the markers. Therefore the geometric relationship of interest for overcoming

the unreliable features in this case is the pattern formed by the gold markers. We encode the relative geometric arrangement of pairs of markers as pairwise factors in a Markov random field (MRF) framework, and use loopy belief propagation to find the most likely correspondence of markers between images. This approach, called RAPTOR (Robust Alignment and Projection estimation for TOmographic Reconstruction) has resulted in successful automatic full precision alignment of electron tomography tilt series.

The second problem of 3D segmentation of whole cells is challenging due to uncertain boundary characteristics. Intensity and intensity gradients based methods easily confuse many non boundary pixels as boundaries, and therfore precision extraction of the cell boundary is difficult, manual and time intensive. We present an efficient recursive algorithm called BLASTED (Boundary Localization using Adaptive Shape and TExture Discovery) to automatically extract the cell boundary using a conditional random field (CRF) framework in which boundary points and shape are jointly inferred with the help of a learned boundary feature detector and shape evolution model. The algorithm learns the texture of the boundary region progressively, and uses a global shape model and shape-dependent features to propose candidate boundary points on a slice of the membrane. It then updates the shape of that slice by accepting the appropriate candidate points using local spatial clustering, the global shape model, and trained boosted texture classifiers. This method has successfully segmented numerous datasets starting from one hand labelled slice each, reducing the processing time from days to hours.

Acknowledgments

I would like to thank my advisor, Professor Mark Horowitz.

I would like to thank my coadvisor, Professor Daphne Koller.

I would like to thank Professor Harley McAdams and Professor Lucy Shapiro.

I would like to thank my friends and colleagues- Geremy, Gal, Fernando, Luis, Ken, Rick, Stephen, Sewoong, ...more

I would like to thank my family. This thesis is dedicated to them.

Contents

List of Figures

Chapter 1

Introduction

(Style thing — I generally don't like to headers in a row. But that is just me

1.1 High Resolution 3D Imaging of Cells

Imaging has been one of the main enablers of new discoveries in the life sciences. The combination of more imaging modalities, advances in image processing and computer vision algorithms, and the availability of fast, inexpensive, parallel computing has poised the field of biomedical imaging for an era of rapid advancement. The mission of this field is quickly being transformed into one of mining large quantities of imaging data to find the most biologically relevant information efficiently.

In particular, the fields of structural biology and cell biology stand to benefit tremendously from the ability to image cells and subcellular structures in three dimensions at molecular resolutions (several *nm* or less). Such a capability gives the biologist a view onto details of relevant structures and their organization inside cells, as well as the ability to study macromolecules in their native state and context inside the cell. Two examples of biological studies in which such detailed structures need to be observed are shown in Fig. 1.1.

The first example (Fig. 1.1 (a)) illustrates the tracing of neurons in a mouse cortex in [JBH$^+$09]. The goal of this work is to extract the connectivity of neurons, and requires high resolution imaging of large quantities of data. This example represents 1 mm^3 of tissue imaged at nanometer resolution. The second example

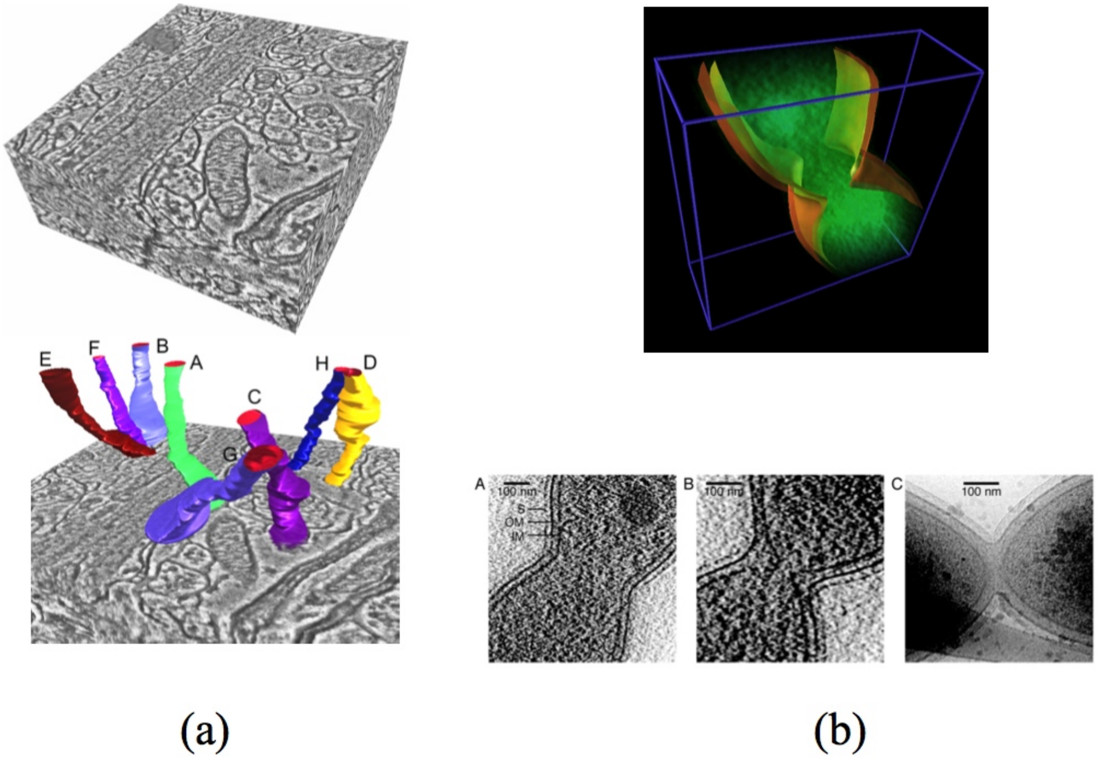


Figure 1.1: Two examples of cell biology studies in which detailed 3D structures must be observed. (a)shows the tracing of neurons in a mouse cortex [JBH$^+$09], and (b) shows the study of spatial and temporal behavior of the cell membrane at the division plane of a Caulobacter crescentus ([CJ05])

(Fig. 1.1 (b)) illustrates a study of spatial and temporal behavior of the cell membrane in cell division of Caulobacter crescentus ([CJ05]), and dealt with 5 datasets of 2 μm^3 each at the same resolution of the first example.

In such studies, the goal is to zoom into a cell, and see the organization and arrangement of groups of structures, such as cell membrane shape deformations, ribosomes, actin filaments, or other organelles in three dimensions. Indeed it has been shown that proteins and other macromolecules in the cell are highly organized (references), and that this organization is significant to the understanding of cell functions. The primary instrument used for capturing images with such detail is the electron microscope. By taking multiple images of the same sample from different angles with the electron microscope, it is possible to reconstruct a three dimensional reconstruction of the sample using tomography. This process is called electron tomography (ET). When the sample is cryogenically prepared, this process is called cryogenic electron tomography (CET).

Cryogenic electron tomography promises to fill an important ”imaging gap” in the biomedical imaging spectrum as depicted in Fig. 1.2. To the right of this gap, there is light microscopy which provides resolutions in the range of hundreds of *nm*. Superresolution techniques such as photo-activated localization microscopy (PALM) [BPS+06], stochastic optical reconstruction microscopy (STORM) [RBZ06] offer higher effective resolution of 10’s of *nm*, but are targeted to individual types of molecules which can be fluourescently labeled. Therefore it is difficult to see spatial organization of different molecular structures. To the left of the imaging gap are spectroscopy techniques, which can resolve high resolution structure of molecules, but the molecules must be isolated first. Therefore the context of spatial arrangement of different moleculare structures within the cell is still not observable. ~~Therefore~~, T~~t~~his gap is crucial for studying the internal molecular organization of cells, and it would be desirable for CET to fill this gap if possible.

While both cases in Fig. 1.1 illustrate the ability and utility of imaging cells in 3D at high resolution, the amount of data being analyzed is quite different. Both cases were at roughly 1 nm^3 per voxel. The first case is from 2009, and contains over 1000 TBytes of data, whereas the second one from 2005 has roughly 10 GBytes of data. Such increase in capacity to generate such images has occurred ~~recently~~ in the past few years due to substantial advances in the microscopes and their automation. Therefore the acquisition of data has achieved true high throughput capability. This trend has put significant pressure on the analysis and post-processing portion of the pipeline, which still requires significant manual intervention. As we shall see, the obstacles to high throughput have been a number of inference tasks in which local features and local regions are not sufficently informative.

CET datasets suffer from two major challenges- low signal to noise ratio (SNR) and missing data. The low SNR is due to limitations on total dose in order to avoid damaging the sample. The missing data is in the form of missing projections from high angles, which present the electron beams with higher effective thickness. Since electrons do not image well after passing through thick regions, the projections from these angles are nonexistent. Therefore the reconstruction is conducted on an incomplete set of components, creating numerous artifacts. These challenges will be explained in more detail.

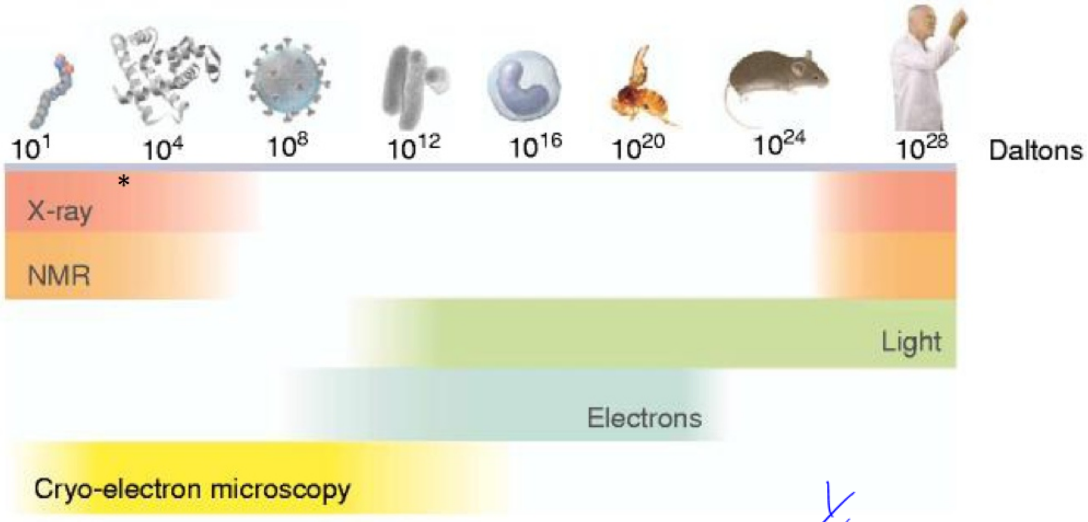
One increasingly used approach to overcoming these challenges is to acquire multiple tomograms of the same underlying object, and average them very carefully to increase the SNR and compensate for the missing projections. This approach requires multiple acquisitions of the same underlying object, and therefore puts additional pressure on the CET pipeline to have high throughput. The goal in this work is to use our knowledge of computer vision, probabilistic graphical models, and inference to help achieve this ~~badly needed~~ high throughput post-processing pipeline. In this thesis, we leverage recent results from computer vision which make use of geometric and spatial context to overcome this uncertainty in a probabilistic framework. Using this approach, we then demonstrate the automation of two particularly time consuming tasks in this pipeline: 2D image registration, and 3D cell boundary segmention.

1.2 Thesis Contributions and Outline

This thesis makes the following contributions towards the goal of achieving high throughput electron tomography:

1-The incorporation of geometric and spatial context for robust object and pattern recognition in electron tomography

Electron tomography datasets have very low quality and therefore weak local features. This makes automatic queries on the images such as feature tracking and object recognition difficult. In our framework we incorporate geometric and spatial context of various types and degrees of locality to strengthen these local features. In



You should reference this figure, if you did not draw it.

Figure 1.2: The biomedical imaging spectrum illustrating the ”imaging gap”

some cases, geometric context is integrated at a low level as part of the local feature itself.

2-The introduction of structured probabilistic models to exploit the above context and local information in an integrated and efficient way

The language and framework in which the geometric and spatial context is incorporated along with the local features is that of structured probabilistic models. Specifically, we employ conditional random fields (CRF's) to describe and relate the relevant variables in our queries of these images. Structured probabilistic models offer two main advantages. Firstly, they provide a flexible means to specify and describe a diverse variety of large complex problems with many variables in terms of many smaller more manageable components which are interrelated. Secondly, they provide a wealth of approximate yet efficient inference algorithms to perform queries on a few variables while combining information from all other variables in the problem. This concept is ideal for exploiting context.

3-Robust correspondence, tracking, and precision image registration using pattern context and conditional random fields

Precision image registration in electron tomography has required the detection, correspondence, and tracking of point features across images. Past approaches to correspondence have attempted this by tracking single features at a time with limited success. Using the context of patterns formed by groups of points in a CRF framework, we achieve robust correspondence of these features, and hence successful tracking and automatic precision registration of images in electron tomography datasets.

4-Holistic 3D boundary detection of cells using novel features based on global shape, local orientation, and nonlocally learned texture

Detection of whole cell boundaries in 3D electron tomography volumes has been difficult due to weak local features and poor discriminative power of intensity and intensity gradient based methods. We have created novel features based on nonlocally learned texture, global shape, and local orientation, and combined them with global shape information in a CRF framework. By performing maximum a posteriori (MAP) inference on this model, we have successfully discovered and detected whole cell boundaries automatically, reducing processing times from days to hours.

I think of these as proper names so it should be Cap Chapter 2, while saying this chapter is not cap. Since you cap Fig 1 you need to cap this too

The rest of this thesis is organized as follows:

In chapter 2, we provide an overview of electron tomography and its pipeline. Specifically, we describe the process of acquiring images with the electron microscope from multiple projections. We then continue with the processing of those images to get 3D volume renditions of cells and subcellular structures, as well as their analysis.

In chapter 3, we provide a brief review of inference with graphical models, and describe how conditional independence is exploited in these models to reduce the complexity of representation. We then briefly describe two relevant inference algorithms to this work, and pose shape and spatial context as latent hidden variables to be jointly inferred with local features within this framework.

Chapter 4 describes the first problem in the CET pipeline to be addressed with the above framework- the precision alignment and registration of 2D images in preparation for the best possible 3D reconstruction. We describe how the tracking and correspondence of features in these images can be achieved by considering their mutual spatial relationships. These relationships along with physical appearance scores of features are encoded into a probabilistic graphical model (a CRF), and inferred efficiently using loopy belief propagation (LBP) to achieve successful tracking and alignment. This work is based on the article [AMC$^+$07].

Chapter 5 goes on to describe another problem in this pipeline also addressed with a similar framework- the 3D segmentation of whole cells from electron tomography volumes. In this case, we show the geometric context of use to be shape, a context that has a continuous parameterization. We also learn the nature of the appearance feature nonlocally throughout the volume. We then jointly encode the shape with the learned features into another CRF. We finally infer the most likely shape incrementally in a prediction measurement update based coordinate ascent on the CRF. This work is based on the article [MHA$^+$09].

We conclude this work in chapter 6.

Don't really need this —
it doesn't say much.

Chapter 2

Cryogenic Electron Tomography (CET)

Electron Tomography is the process of taking multiple electron micrographs of a biological sample (usually a whole cell or subcellular structure) from different angles, and tomographically reconstructing a 3D image of the sample [KK03, Bau02]. In cryogenic electron tomography (cryo-ET), the sample is first flash frozen to retain the biological and molecular structure as much as possible, enabling the study of macromolecular cell features such as cell membrane, surface layer (S-layer) components, ribosomes, filaments, and cytoskeletal structures [GCJ08, OFK+06] in their native natural environment. This process is depicted graphically in Fig. 2.1.

2.1 Image Formation in the Electron Microscope

-Image formation in the electron microscope - by phase contrast- electron scattering, need to limit thickness

Need to be written?

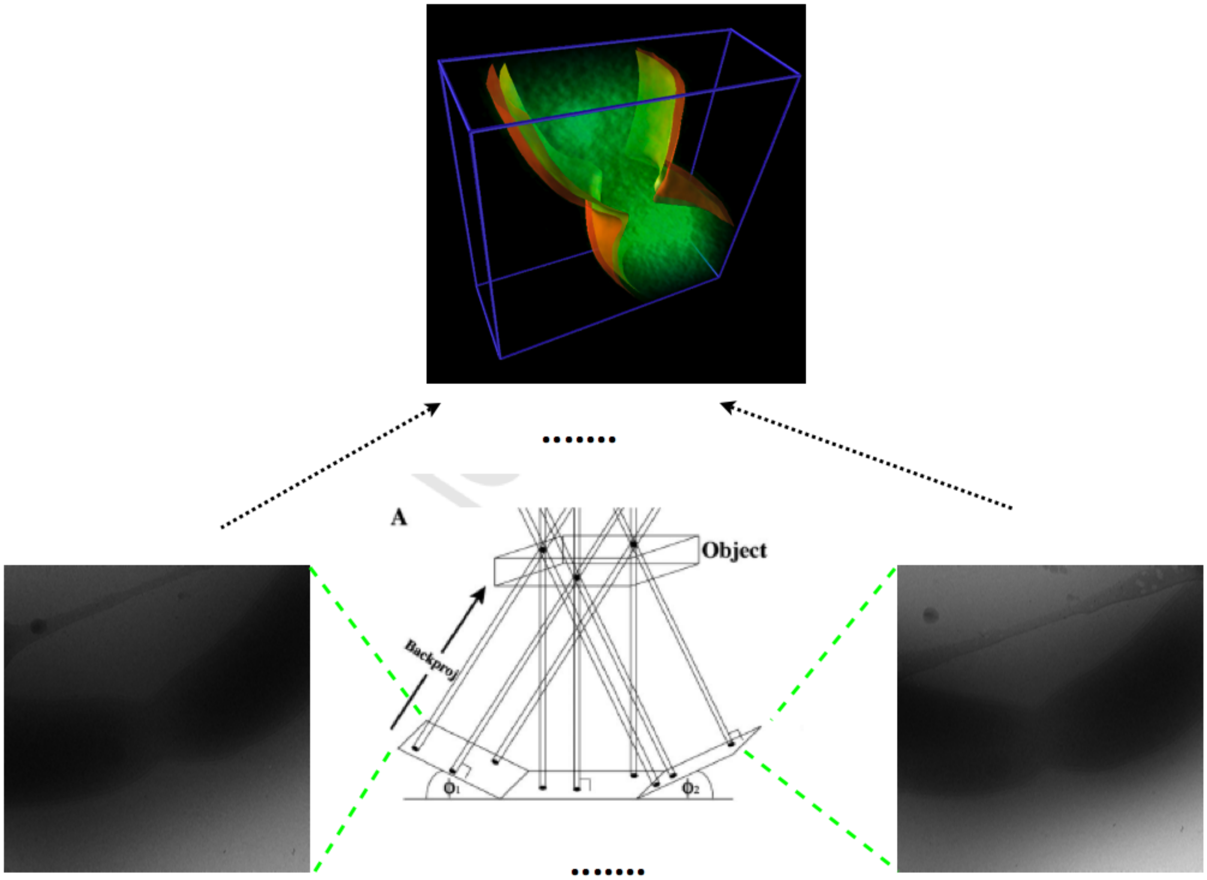


Figure 2.1: Process of electron tomography.

2.2 Tomographic 3D Reconstruction ~~from 2D Images~~

Not needed

Tomography is the estimation of a volume's interior from its projections. By rotating a 3D object to different angles and imaging the integrated effect of the density along the path, its 2D projections, one can use those projections to reconstruct the density map of the subject. We will now mathematically describe both the projection and reconstruction of an object. Our definitions will be on 2D objects with 1D projections, but the same concept can be readily extended to 3D objects with 2D projections.

2.2.1 The Forward Problem- Projection as Radon Transform

We start with the process of taking projections. Consider a 2D sample that has been rotated to an angle ϕ in a coordinate system described by coordinates x_1 and x_2. The sample has a density function $u(x_1, x_2)$. We then take a projection of this sample onto a 1D line by taking parallel beams orthogonal to this line, and integrate the density of the sample along these beams. Each beam has an offset ρ with respect to the 1D line, and is called $L_{\rho,\phi}$. This situation is depicted graphically in Fig. 2.2. The value seen at each point ρ on the projection line is the line integral of the density along $L_{\rho,\phi}$:

$$R(\rho, \phi) = \int\int_{L_{\rho,\phi}} u(x_1, x_2)dx_1dx_2 \tag{2.1}$$

However, the line $L_{\rho,\phi}$ can be described by the equation:

$$\rho = x_1cos(\phi) \overset{\underline{?}}{\boxed{+}} x_2sin(\phi) \tag{2.2}$$

Therefore Eq. 2.1 can be written as

then this is correct

$$R(\rho, \phi) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} u(x_1, x_2)\delta(\rho - x_1cos\phi + x_2sin\phi)dx_1dx_2 \tag{2.3}$$

This relation is known as the *Radon transform* of $u(x_1, x_2)$, and transforms a function in real space (x_1, x_2) to the space of lines (also known as Radon space) (ρ, ϕ). A line in real space will transform to a point in Radon space. Also, a point in real space will transform to a sinusoid in Radon space.

We can now think of each projection of angle ϕ of the sample as the Radon transform of the sample for a fixed value of ϕ. Therefore the collection of all projections of the sample form a sampled version of the entire Radon transform of the sample.

2.2.2 The Inverse Problem- Reconstructing from Projections as the Inverse Radon Transform

We would like now to estimate the original sample's density function given its projections, i.e. the measured samples of its Radon transform. We start by taking the 1D Fourier transform of one of the projections with respect to ρ:

$$S_\phi(r) = \mathscr{F}_\rho R(\rho,\phi) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} u(x_1,x_2)\delta(\rho - x_1 cos\phi + x_2 sin\phi)e^{-i2\pi\rho r}dx_1dx_2d\rho \tag{2.4}$$

where r is the dual variable for the variable transform. If we introduce two new variables

$$k_1 = rcos\phi \tag{2.5}$$

$$k_2 = rsin\phi \tag{2.6}$$

we can see that

$$S_\phi(k_1,k_2) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} u(x_1,x_2)e^{-i2\pi(k_1x_1+k_2x_2)}dx_1dx_2 \tag{2.7}$$

which is simply the 2D Fourier transform of $u(x_1,x_2)$. However, k_1 and k_2 are restricted in this formulation through Eq. 2.6 to be on one line which passes through the origin with angle ϕ in Fourier space. Therefore the 1D Fourier transform of the projection from angle ϕ of $u(x_1,x_2)$ is a slice of the 2D Fourier transform of the entire $u(x_1,x_2)$ which passes through the origin of the Fourier plane with angle ϕ. This fact is known as the *Fourier slice theorem.*

It is possible to reconstruct the function $u(x_1,x_2)$ using the Fourier slice theorem. By taking the Fourier transform of each projection $S_\phi(r)$, we can assemble a sampled version of the Fourier transform of the entire function, and then invert it. It can be shown that

$$\hat{u}(x_1, x_2) = \int_0^{\pi} \int_{-\infty}^{+\infty} S_\phi(r)|r|e^{i2\pi rt} dr d\phi|_{t=x_1 cos\phi + x_2 sin\phi} \tag{2.8}$$

[illegible]?

The inner integral in Eq. 2.8 is the inverse Fourier transform of $S_\phi(r)$ filtered by a ramp filter $|r|$ evaluated at $t = x_1 cos(\phi) + x_2 sin(\phi)$. If we take a fixed point (x_1, x_2), there corresponds a value t for a fixed angle ϕ. The contribution of this integral to the reconstruction will be constant for all points (x_1, x_2) along the line $t = x_1 cos\phi + x_2 sin\phi$. For this reason this value is considered t be backprojected evenly onto this line, and this process is commonly referred to as *filtered backprojection*. The value of the recovered density $\hat{u}(x_1, x_2)$ at each point (x_1, x_2) is the sum of all contributions of this inner integral for all values of ϕ.

This has been a brief description of the vast subject of tomographic reconstruction from projections using the Radon transform. While the derivation was for projection of a 2D object onto one dimensional lines, it is readily extended to the case of projecting 3D objects onto 2D planes. For more details the reader is referred to many excellent references ([KS01, Nat01, Her95, Bra95, Rad17, Rad86]).

2.3 The Electron Tomography Pipeline and Challenges

In order to avoid damage to the sample, the microscopist must observe a limitation on total dose applied. In the case of cryogenic electron tomography, this dose is smaller than in other forms of electron tomography (get numbers). An example of a radiation damaged sample is shown in Fig. 2.3. As a result, the total signal to noise ratio in each projection as well as in the final reconstruction will be very low. An example of a single projection is shown in Fig. 2.4(a), and a single slice from a tomographic reconstruction is shown in Fig. 2.4(b).

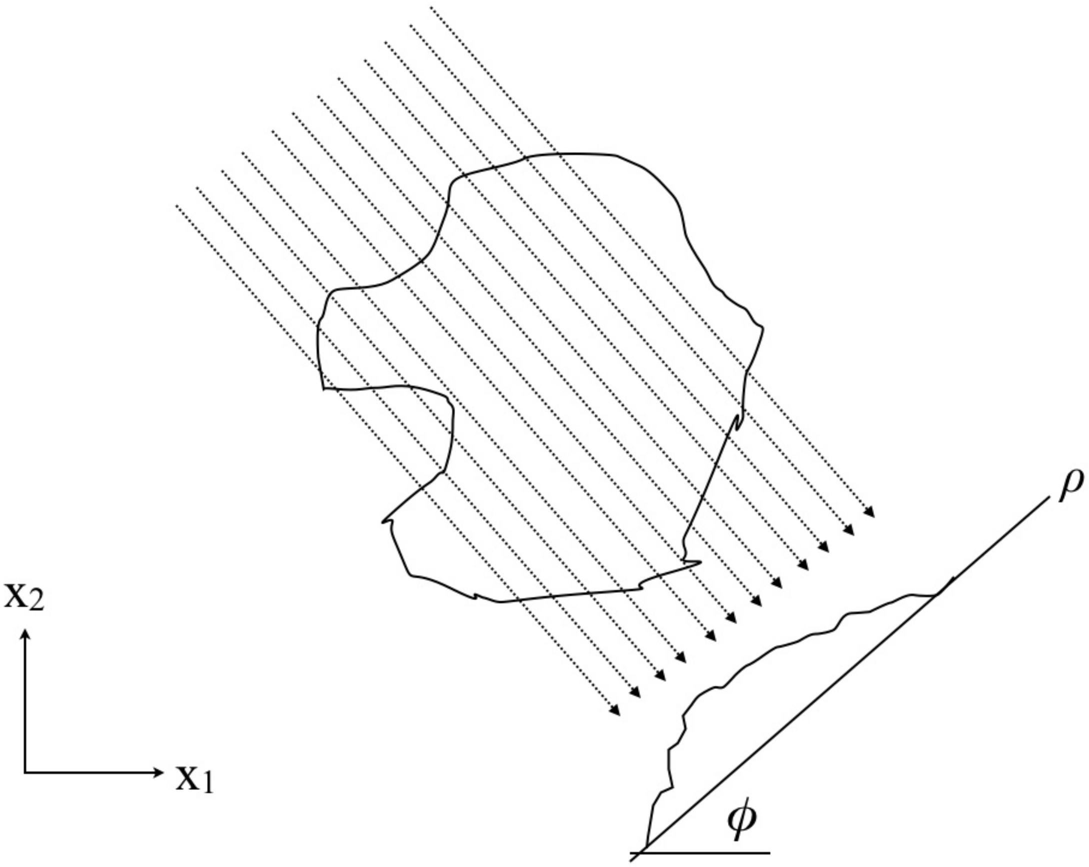


Figure 2.2: Projecting a 2D object onto a 1D line at an angle ϕ.

No it is not a result of scattering

Image formation in the electron microscope is a result of electron scattering, which causes phase contrast. As electrons travel through thick samples, they scatter multiple times, resulting in poor images. Therefore, A as the sample gets rotated to high tilt angles, the effective thickness experienced by the electrons ~~is too high~~ becomes larger, and projections of the sample are not usable. Typically, this limits the angular range of the acquisition to $+/-60 degrees$. Since tomography requires a full range acquisition of $+/-90 degrees$ for a faithful reconstruction, electron tomographic reconstructions suffer from artifacts due to missing projections, which directly translate to corresponding missing regions in Fourier space due to the Fourier slice theorem. ~~Therefore~~ These artifacts are known as the "missing wedge" effect. An example of

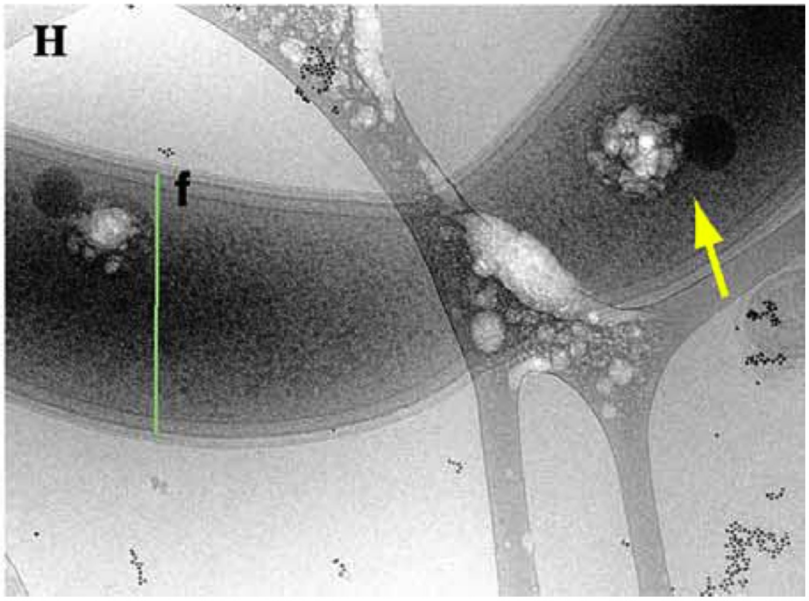


Figure 2.3: Radiation damage incurred by a Caulobacter crescentus at liquid nitrogen temperatures when exposed to 21,800 e^{-}/nm^{2} ([CD05]).

the missing wedge effect is shown for a normal image and an electron tomographic reconstruction in Fig. 2.5. The main consequence of noise and artifacts in these images is the lack of reliable low level local features that would be typically used for higher level recognition tasks. This will be discussed in more detail in section 3.1.

2.4 Overcoming Image Challenges: the Need for High Throughput

To fulfill the promise of 3D high resolution imaging of cells and subcellular structures, it is necessary to overcome the uncertainty from noise and artifacts. This can be done by imaging many instances of the same underlying object, then registering and averaging them. If these instances are at random orientations, and the noise in

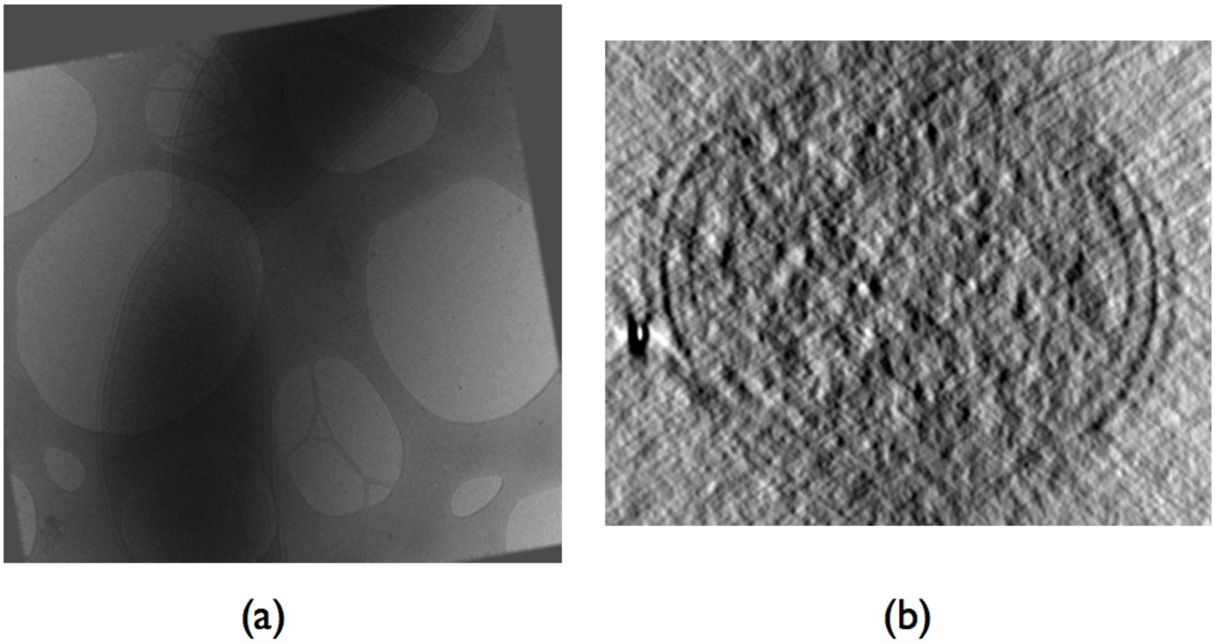


Figure 2.4: Single projection of a whole cell (a) and single slice of a reconstructed tomogram (b)

each instance is independent from the noise in other instances, averaging large numbers of these instances can produce higher quality renditions of the objects. This process is called subtomogram averaging and has been an active area of research by the electron tomography community ([BSL$^+$08, SMVC09]). Two examples are shown in Figs. 2.6 and 2.7 . The first example involves the study of the flagellar motor ([MLJ06]). Several individual images of this complex molecular machinery, are shown in Fig. 2.6 (a). Several hundred instances of this motor are averaged manually exploiting its seven way symmetry. The result is shown in Fig. 2.6 (b), which shows significant structural detail at the molecular level. The second example involves the surface layer (SLayer) of the Caulobacter crescentus bacterium, which is known to have a periodic macromolecular structure. Small individual patches of this layer obtained from the tomogram, are shown in Fig. 2.7, suggest a periodic pattern, but little other detail of the molecular structure. However, averaging several thousand such patches with an automatic procedure described in ([ACM+10]) reveals a much more informative result, shown in Fig. 2.7 (b).

As was mentioned previously

To facilitate this higher resolution average images ~~process,~~ large amounts of data must be processed, requiring a high throughput pipeline. The electron tomography pipeline is shown in Fig. 2.8. It can be divided into two major parts- the acquisition of images, and the postprocessing of images.

The acquisition part of the pipeline involves sample preparation and images series acquisition. Samples are obtained by growing populations of cells on carbon grids, and then prepared usually by either chemically treating them and embedding them in resin (plastic sections) or by flash freezing the sample to liquid nitrogen temperatures (cryogenic electron tomography). In both cases, the preparation aims to keep the sample rigid and to make it tolerate a significant amount of electron dose before sustaining damage. Once the sample is prepared, it is inserted into a special holder which is then loaded onto the electron microscope stage. The series of images is then acquired sequentially. For each image in the series, the microscope stage is carefully rotated to a certain angle (called the tilt angle), and the projection image for that tilt angle is then acquired. When planning an image series acquisition, the microscopist must plan for the set of tilt angles and the dosage budget for each projection.

really? I thought they were grown on other media + placed on the grids

The postprocessing part of the pipeline takes the raw set of images from the image series acquisition and gives out the final data and results. The first step in this process is to precisely align the images in the series, since the mechanical stage is not capable of controlling the position of the holder precisely enough to avoid misalignment. The aligned image series is then used to obtain a 3D reconstruction with tomography as described in section 2.2 to obtain a 3D volume representation of the sample. The volume is then analyzed to obtained the various subjects of study through segmentation and recognition. Finally, the identified subvolumes are analyzed for specific attributes, and then averaged in cases of subtomogram averaging.

In recent years, a number of advances in microscope automation have given the acquisition portion of the pipeline high throughput capability. ~~Pipeline~~ T throughputs of several hundred tomograms per month for a lab are now ~~a reality and~~ not uncommon. With a implies each tomogram having a size of a few gigabytes, it is not practical to require any significant manual intervention of the postprocessing of these tomograms. Therefore the bottlenecks in realizing a high throughput electron tomography pipeline lie in the postprocessing section of the pipeline.

Why does size matter?

A common reason that various tasks in the postprocessing require manual intervention is the need for ~~inference and~~ to recogni~~tion~~ze features in images containing un~~reliable~~ very noisy data lo~~cal features~~ as discussed in section 2.3. For example if we believe a boundary to be a connected set of edge pixels and the edges are weak in a local region, that boundary will appear as weak and incoherent, as shown in Fig. 2.9 (a). However, by exploring the context of the surrounding area of that local region in the original image, it is possible to search for and acquire the cell boundary (b). Human beings are very good at inferring that the boundaries by exploiting context in the image. In order to overcome the uncertainty in t~~hese~~ the local features, we seek to employ the same intuition as humans do and apply context wherever possible. In the next section, we will explore concrete ways to encode context, and to use this encoded context to make efficient inferences.

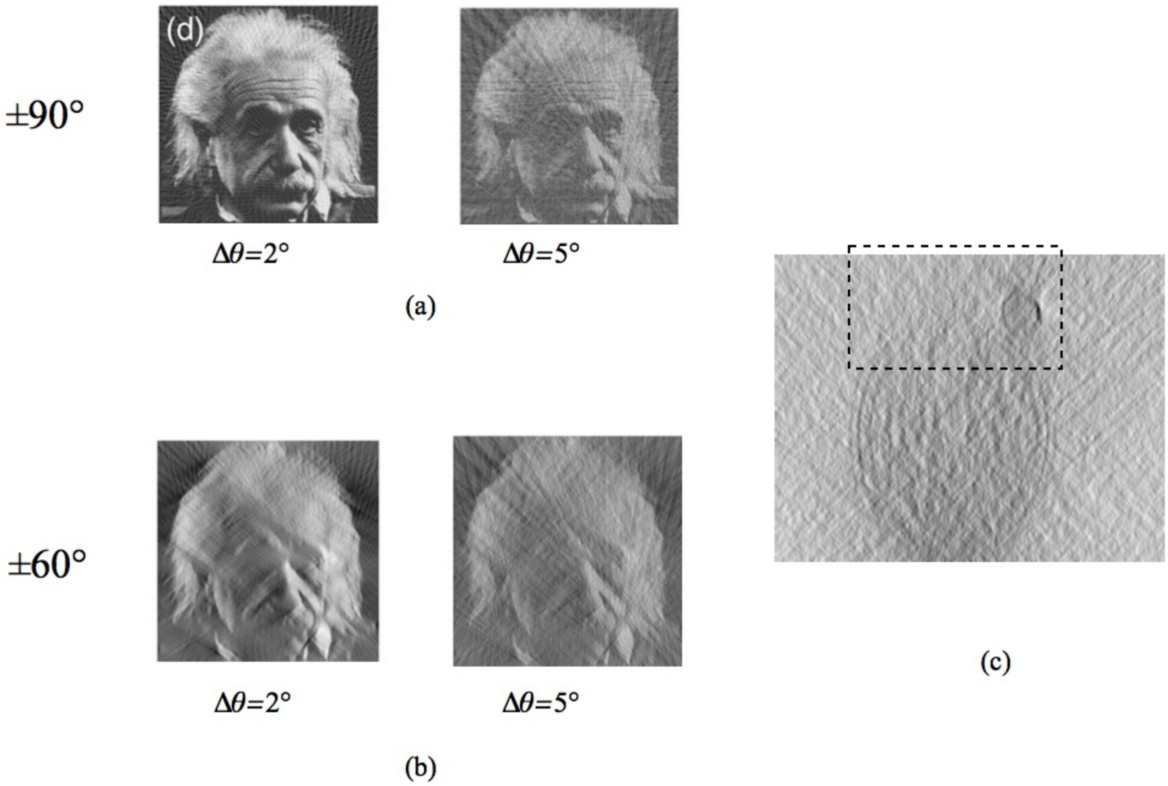


Figure 2.5: (a)Normal reconstruction of test image of Einstein, using angular spacings of 2 and 5 degrees, (b)Reconstruction of Einstein test image with missing wedge, using limited angular range of 60 degrees, and (c)slice of reconstructed tomogram of a Caulobacter crescentus cell with missing wedge- note missing horizontal features.

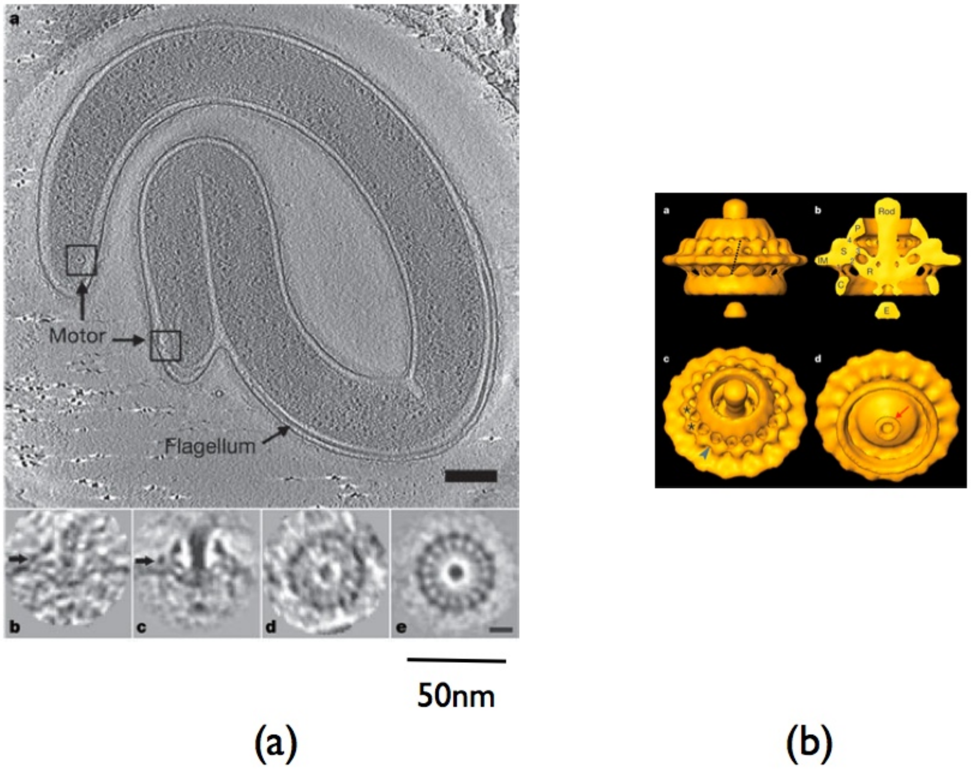


Figure 2.6: Flagellar motor viewing through averaging ([MLJ06])

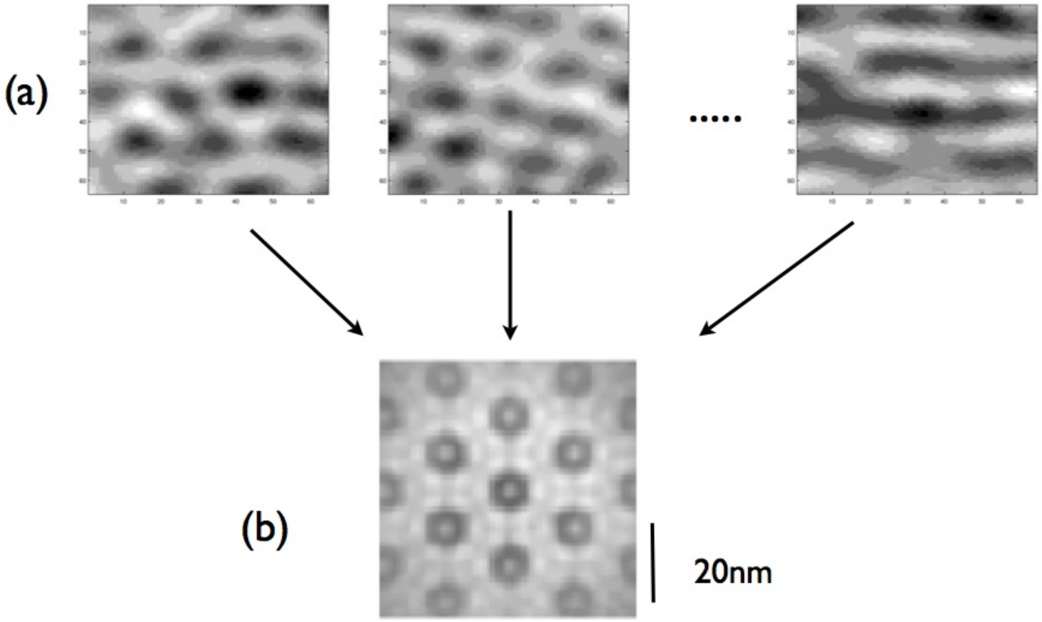


Figure 2.7: Caulobacter surface layer (SLayer) viewing through averaging ([ACM+10])

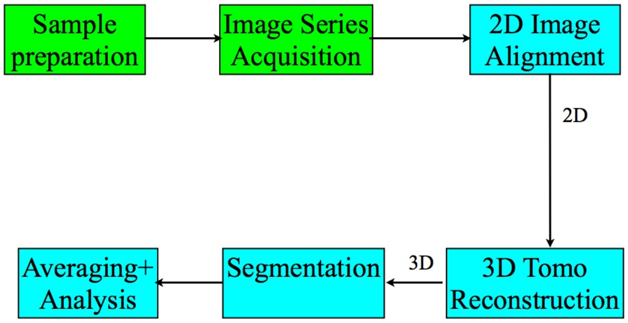
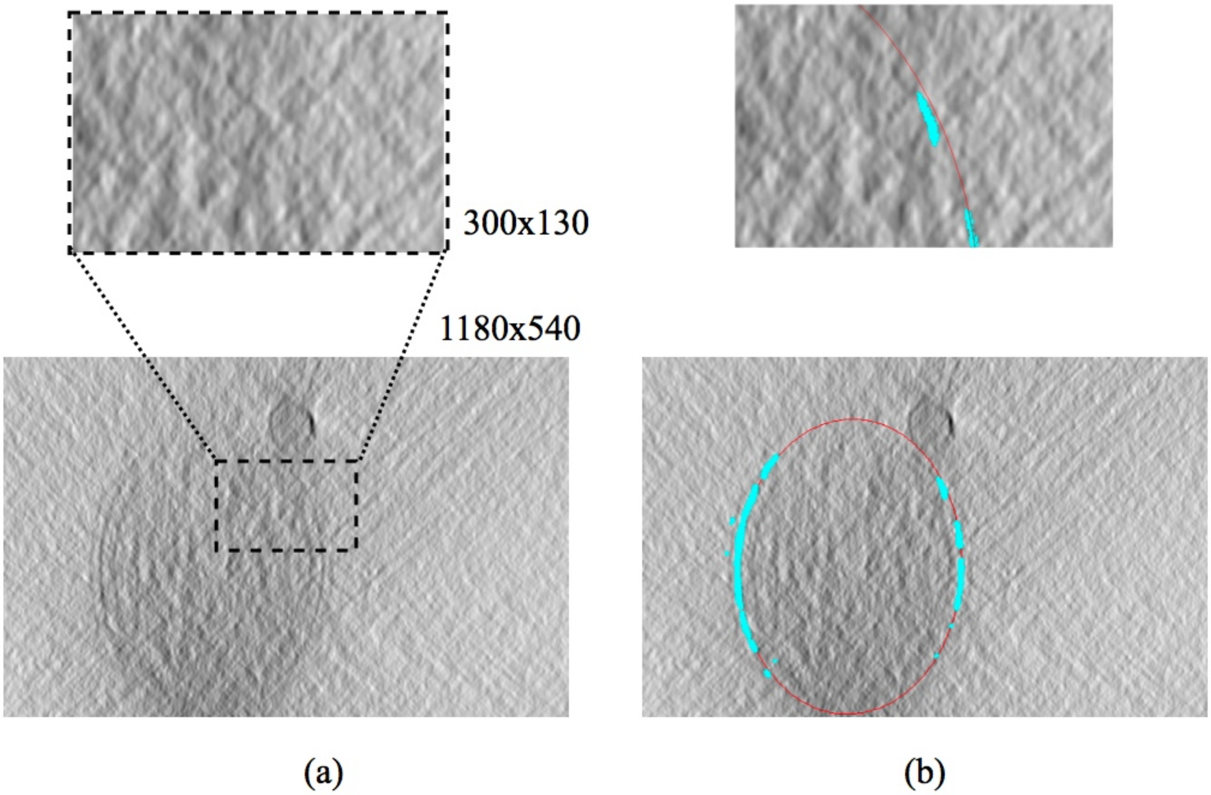


Figure 2.8: Electron tomography pipeline

Shouldn't be a full page image



An example of

Figure 2.9: Uninformative local regions in electron tomography. It is difficult to find the cell boundary using only the local region shown in (a). By looking at the context ~~of the~~ surrounding ~~of~~ that region in the original Image, it is possible to successfully find the cell boundary (b).

image since it is not a name

Chapter 3

Modeling Geometric Context with Graphical Models

3.1 Local Features and Geometric Relationships

Computer vision is a highly developed and rapidly advancing field. A common approach to scene understanding in modern computer vision is to start from obtaining local features, and then feed them to increasingly higher level understanding tasks to reach an overall understanding of the image. This approach is known as the *bottom up* approach in computer vision.

3.1.1 Local Features - Bottom Up vs. Top Down Approach

Local features are image characteristics that are extracted from local patches of the images. A simple example of local feature detection in images is edge detection, which uses intensity gradients to detect boundaries in a local region of an image. In natural images with high enough quality, edge detection can sometimes be used to create a processed image that is sufficiently informative for higher level queries. An example of this is shown in a photograph of an aircraft in Fig. 3.1 (a) and (b). The output of edge detection captures the outline of an aircraft and other significant objects in the image, and serves as a good starting point for higher level queries such as segmentation and object classification.

In more typical cases (and indeed in most cases of electron tomography images), however, local features are not very informative in and of themselves. Applying the same edge detection to a slice of an electron tomogram results in many false responses (Fig. 3.1 (c) and (d)). Therefore the output image is not very informative or useful for any subsequent recognition task, making a purely bottom up approach infeasible. In such cases, it is helpful to incorporate any available higher level information about the object or geometry to be inferred in order to guide the local feature detection. Such an approach is known as a *top down* approach. A simple example of higher level information is the numbers of each type of feature found in a candidate object, e.g. number of legs and ears and tails detected when trying to find an animal. An approach which uses the number of features without regard for their arrangement is called a "bag of features" approach. Bag of features has been used widely and successfully for a large number of computer vision tasks in natural images.

3.1.2 Geometric Context

Another example of high level information in an object is the arrangement of its member features. In the animal detection case, knowing that the head is at the opposite end of the tail, or the legs are at the bottom of the animal can be valuable cues for inference. In the case of cell boundary detection, (e.g. the case of Fig. 3.1 (c) and (d)), it would help to know that the edges we seek lie in an elliptical region, and have certain orientations which are consistent with some underlying ellipse. Such underlying geometric information is considered high level information, and the constraints associated with this information are geometric ones.

More robust recognition and inference in images is enabled by combining information from the bottom up approach and the top down approach, thus integrating high and low level information in an image. Recent work in the computer vision community has leveraged this notion to achieve better recognition in natural images. Indeed, recent works in object detection by [FFJS08, OPZ08] have used local shape formed by feature points in addition to the features' appearances themselves to achieve better overall detection. Most recently, [HEPK09] used learned global shape and appearance in a probabilistic model to localize landmarks and outlines of objects of a particular class. It is this line of work that inspires ~~the basis for~~ our approach to achieving better inference in the electron tomography pipeline.

3.2 Graphical Models

We seek a suitable framework to formally encode the necessary information in order to facilitate combining top down and bottom up approaches and achieve robust inference in images. Such a framework must be flexible enough to capture and encode a wide and diverse types of context, such as high level geometric relationships or nonlocal low level appearance information. It must also lend itself to efficient optimization methods. A framework which satisfies these requirements is that of graphical models.

3.2.1 Representation

Graphical models, or structured probabilistic models, are a vast subject with many excellent references including ([KF09, Bis06]), and we will provide only a brief overview of the main concepts here. In graphical models, individual variables are depicted as nodes in a graph. Edges are then inserted between nodes (variables) only if they are directly dependent on one another. The edges can be directed or undirected. A directed edge from one variable to another indicates a conditional probability distribution (CPD) of the child variable given the parent variable. Undirected edges indicate an unnormalized compatibility function between variables in a fully connected subgraph. If all the edges in the graph are directed and the graph is acyclic, the graph is a *Bayesian network*. If all the edges in the graph are undirected, the graph is a *Markov network*. In either case, there are clear statements of conditional independence that can be made about variables given other variables. In the case of Markov networks (and with some exceptions in Bayesian networks), variables are conditionally independent of all other variables given their neighbors. Therefore, the joint distribution over all variables factorizes into a product of many smaller terms over few variables. In a Bayesian network, the factors are CPD's, and in a Markov network, the factors are compatibility functions. The set of variables found in the argument of each factor is called the *scope* of that factor. In this work, we are primarily concerned with Markov networks.

An example of a Markov network is shown in Fig. 3.2. This example has M variables $X_i, i = 1...M$. The joint distribution over these variables factorizes into smaller terms.

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \prod_c \phi_c(\mathbf{X}_C = \mathbf{x}_c) \tag{3.1}$$

really shouldn't be indented

where Z is a normalizing constant called the *partition function*, and $\phi_c(\mathbf{X}_C = \mathbf{x}_c)$ are factors over a fully connected subgraph, known as a *clique* denoted by C. The set of variables in clique C are denoted by $\mathbf{X}_C$. A *maximal clique* is a clique that is not a subset of any other clique. A factor is not required to be over a maximal clique.

For example one such factorization could be:

$$P(X_1, ...X_i, ...X_M = x_1, ..x_i, ...x_M) = \frac{1}{Z} \prod_i \phi_i^s(X_i = x_i) \prod_{j \subset N(i), j>i} \phi_{i,j}^p(X_i = x_i, X_j = x_j) \tag{3.2}$$

where $N(i)$ refers to neighbors of node i in the Markov network, and the $j > i$ condition is to avoid double counting of factors $(\phi_{i,j}^p(X_i, X_j) = \phi_{j,i}^p(X_j, X_i))$. Here we chose cliques with maximum size of 2 variables.

Various factorizations are possible in Markov networks, as factors are not required to be over a maximal clique. For example in Fig. 3.3 there are 3 variables, 3 pairwise cliques, and 1 triplet clique (which is the maximal clique). Therefore both of the following factorizations may apply to this graph:

$$\phi_{C_{12}}(X_1 = x_1, X_2 = x_2)\phi_{C_{23}}(X_2 = x_2, X_3 = x_3)\phi_{C_{13}}(X_1 = x_1, X_3 = x_3) \tag{3.3}$$

$$\phi_{C_{123}}(X_1 = x_1, X_2 = x_2, X_3 = x_3) \tag{3.4}$$

Both of these factorizations satisfy the same conditional independence relationships, but the first one has fewer degrees of freedom than the second, and can encode a smaller space of possible distributions over these 3 variables. However, as we shall see, it requires less complexity when performing queries on its variables. This performance complexity tradeoff is commonly encountered when using probabilistic graphical models.

A representation which makes the factorization explicit is the *factor graph.* A factor graph is a bipartite graph which contains *factor nodes* in addition to variable nodes. Each factor node represents one factor $\phi_C(\mathbf{X}_C)$, and connects to the variables in $\mathbf{X}_C$, the scope of that factor. Two distinct factor graphs which represent the factorizations in Eq. 3.3 and Eq. 3.4 are shown in Fig. 3.4 (a) and (b) respectively.

A graph which encodes $P(\mathbf{X})$, a joint distribution over a set of variables $\mathbf{X}$, is called a *generative* model, as it can be used to generate samples for any subset of the variables in $\mathbf{X}$. It is often desirable to encode a conditional distribution $P(\mathbf{X}|\mathbf{Y})$, where $\mathbf{Y}$ is a set of *observed features* that have fixed values and are disjoint from $\mathbf{X}$. We prefer to treat $\mathbf{Y}$ as observed features rather than variables with a distribution, since this distribution may be complex and poorly understood. This situation is not uncommon for image features used as observations. A model which encodes such a distribution is called a *discriminative* model, and cannot be used to generate samples for the evidence variables. A Markov network that models this relationship is called a *conditional random field*, or CRF ([LMP01]). The CRF nodes are $\mathbf{X} \cup \mathbf{Y}$, and $\mathbf{Y}$ are shown grayed out to emphasize that they are fixed observations and not variables. An example CRF is shown in Fig. 3.5. The main difference with a figure placement is bad

normal Markov netowrk is that a factor's scope must contain variables only from $\mathbf{X}$. The conditional distribution for a CRF is defined by:

$$P(\mathbf{X} = \mathbf{x} | \mathbf{Y} = \mathbf{y}) = \frac{1}{Z(\mathbf{y})} \prod_{c} \phi_c(\mathbf{X}_C = \mathbf{x}_c; \mathbf{y}) \qquad (3.5)$$

where $Z(\mathbf{y})$ is the partition function over $\mathbf{X}$ given the observations $\mathbf{Y} = \mathbf{y}$, and $\phi_c(\mathbf{X}_C = \mathbf{x}_c; \mathbf{y})$ is a compatibility function indicating preference for the assignment $X_c = \mathbf{x}_c$ given $\mathbf{Y} = \mathbf{y}$. This compatibility function is often abbreviated to omit the observation, and is simply written as $\phi_c(\mathbf{X}_C = \mathbf{x}_c)$.

Lastly, we consider a class of graphical models called *temporal models*, in which a common subgraph gets repeated over time and unfolds to yield an extended unfolded graph. When the subgraph is a CRF, the resulting graph is called a *linear chain CRF* ([KF09, Bis06]). A linear chain CRF is shown in Fig. 3.6. In general, temporal models can repeat over time indefinitely. However, in this work, we will work with temporal models with finite length.

No ¶ one sentence

We next explore how to use these representations to make probability queries.

3.3 Estimation with Inference

Our goal is to estimate geometry from images. To encode geometry, one can introduce relevant variables such as shape parameters, or locations of point features in the image, and/or physical appearance variables. By specifying local relationships between these variables, we can construct a conditional distribution over all the unknown variables given the evidence variables in the images. This model now has a set of observed (evidence) variables $\mathbf{E}$ as well as hidden variables $\mathbf{X}$, and we may be interested in a subset of the hidden variables, such as shape parameters. We can then perform various probability queries to estimate the desired geometry variables. Such queries are referred to as inference.

We consider two types of queries: *conditional probability queries* and *maximum* *a posteriori* or *MAP* queries. In a conditional probability query we seek the conditional probability distribution

$$P(\mathbf{Y}|\mathbf{E} = \mathbf{e}) \tag{3.6}$$

where $\mathbf{Y} \subset \mathbf{X}$ are query variables, and $\mathbf{E} = \mathbf{e}$ are the observed values for evidence variables $\mathbf{E}$. In a MAP query, we seek the most likely joint assignment to *all* the unknown variables $\mathbf{X}$ given the observations $\mathbf{E} = \mathbf{e}$:

$$MAP(\mathbf{X}|\mathbf{e}) = \arg\max_{\mathbf{x}} P(\mathbf{X} = \mathbf{x}, \mathbf{E} = \mathbf{e}) \tag{3.7}$$

To perform a conditional probability query on the variables $\mathbf{Y}$, one must calculate the marginal of $\mathbf{Y}$ by summing out the variables in $\mathbf{X}$ that are not in $\mathbf{Y}$. In the worst case, this operation has complexity which is exponential in the number of variables to be summed out and may not be very tractable. However, in many graphs this complexity is much lower due to an amenable graph structure. When the summation can be carried out exactly, this query is an exact inference. When this cannot be done due to excessive complexity, many approximate yet effective inference techniques can be used.

Performing a MAP inference in a Markov network is equivalent to maximizing an energy function, in this case the unnormalized distribution over that graph. Once again, if this maximization can achieve a global maximum (e.g. if the energy function is convex), the inference is exact. If this is not possible or practical, many approximate yet effective optimization techniques can be used to achieve approximate inference.

In this work, we will use two approximate inference techniques. These are loopy belief propagation (LBP) for performing conditional probability queries, and predictive update for MAP inference on a finite duration temporal model.

3.3.1 Loopy Belief Propagation

Belief propagation, or BP, aims to calculate marginal distributions by summing out variables in distributions. It is a form of message passing which allows messages to travel through graphs. Message passing is the process of summing variables locally and combining intermediate results to calculate marginals. The intermediate results are called messages.

Belief propagation is accomplished by iteratively calculating messages locally at each node in a factor graph and propagating them. There are two types of messages: messages from factor nodes to variable nodes and vice versa. These are signified by $m_{f \to i}(X_i)$ and $m_{i \to f}(X_i)$ respectively. We define the *belief on a variable* X_i to be the product of all incoming messages to variable X_i from the factor nodes connected to it:

Probably don't want to use C for both unless I am confused.

what is X_C doing here

$$b_i(X_i) = \prod_{C : X_i \in \mathbf{X}_C} m_{f \to i}(X_i) \tag{3.8}$$

not the same as

Similarly, we define the *belief on a factor* f_C to be the product of the factor itself and all incoming messages to that factor from the variable nodes connected to it:

$$b_C(\mathbf{X}_C) = \phi_C(\mathbf{X}_C) \prod_{i : X_i \in \mathbf{X}_C} m_{i \to f}(X_i) \tag{3.9}$$

Each iteration of belief propagation is a pair of steps carried out at all nodes in which a *sum product algorithm* is executed. Firstly, the messages from all variable nodes to all factor nodes are calculated:

this is C

$$m_{i \to f}(X_i) = \prod_{X_i \in \mathbf{X}_C} b_i(X_i) \tag{3.10}$$

Secondly, the messages from all the factor nodes to all the variable nodes are calculated:

$$m_{f \to i}(X_i) = \sum_{j:X_j \in \mathbf{X}_C \setminus X_i} b_C(\mathbf{X}_C) \tag{3.11}$$

where $\mathbf{X}_C \setminus X_i$ is the scope of the factor excluding X_i, and $b_C(\mathbf{X}_C)$ is obtained from Eq. 3.9. After this step, the belief on X_i ($b_i(X_i)$) can be calculated from Eq. 3.8.

It can be shown that continuing this process on a tree will result in convergence of the beliefs $b_i(X_i)$ to the unnormalized marginal of X_i after a finite number of steps ([YFW05, KF09, KMFaL01, Bis06]). However, in a graph with loops, there is no guarantee of such convergence, and in general there will be some error between the normalized version of $b_i(X_i)$ to the marginal of X_i. Therefore in the presence of graphs with loops, belief propagation is called loopy belief propagation (LBP) and is approximate. In LBP, convergence criteria such as maximum number of iterations and error thresholds must be specified.

3.3.2 Predictive Update for Temporal Models

We now explore inference on temporal models. We first consider the simple linear chain CRF shown in Fig. 3.6. In this model, there is one state variable X_i and one observed variable O_i at each time slice i. We will denote a sequence of variables $X_1 .... X_i$ by $X_{1:i}$. The goal is to estimate the distribution (and ultimately most likely value) of X_i given the observed values of the variables $O_{1:i}$, i.e. $P(X_i|O_{1:i})$. This task is also referred to as tracking.

We consider at each slice i the following two CPD's: $P(X_i|X_{i-1})$ and $P(O_i|X_i)$ (to which we do not have direct access in the linear chain CRF). These are known respectively as the *transition model* and *observation model*. To make the inference scalable with time, it is desirable to estimate $P(X_i|O_{1:i})$ recursively from $P(X_{i-1}|O_{1:i-1})$ and recent observations. We have

$$\begin{aligned} P(X_i|O_{1:i}) =& \frac{P(O_i|X_i, O_{1:i-1})P(X_i|O_{1:i-1})}{P(O_i|O_{1:i-1})} \\ =& \frac{P(O_i|X_i)P(X_i|O_{1:i-1})}{P(O_i|O_{1:i-1})} \end{aligned} \tag{3.12}$$

where $P(O_i|X_i, O_{1:i-1}) = P(O_i|X_i)$ due to conditional independence, and

$$P(X_i|O_{1:i-1}) = \sum_{X_{i-1}} P(X_i|X_{i-1})P(X_{i-1}|O_{1:i-1}) \tag{3.13}$$

and

$$P(O_i|O_{1:i-1}) = \sum_{X_i} P(O_i|X_i)P(X_i|O_{1:i-1}) \tag{3.14}$$

Eq. 3.12, 3.13, and 3.14 are the basis for recursively and incrementally calculating $P(X_i|O_{1:i})$ from $P(X_{i-1}|O_{1:i-1})$. As these equations suggest, the calculation is done in two steps. The first step is the *time update*, or prediction step, which calculates $P(X_i|O_{1:i-1})$ using Eq. 3.13. The second step is the *measurement update*, which calculates $P(X_i|O_{1:i})$ using Eq. 3.12 and 3.14. In effect, the time update step predicts the distribution of X_i from the set of previous observations $O_{1:i}$, and the measurement update step refines that distribution by looking at the most current observation O_i. There are many manifestations of this concept, as well as many excellent references ([KF09, Bis06, AMG02, For73, AM79]), depending on the application and knowledge of the distributions. When the two CPD's are based on linear relationships of the variables with white Gaussian noise, the optimal solution is found exactly and in closed form by the $Kalmanfilter$ ([Kal60, AM79, AMG02, Bis06]). However, quite often such assumptions may not hold, and we may not have direct knowledge of the two distributions. In such cases, time and measurement updates cannot be performed exactly, and suboptimal approximate methods become necessary.

One class of approximate inference methods for temporal models is to perform the time and measurement updates empirically by generating samples using Monte Carlo methods. This approach is known as *particle filtering* ([DDFG01, KF09, Bis06, AMG02]), and is appropriate when we have a way to generate samples from the appropriate CPD's.

Sometimes we may not even have a way to generate samples in the temporal model. This may be indeed the case when we use a linear chain CRF, and our unnormalized compatibility functions do not lend themselves to readily generating samples. Another class of approximate methods that can be used for these cases is to perform an energy maximization using *max product* versions of Eq. 3.12, 3.13, and 3.14. The max product version of these equations replaces the sum operator with the max operator, and puts out the most likely assignment with its probability instead of a distribution over all possible assignments.

Concretely, we would try to maximize the unnormalized distribution up to slice i:

$$\begin{aligned}
E_i &= \prod_{j=1}^{i} \phi_j^t(X_{j-1}, X_j)\phi_j^o(X_j, O_j) \\
P(X_{1:i}, O_{1:i}) &= \frac{1}{Z_i} E_i \\
E_i^{max} &= \max_{x_{1:i}} E_i \\
x_{1:i}^{max} &= \arg\max_{x_{1:i}} E_i \qquad (3.15)
\end{aligned}$$

where Z_i is a normalization constant for slice i, $\phi_j^t(X_{j-1}, X_j)$ and $\phi_j^o(X_j, O_j)$ are the pairwise compatibility functions corresponding to transition and observation model relationships respectively, and $\phi_j^t(X_0, X_1) = 1$ by definition.

We estimate E_i^{max} and $x_{1:i}^{max}$ by solving the following recursion:

$$
\begin{aligned}
E_i^{max} &= \max_{x_i} \phi_j^o(X_i = x_i, O_i = o_i) \max_{x_{i-1}} \phi_i^t(X_{i-1} = x_{i-1}, X_i = x_i) E_{i-1}^{max} \\
E_1^{max} &= 1 \qquad (3.16)
\end{aligned}
$$

This recursion allows an efficient calculation of the MAP estimate. The inner maximization in Eq. 3.16 can be viewed as a time update step, and the outer maximization can be viewed as a measurement update step. Since this procedure for MAP inference is not guaranteed to obtain the global maximum, it is approximate.

While the max product version of time and measurement update is an approximate, suboptimal method, its flexibility and efficiency for use in discriminative temporal models like a linear chain CRF are significant advantages. One can view this method on a linear chain CRF also as a block coordinate ascent method on an energy function. In this view, the energy function is the product of all the compatibility functions up to time i, and a MAP estimation is performed by varying only one group of variables (the scope of a particular factor) at a time. To make this method more optimal, it is possible to go back and vary previously varied variables in a forward backward or other multiple pass strategy.

In this chapter we have established geometric context as a key source of information in our noisy and uncertain image inference problem. We have also shown that a framework based in graphical models, and specifically a conditional random field can be useful to model and exploit this information. We are now ready to apply this intuition and framework to address two specific problems in the ET pipeline: registration of 2D images and segmentation of 3D volumes.

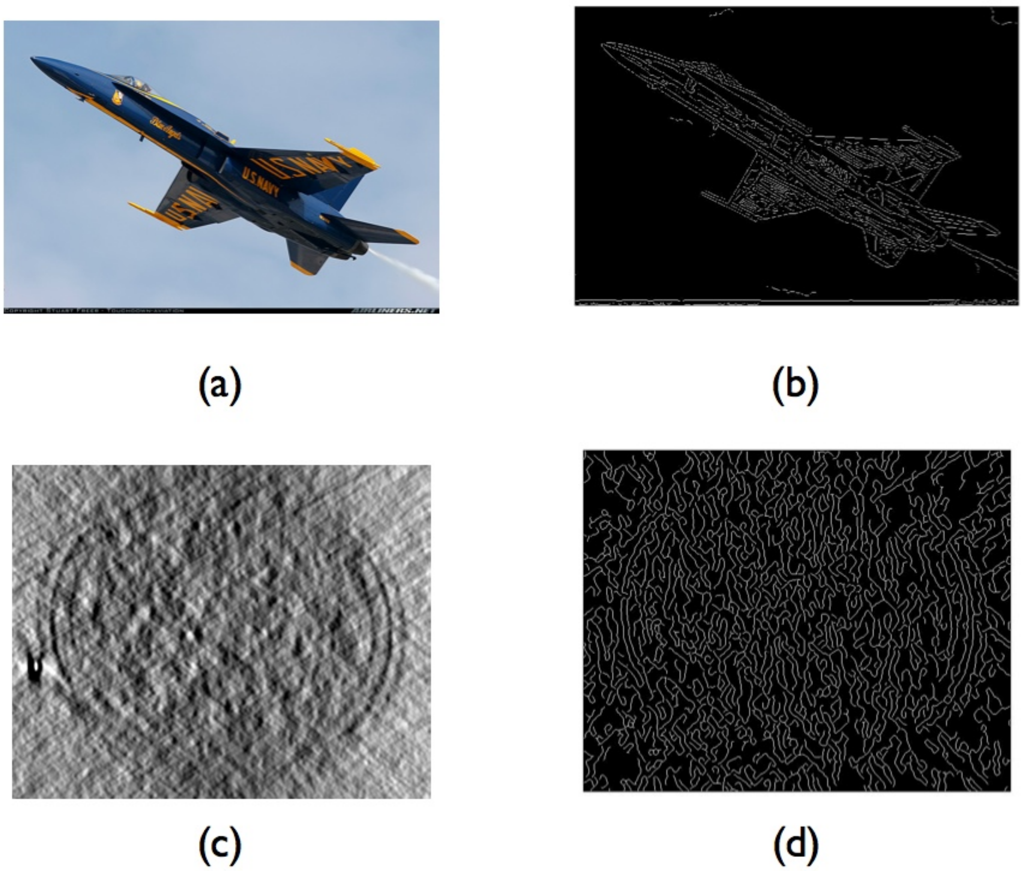


Figure 3.1: Local features in natural images and electron tomography images. A natural image of an aircraft against sky background(a) and its edge detected version (b). A single slice of a reconstructed tomogram (c) and its edge detected version (d).

figs should be next to refs.

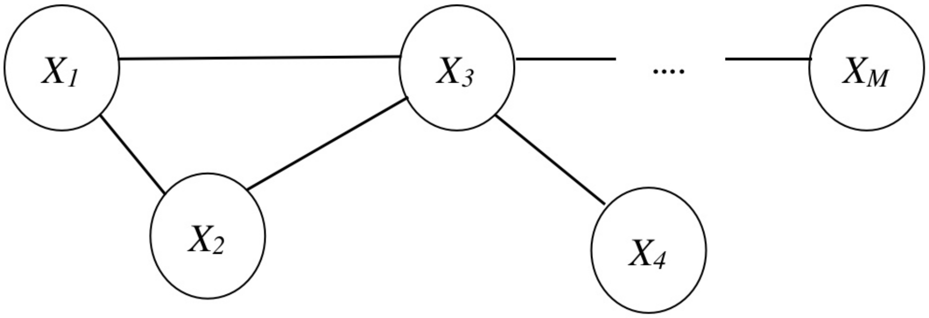


Figure 3.2: A Markov Network.

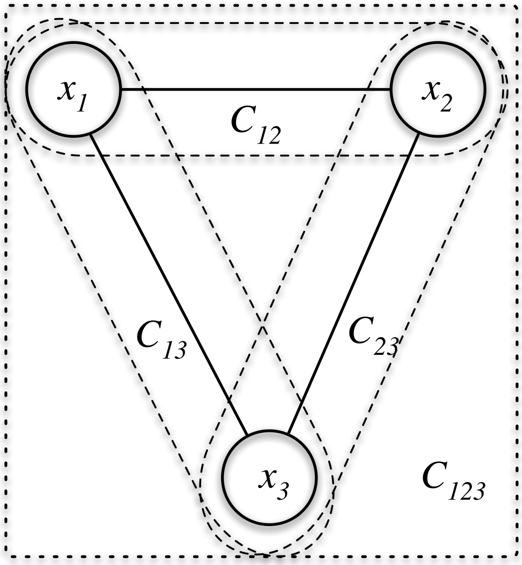


Figure 3.3: A Markov network with 3 variables, 3 pairwise cliques, and 1 triplet clique.

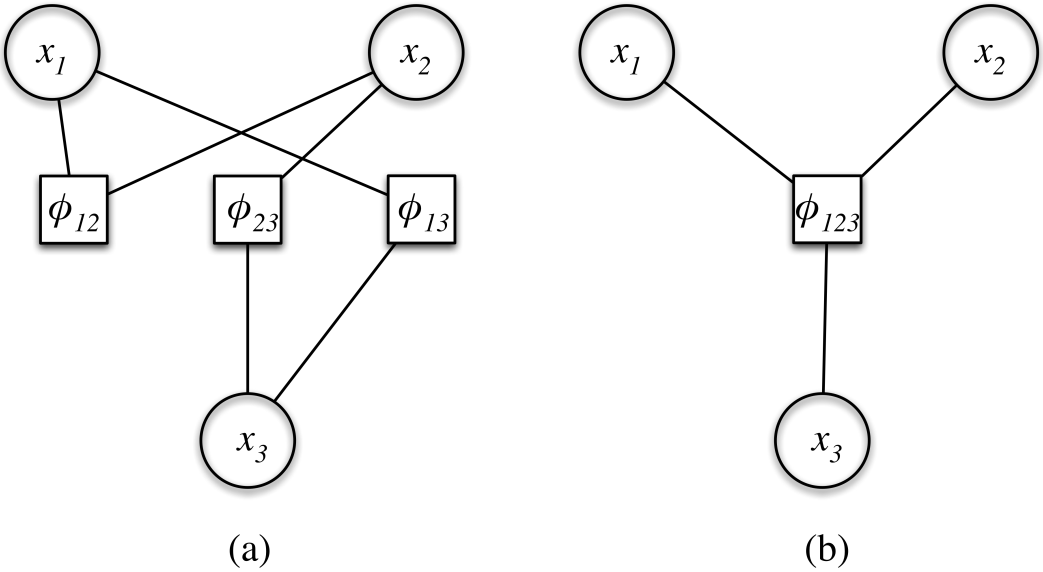


Figure 3.4: Two factor graphs for the Markov network in Fig. 3.3. (a) the factorization of Eq. 3.3 and (b)the factorization of Eq. 3.4

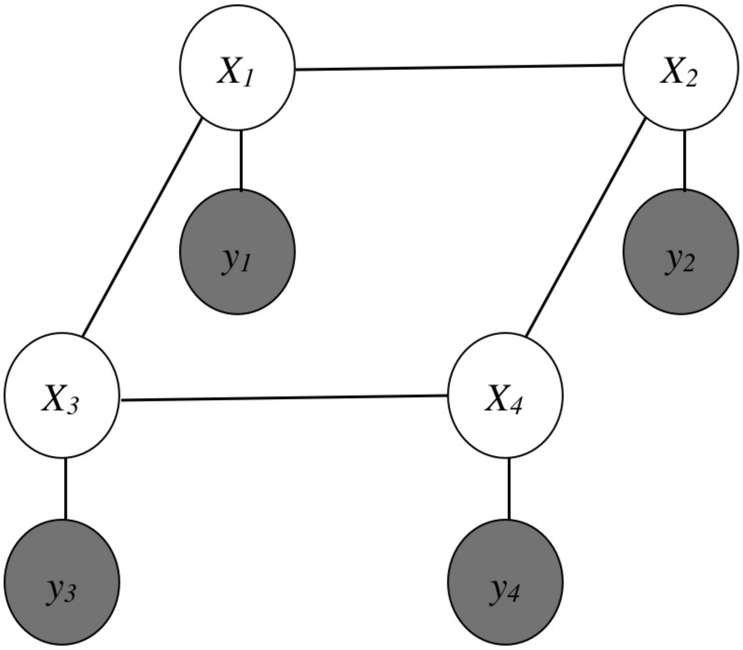


Figure 3.5: A conditional random field (CRF) with 4 variables $X_1..X_4$, and 4 observed features $y_1....y_4$.

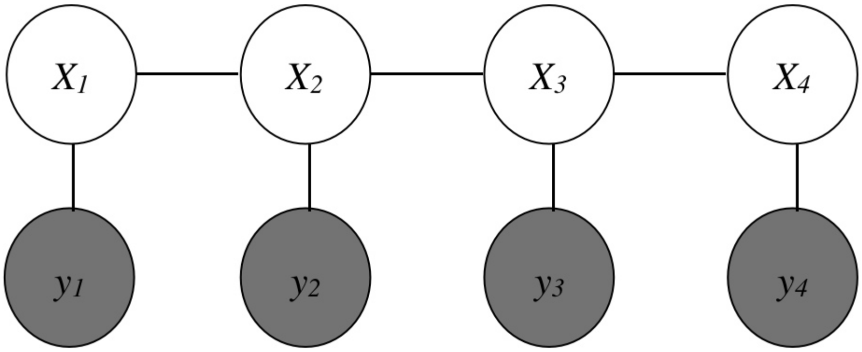


Figure 3.6: A linear chain CRF with one state variable and one observed feature

Chapter 4

2D Image Registration

The first task to be carried out on images from the electron microscope is their alignment (Fig. 2.8), typically requiring significant manual intervention by a user due to uncertain and weak local features. The precision of this alignment directly affects the resolution of the reconstruction, and therefore it must be carefully executed. The goal in this chapter is to automate this task and overcome this bottleneck. We will see that by incorporating geometric context into a probabilistic framework it is possible to to overcome this uncertainty and achieve this goal.

For aligning a stack of images, it is necessary to estimate each of their projection maps, and effectively reproject them to a consistent reference frame, such that their playback is smooth. This is equivalent to image stabilization when a movie is taken with a handheld camera on high zoom. The high magnification amplifies small mechanical motions of the camera in the video. The of an electron microscope ~~being a~~ high magnification ~~imaging device~~ also amplifies even the smallest motion of the mechanical stage, and must be corrected for. We shall see that this problem can be broken down into two subproblems: 1)detection and tracking of features across images (correspondence), and 2)calculating the transform to be applied to each image to ensure smooth playback. The second problem is also known as structure from motion, and depends on the solution to the first problem.

has this been defined?

4.1 Precision Registration of 2D Images

Image stabilization for natural scenes relies on reliable detection of distinct features in different frames so it can ~~for subsequent~~ correspond~~ence,~~ track~~ing~~, and finally motion correction the images ~~as described above~~. However, such features are not generally available in ET images, and this process is prohibitively difficult even when done manually. At the same time, the precision of the alignment directly determines the effective resolution of the reconstruction, and is required for high resolution images ~~must be considered carefully~~. For the express purpose of enabling precision alignment, biologists inject the sample with fixed diameter spherical gold beads which show up as high contrast features, as shown in Fig. 4.1. Even with this aid in place, detection of the bead features can be difficult, especially at the higher tilt angles as shown in Fig. 4.1(b) for 50 degrees. Another problem for alignment is that even if the beads are detected accurately, they are not very distinct from one another, which can complicate their tracking. This is especially problematic when beads are grouped together, as is also the case in Fig. 4.1.

An example of corresponding markers in a pair of cryo ET images is shown in Fig. 4.2. When a marker is isolated and its corresponding marker is nearby, the choice of correspondence is straightforward. The correspondences depicted with yellow arrows are such cases. In other cases however, the above mentioned problem of grouping markers can make the choice confusing, especially if false positive candidates have been detected. The red arrows in the figure illustrate such cases. As we shall see, a key to overcoming this this confusion is the consideration of the spatial relations between multiple markers.

4.2 Previous Methods

Extensive work has been published for automatic marker detection and correspondence (e.g. [KMM96, RHMM99, BHE01, BZ06, HB07, ZKB$^+$07, Fra06]) which has performed well for many datasets. However, correspondence for low SNR cryo-ET images is still prone to error for reasons previously described. For the most part, these methods attempt to correspond individual markers one at a time in adjacent

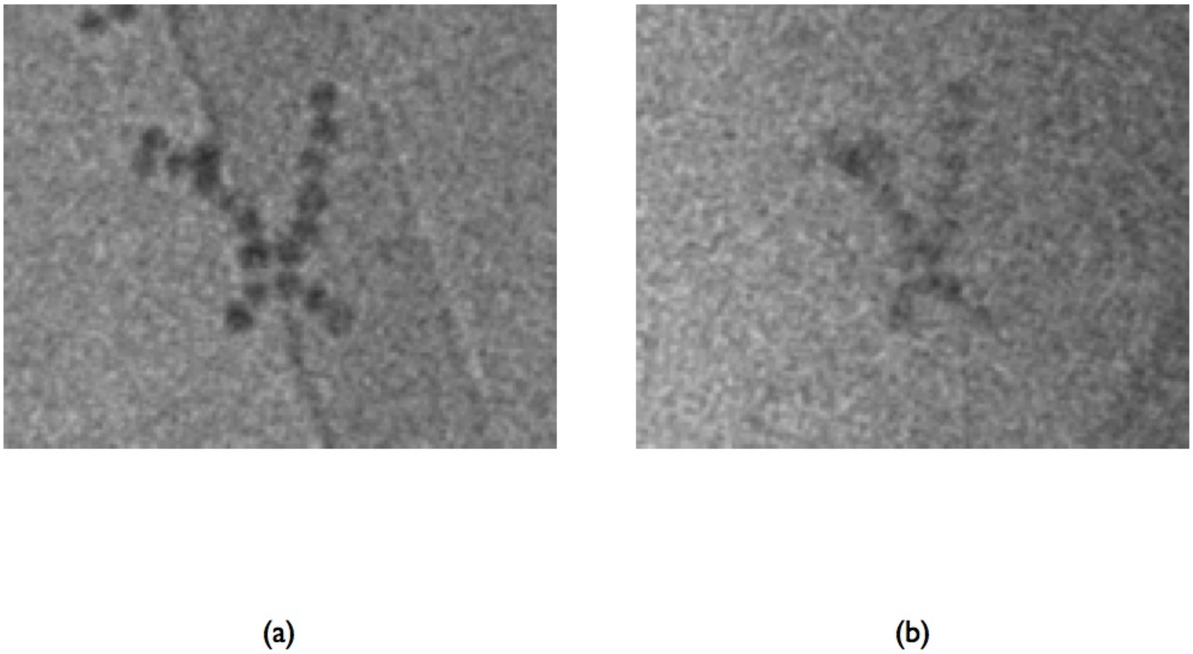


Figure 4.1: Gold beads injected into the sample for alignment (a)0 degree projection (b)50 degree projection

images using assumptions about the underlying projection geometry. The hope is that with a better prediction of where the corresponding marker should be, a better correspondence decision can be made. Such approaches will be fundamentally limited by the accuracy of the detection and prediction of marker locations.

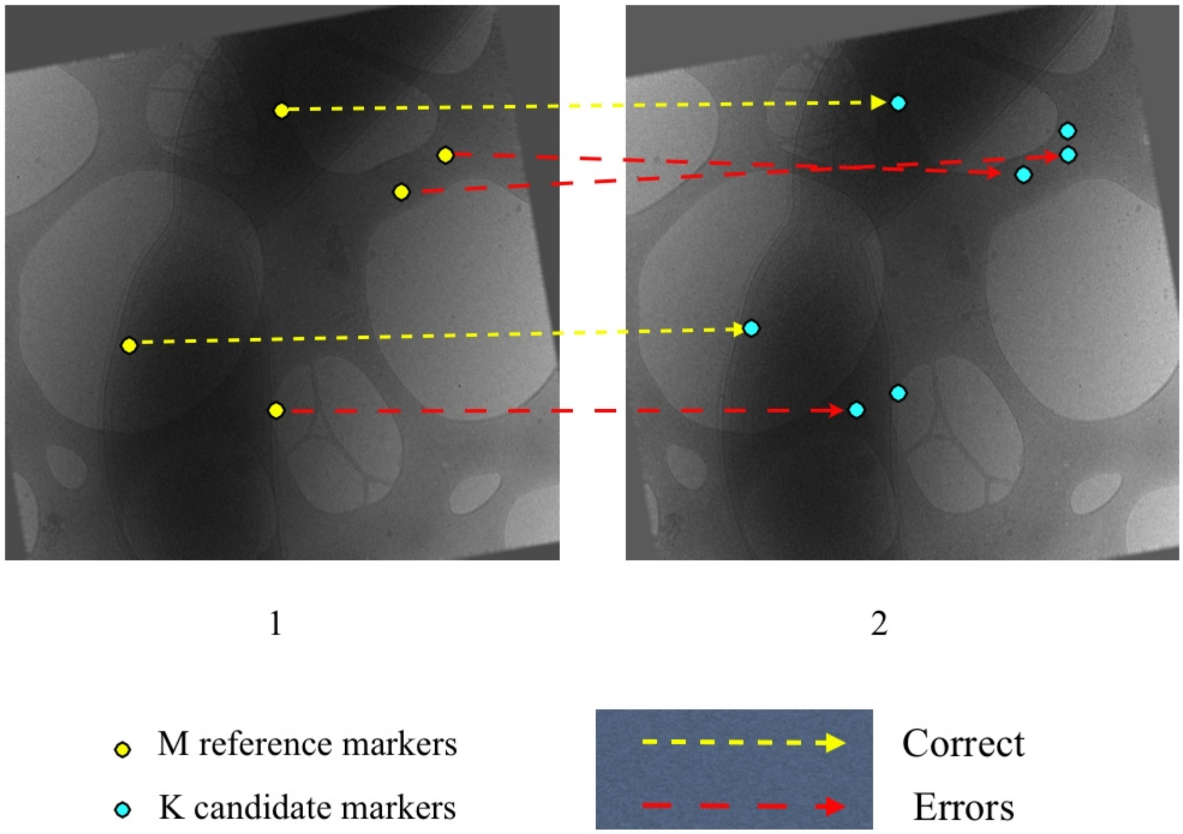


Figure 4.2: Finding correspondence for 5 markers in left image from among 7 possible markers in right image

4.3 RAPTOR:Tracking Groups of Markers Using Graphical Models

[1]

A fundamental idea for overcoming the uncertainty in the marker detection and correspondence is to consider matching groups of markers. Rather than model

[1] RAPTOR was joint work with Fernando Amat, and the entire work is presented here to preserve context. The work in sections 4.4 and 4.5.1 was led by Farshid Moussavi and the work in sections 4.5.2 and 4.6 was led by Fernando Amat.

the underlying projection geometry for corresponding one marker at a time, we infer 2D geometric similarities between groups of markers in neighboring images. For example, in addition to searching for a single matching marker in an adjacent image, we also search for a matching group of markers that has a similar spatial arrangement in an adjacent image. This intuition is captured and exploited in the method we present here called RAPTOR- Robust Alignment and Projection estimation for TOmographic Reconstruction.

4.4 Probabilistic Framework

We seek the most likely projection model given the observed 2D image data. To find this model involves detecting marker locations in each image, recovering the marker trajectories across the tilt series, and then finally fitting a projection model to the estimated trajectories. We define the following variables. Let $\mathbf{R} = \{R_j\}$ be the set of 3D locations of M markers in the original volume $R_j \in \mathbb{R}^3, j = 1...M$. Let there be N projected images of this volume, and $\boldsymbol{\theta}_i$ be the projection parameters (projection angles, rotations, translations) for the i^{th} projection in the image tilt series, $i = 1...N$. Let $\mathbf{I}_i$ be the i^{th} projection image. Then let $\mathbf{S}_i$ be an ordered set of 2D point locations $\mathbf{S}_i = \{S_{i,j}, S_{i,j} \in \mathbb{R}^2, j = 1...M\}$ corresponding to the 3D location in $\mathbf{R}$. Also let $\mathbf{B}_i$ be the set of K unordered noisily detected observations of $\mathbf{S}_i$ in $\mathbf{I}_i$, $\mathbf{B}_i = \{B_{i,l}, B_{i,l} \in \mathbb{R}^2, l = 1...K\}$. We take $K > M$ to allow for false positives and poorly detected true positives. Let the observable $\mathbf{O}_i$ be the union of $\mathbf{B}_i$ and image $\mathbf{I}_i$. For convenience, we define $\boldsymbol{\theta}$, $\mathbf{S}$, and $\mathbf{O}$ respectively be the sets $\{\boldsymbol{\theta}_i\}$, $\{\mathbf{O}_i\}$, and $\{\mathbf{S}_i\}$ over all $i = 1...N$. These quantities are shown graphically in Fig. 4.3. The relationship between these variables is captured in the conditional random field (CRF) shown in Fig. 4.4. The variables to be inferred are $\mathbf{R}$, $\boldsymbol{\theta}$, and $\mathbf{S}$; and the observed features are the $\mathbf{O}$.

We seek to find the best projection model by doing a MAP estimation on the variables, i.e. finding the arguments which maximize the probability of $\boldsymbol{\theta}$, $\mathbf{S}$, and $\mathbf{R}$ given the observed values of $\mathbf{O}$. This probability can be expressed as the product

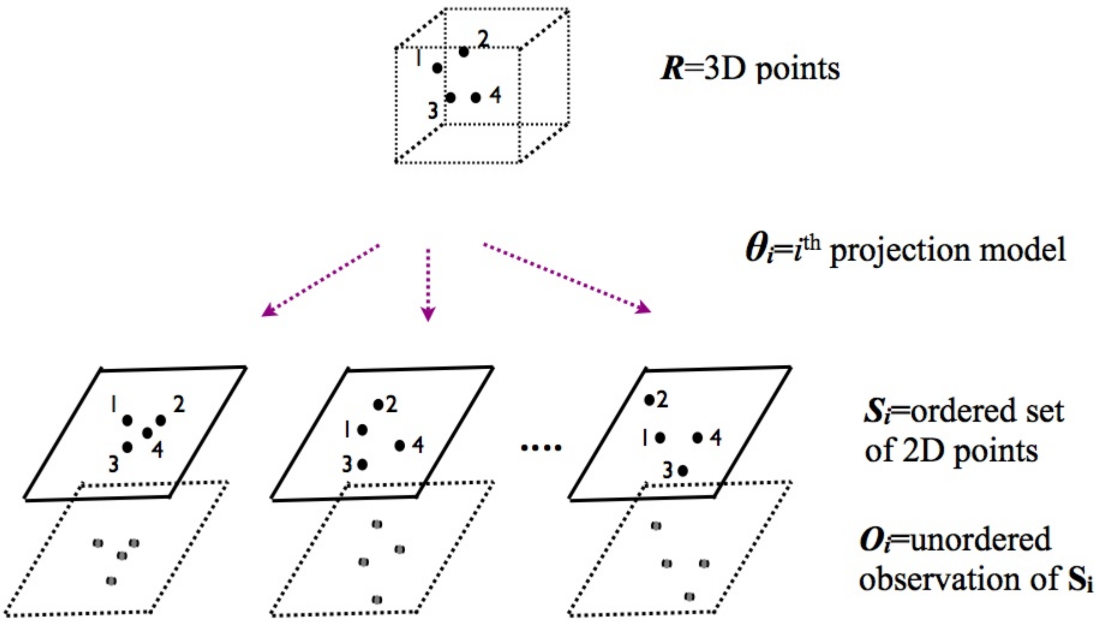

1
2
3
4
R=3D points
θ_i=*i*th projection model
S_i=ordered set of 2D points
O_i=unordered observation of **S_i**
....

Figure 4.3: Variables in the alignment with markers problem

of two terms:

$$\mathbf{P}(\mathbf{R},\mathbf{S},\boldsymbol{\theta}|\mathbf{O}) = P(\mathbf{S}|\mathbf{O})P(\mathbf{R},\boldsymbol{\theta}|\mathbf{S},\mathbf{O}) \tag{4.1}$$

The two terms in Eq. 4.1 represent two subproblems in alignment. The first term is the correspondence of markers, or tracking problem. The second term is the projection model estimation from the corresponded markers and their locations., and ~~It~~ is solvable by a variety of approaches available from structure from motion (SFM) when the projection maps are affine. Further study has been done on the case of nonlinear

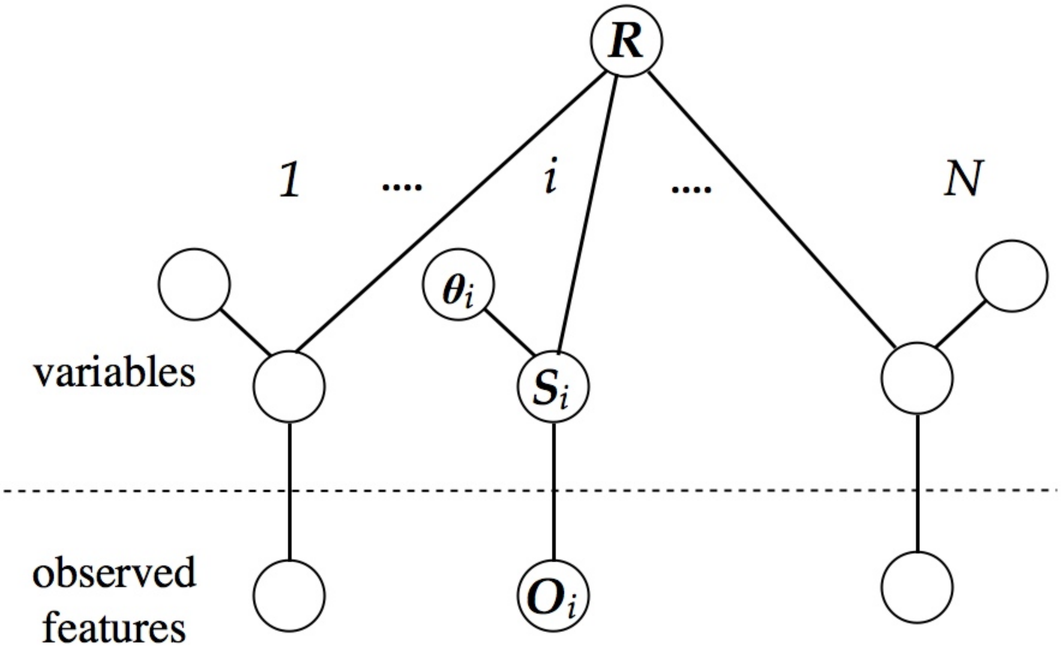


Figure 4.4: Conditional random field relating the variables in the alignment

Footnote

is that good, bad?

projection maps in [LBPE06]. In this work, we model and solve the two problems separately and disjointly. We focus below first on the tracking problem, which has been less well understood and has been the main reason for needing manual intervention. We subsequently describe our model and solution for projection model estimation in section 4.6.

The tracking problem can be represented in terms of a separate CRF. From the overall CRF in Fig. 4.4, it can be seen that if $\mathbf{R}$ and $\boldsymbol{\theta}_i$ are not given, the $\mathbf{S}_i$ are dependent on each other, and can be modeled as a separate CRF in which all the $\mathbf{S}_i$ are fully connected as shown in Fig. 4.5. We seek to find the ordered set of markers $\mathbf{S}_i$ by corresponding the unordered noisy observations $\mathbf{O}_i$, thus maximizing the probability $P(\mathbf{S}|\mathbf{O})$. This maximization is performed on the CRF in Fig. 4.5, conditioned on the observed evidence $\mathbf{O}$.

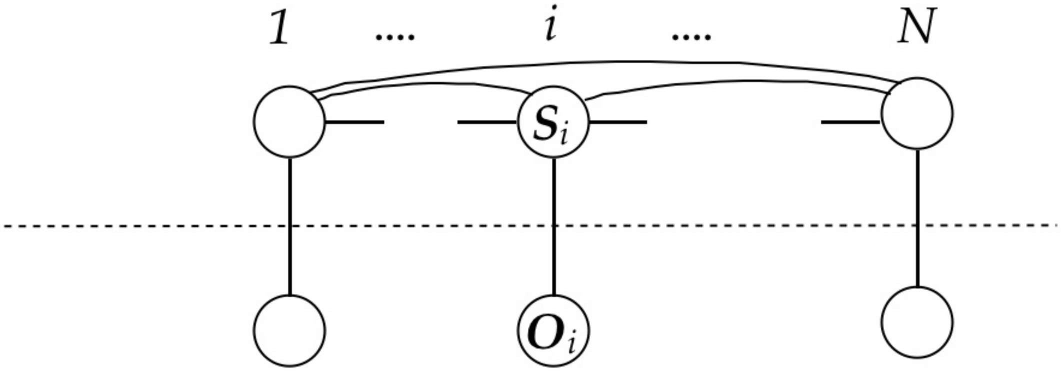
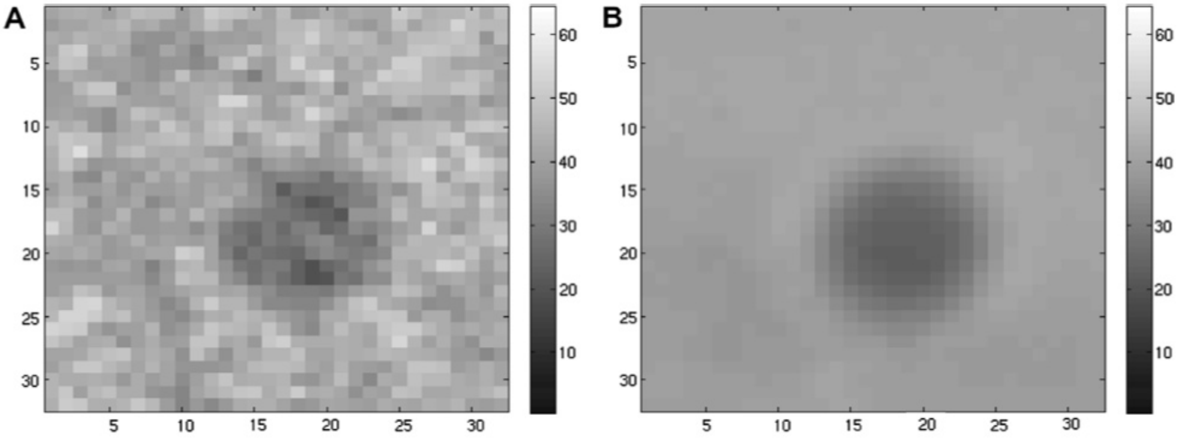


Figure 4.5: Conditional random field relating the variables in the tracking subproblem

The first step in the marker tracking is to obtain the observations $\mathbf{O}_i$. This involves detecting markers in all the images, and providing a score for each detected marker. For this step, a template is required. The user picks a small number (we found 3 to be sufficient) of marker centers, and the program then averages the corresponding marker patches. This provides a preliminary template. These can also be picked automatically with a synthetic template which is generated based on marker diameter and applying a Gaussian blur. We then cross-correlate this template with all the images at tilt angles between +30 and −30 and do a peak search and select the first five peaks from each of those images. The particles corresponding to these peaks are then averaged to obtain a template that is used for the final cross-correlation. The noise of the new template is much reduced relative to the preliminary template. The selection of the first three markers for the preliminary template is the only manual step required in RAPTOR, and is much faster than a standard initial seeding step used in many packages. Fig. 4.6 shows a comparison between an individual (of three) marker used for the preliminary template and the final template generated by this approach. In a last pass, we once again cross-correlate the final template with all the images, do a peak search, obtain all the peaks and apply a threshold. The resulting set of peaks in each image is our list of candidates in that image. If the number of target markers is M, the number of peaks selected as candidates per image ranges from $3M$ to $4M$.

Having obtained the observable features $\mathbf{O}$, we next seek to find the ordered set of markers $\mathbf{S}$ by corresponding them, thus maximizing the probability $P(\mathbf{S}|\mathbf{O})$ and performing a MAP inference over the CRF in Fig. 4.5. This ideally would involve inferring all the $\mathbf{S}_i$ at once, which represents a large combinatorial problem. Rather, to contain the complexity, we infer only for a pair of nearby images first, and then combine these local results heuristically to find the solution for all the images. First, we will describe the correspondence of markers for one pair of nearby images.

We consider a pair of images $\mathbf{I}_{i-1}$ and $\mathbf{I}_i$. $\mathbf{S}_{i-1} = \{S_{i-1,j}\}$ and $\mathbf{S}_i = \{S_{i,j}\}$ $(j = 1...M)$ are the random variables corresponding to M markers in image $\mathbf{I}_{i-1}$ and $\mathbf{I}_i$ respectively. Recall $\mathbf{B}_i = \{B_{i,l}\}$, $(B_{i,l} \in \mathbb{R}^2, l = 1...K)$ be the locations of K markers detected in image $\mathbf{I}_i$ (typically we take $K > 3M$). We also have by definition $\mathbf{O}_i = \{\mathbf{I}_i\} \cup \mathbf{B}_i$. The variables $S_{i,j}$ can take on values from $\mathbf{B}_i$, as well as a "null" assignment signifying no correspondence for marker R_j in image $\mathbf{I}_i$. For this pair of images, our goal is maximize $P(\mathbf{S}_i|\mathbf{S}_{i-1}, \mathbf{O}_{i-1}, \mathbf{O}_i)$. The joint distribution $P(S_{i,1}, ..S_{i,j}, ..S_{i,M}|\mathbf{S}_{i-1}, \mathbf{O}_{i-1}, \mathbf{O}_i)$ assigns a probability value for each assignment of the $S_{i,j}$. Our goal is to find the assignment that maximizes this distribution



to much space

Figure 4.6: Single marker image and average marker image after detection of hundreds of markers

$$\mathbf{S}_i^* = \arg \max_{S_{i,1}..S_{i,M}} (P(S_{i,1}, ..S_{i,j}, ..S_{i,M}|\mathbf{S}_{i-1}, \mathbf{O}_{i-1}, \mathbf{O}_i)) \tag{4.2}$$

We represent this conditional distribution using a Markov graph, in which each node is the variable $S_{i,j}$, and connections are between nodes which are members of one pattern. In general, if we are not conditioning on $S_{i-1,j}$, all patterns and correspondences would be possible, and we would connect all nodes $S_{i,j}, j = 1...M$, to get a fully connected graph. However, we can use the values from the variables

What approx are you making? Explain why pairs are enough.

$S_{i-1,j}$ to ~~simplify the graph and~~ give ~~it~~ this graph a simpler structure. We determine a pair of variables S_{i,j_1} and S_{i,j_2} to be directly dependent on each other (and hence have an edge between them in the Markov graph) if their predecessors in the previous image (S_{i-1,j_1} and S_{i-1,j_2} respectively) are within a certain distance of each other. This distance is fixed and is chosen as a model parameter. This simple criterion directly gives us the Markov graph structure, an example of which is shown in Fig. 4.7. This example follows the example shown in Fig. 4.2, in which 5 markers in image $\mathbf{I}_{i-1}$ needed to find their correspondences in image $\mathbf{I}_i$.

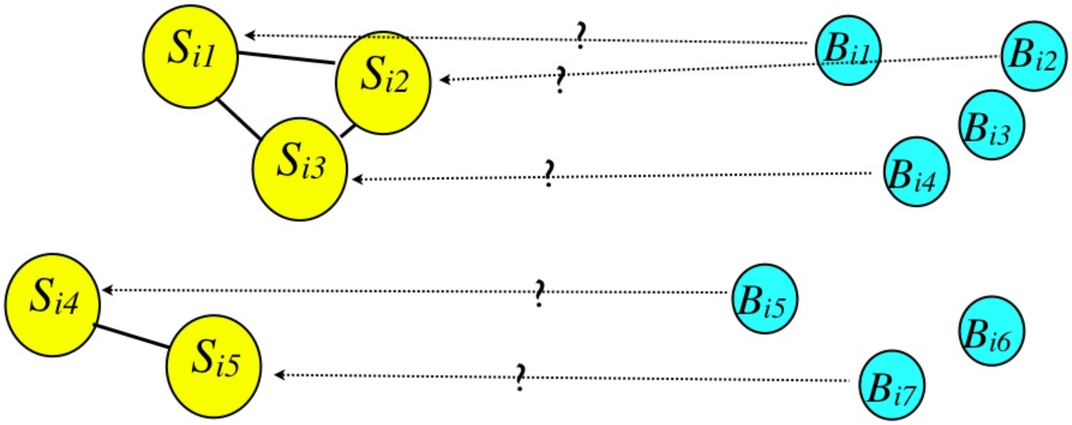


Figure 4.7: Markov graph for correspondence of markers in a pair of images. The nodes $S_{i,j}$ are variables which can take on values $B_{i,l}$

This Markov graph defines the following factorized conditional distribution, as described in section 3.2:

$$P(S_{i,1},..S_{i,j},..S_{i,M}|\mathbf{S}_{i-1},\mathbf{O}_{i-1},\mathbf{O}_i) = \frac{1}{Z}\prod_{j_1}\phi^s_{j_1}(S_{i,j_1})\prod_{S_{i,j_2}\subset N(S_{i,j_1}),j_2>j_1}\phi^p_{j_1,j_2}(S_{i,j_1},S_{i,j_2}) \tag{4.3}$$

where Z is a normalizing constant, and we have dropped the observable features terms on which the distribution is conditioned in the factors for brevity. To complete the definition of this distribution, we need to define the singleton and pairwise factors, ϕ^s and ϕ^p respectively.

First we will define a function that will be common to both types of factors. Let v_1 and v_2 be two vectors in $\mathbb{R}^2$, and k be a constant. We define

$$f_k(v_1,v_2) = e^{(-k||v_1-v_2||)} \tag{4.4}$$

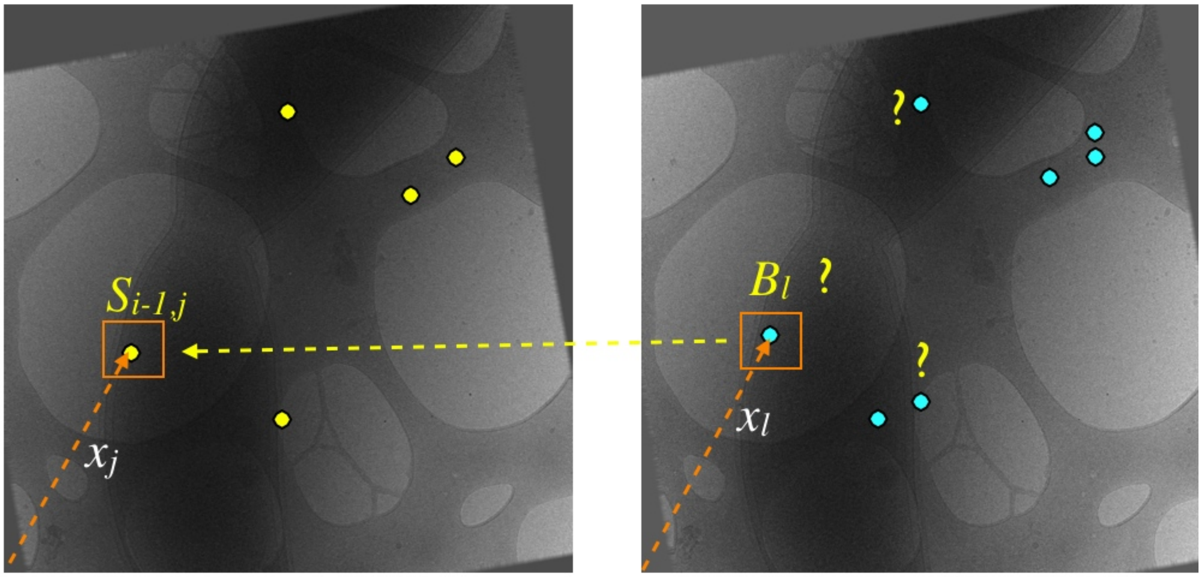
is this really not squared?

which is a Laplacian rolloff with the norm of the vector difference between v_1 and v_2 with rate k.

The singleton factors $\phi^s_{j_1}(S_{i,j_1} = B_{i,l_1})$ encode how likely it is that marker S_{i-1,j_1} in image $\mathbf{I}_{i-1}$ corresponds to marker B_{i,l_1} in image $\mathbf{I}_i$, i.e. the j_1^{th} marker in image $\mathbf{I}_i$ is B_{i,l_1}. The criteria we use for this likelihood are physical similarity and proximity. The components of the factor are $NCC(j_1,l_1)$- the normalized cross correlation of a patch centered at marker S_{i-1,j_1} in image $\mathbf{I}_{i-1}$ with a patch centered at marker B_{i,l_1} in image $\mathbf{I}_i$, and a Laplacian rolloff factor with the distance between these two markers:

$$\phi^s_i(S_{i,j_1} = B_{i,l_1}) = NCC(j_1,l_1)f_{k_1}(x_{j_1},x_{l_1}) \tag{4.5}$$

where x_{j_1}, x_{l_1} are vectors from the origin to marker locations S_{i-1,j_1} and B_{i,l_1} respectively, and k_1 is a model parameter constant. These quantities are depicted graphically in Fig. 4.8.



how do you define the origin?

Figure 4.8: Patches and vectors used for the singleton factors

The pairwise factors $\phi^p_{j_1,j_2}(S_{i,j_1} = B_{i,l_1}, S_{i,j_2} = B_{i,l_2})$ encode how likely it is that the pair of markers S_{i-1,j_1}, S_{i-1,j_2} in image $\mathbf{I}_{i-1}$ correspond to the pair of markers B_{i,l_1}, B_{i,l_2} in image $\mathbf{I}_i$, i.e. the j_1^{th} and j_2^{th} marker in image $\mathbf{I}_i$ are B_{i,l_1} and B_{i,l_2} respectively. The criteria for this assignment are purely geometric ones- we favor correspondences between pairs of markers which form similar vectors. Let $v_{j_1 j_2}$ be the vector formed from marker S_{i-1,j_1} to marker S_{i-1,j_2} in image $\mathbf{I}_{i-1}$, and $v_{l_1 l_2}$ be the vector formed from marker $B_{i.l_1}$ to marker B_{i,l_2} in image $\mathbf{I}_i$. A good pairwise correspondence is one in which the vectors formed by the pairs are as close to each other as possible, i.e. their vector difference is small. We also include the proximities between the respective markers and their candidates for correspondence, such that we prefer correspondences where S_{i-1,j_1} is near B_{i,l_1}, and S_{i-1,j_2} is near B_{i,l_2}. The pairwise factors are thus defined as:

hmm — I thought you defined f_K w/ vectors to use it here too but you don't — Why define it that way?

$$\phi^p_{j_1,j_2}(S_{i,j_1} = B_{i,l_1}, S_{i,j_2} = B_{i,l_2}) = e^{-k_2||v_{j_1,j_2} - v_{l_1,l_2}||^2} f_{k_1}(x_{j_1}, x_{l_1}) f_{k_1}(x_{j_2}, x_{l_2}) \qquad (4.6)$$

where v_{j_1,j_2} is the vector from marker S_{i-1,j_1} to marker S_{i-1,j_2}; v_{l_1,l_2} is the vector from marker B_{i,l_1} to marker B_{i,l_2}; x_{j_1}, x_{j_2}, x_{l_1}, x_{l_2} are vectors from the origin to markers S_{i-1,j_1}, S_{i-1,j_2}, B_{i,l_1}, and B_{i,l_2} respectively; and k_2 is another model parameter constant. These quantities are depicted graphically in Fig. 4.9. Note that the rolloff factors f_{k_1} occur in both the singleton and pairwise factor equations (Eq. 4.5 and 4.6). This has the effect of making the rolloff steeper when a marker has more neighbors in the graph, making the search more constrained when there is more crowding. Specifically, the effective rolloff rate becomes linear with the number of neighbors (degree) of the marker's variable in the graph. is this what you want?

This completes the definition of our model for the conditional distribution over $S_{i,j}$'s given $S_{i,j-1}$'s, $\mathbf{O}_{i-1}$, and $\mathbf{O}_i$.

4.5 Inference of Marker Correspondences

Having defined the conditional distribution over marker assignments $\mathbf{S}_i$ given $\mathbf{S}_{i-1}$, $\mathbf{O}_{i-1}$, and $\mathbf{O}_i$ for one pair of images, we then seek the most likely trajectories given the observed features. This inference is executed in two stages- locally for each pair of neighboring images, followed by globally for the whole set of images.

4.5.1 Inference of Marker Correspondences in One Image Pair

We first seek to infer the most likely joint assignment to the marker correspondences in each pair of nearby images, thus performing the MAP query of Eq. 4.2. Even

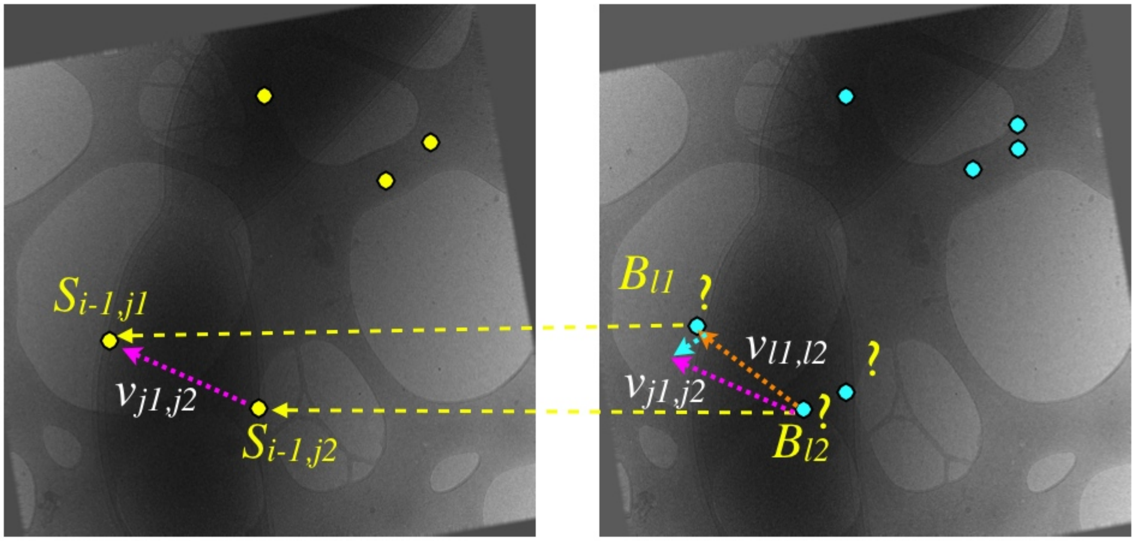


Figure 4.9: Vectors used for the pairwise factors

though we have defined this distribution, maximizing it is still an expensive combinatorial task, as it must evaluate an exponentially large number of assignments to find the highest scoring one. Furthermore, the distribution is in general not convex. Therefore for any scenario other than a very small number of markers, we must consider more efficient approximate inference methods, as discussed in section 3.1. In ET images we typically have dozens to several hundred markers.

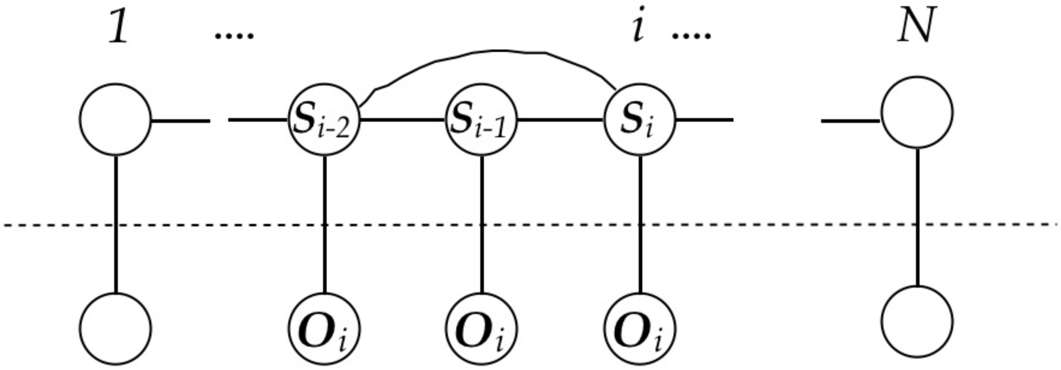
It is possible to use approximate MAP estimation, which attempts to solve Eq. 4.2 with no guarantees of reaching a global optimum. In such an approach, significant care should be taken to deal with outliers due to false positives in the marker detection. An alternative strategy is to perform a marginal query for the individual assignments and then combine them. This marginal query seeks the marginal distribution for each marker's correspondence from the joint distribution of all correspondences. Since the graph has loops in general as well as many variables, a marginal query inference is approximate, and we chose loopy belief propagation (LBP), which works well in practice on such graphs. A brief introduction to LBP was provided in section 3.3.1. [For a detailed description of LBP the reader is referred to [Bis06, KF09].]

This should be in not here

4.5.2 Global Correspondence from Image Pair Correspondence

Having obtained the correspondence between two images, we would like to now construct the global correspondences. If the correspondence for two images were perfect and had no mistakes, this would have been trivially accomplished by cascading the results of sequential pairs of images. However, we would like to provide robustness by considering redundant pairs of images and cross checking. This effectively amounts to pruning the CRF in Fig. 4.5 such that images more than a certain distance from each other in the tilt series will not be directly related, as shown in Fig. 4.10. However, all image correspondences will still be related indirectly. We next describe a heuristic method that combines the results of pairwise image correspondence inferences to get the global correspondence.

The first step of global correspondence is to build trajectories. To do this we use a multi-level approach. The first level contains correspondences between the i^{th} image and the $(i+1)^{th}$ image. The second level contains the correspondences between the i^{th} and the $(i+2)^{th}$ image. We use up to 3 levels in images spaced up to 2 apart because after that it is difficult to find correspondences due to the tilt difference between projections. We initialize the first trajectory as the first point in image 1 and level 1. We call this point p_{11} (in general, we define p_{ij} as the point in image i and trajectory j). Local correspondence gives us the location of this trajectory in image 2, i.e. p_{21}. Next, we try to find p_{21} in the local correspondence



Need better caption

Figure 4.10: Pruned CRF relating the variables in the tracking subproblem

between image 2 and image 3. If we find it, that returns p_{31} and we repeat the process for this new location to find p_{41}. We use the extra levels in two ways. First, if a local correspondence for a track is missing, we look in the next level of the global correspondence to complete the track. For example, if we cannot find p_{31} from p_{21} in the first level, we try to find p_{41} from p_{21} in the second level, which contains local correspondence between image 2 and image 4. If we find p_{41}, we continue the process for this new location to try to extend the trajectory. If we do not find it, we try to find p_{51} from p_{21} in the third level, which contains local correspondence between image 2 and image 5. We iterate this process until the first trajectory can not be extended longer, either because we reach the last image (full trajectory) or because one location can not be found in any level (partial contour). To create new trajectories, we use points in each level that have not been used in previous trajectories, since they are potential seeds of new trajectories. Having multiple match levels allows trajectories to recover from occlusion. Occlusion occurs if a marker present in one image can not be found, even by a human eye, in the next image. A variety of events can create occlusion. For example, a fiducial marker in the edge in one image can move out of the field of view in the next image. Or some other feature in the sample can occlude the marker in some views (two clouds of markers crossing each other.

The second way of using a multi-level approach, and the second step of global correspondence, is to add confidence in the trajectories obtained in the first step. We do this using the redundant information contained in the local correspondences that have not been used in the first step. For example, suppose we found p_{21}, p_{31}, and p_{41} using the first step. We can check this path against the local correspondence for p_{21} in image 4, which is contained in the second level. We count the number of times such comparisons fail in each trajectory. If this occurs more than one fifth of the length of this particular trajectory, we drop the trajectory. Otherwise, we retain the consistent part of the trajectory.

how?

P21 is in image 2 right?

Finally, only trajectories of certain length are kept to estimate the projection model. The length is selected automatically to assure that we have enough points in each image to estimate the projection model. As explained in the next section, the projection estimation is able to handle partial trajectories of any length across the tilt series. This is extremely useful at high tilt angle, where many markers are present in only a few images before disappearing.

4.6 Projection Model Estimation

Having obtained an estimate for the ordered set of points **S**, we are now ready to estimate the projection parameters $\boldsymbol{\theta}$, thus maximizing the second term in Eq. 4.1. The projection model finds a common origin of coordinates for all the images in order

? is this what you mean?

to align them. Details about projection models can be found in [LBPE06]. We use convex optimization techniques to estimate a linear projection model. For the i^{th} image in the tilt series and j^{th} marker trajectory we have the following projection model:

$$\begin{bmatrix} \mathbf{S}_{x,i,j} \\ \mathbf{S}_{y,i,j} \end{bmatrix} = G_i \begin{bmatrix} \mathbf{R}_{x,j} \\ \mathbf{R}_{y,j} \\ \mathbf{R}_{z,j} \end{bmatrix} + \begin{bmatrix} t_{x,i} \\ t_{y,i} \end{bmatrix} \tag{4.7}$$

where G_j is a $2x3$ matrix representing an affine 3D to 2D transformation, $t_{x,i}$ and $t_{y,i}$ are 2D translation parameters. These quantities together represent the projection parameters $\boldsymbol{\theta}_i$. The unknowns in this equation are therefore $\mathbf{R}_j$, G_i, $t_{x,i}$, and $t_{y,i}$. To solve for these unknowns, we solve the following minimization problem:

$$min \sum_{i=1}^{N} \left(\sum_{j=1}^{M} w_{i,j} L \left(\begin{bmatrix} \mathbf{S}_{x,i,j} \\ \mathbf{S}_{y,i,j} \end{bmatrix} - G_i \begin{bmatrix} \mathbf{R}_{x,j} \\ \mathbf{R}_{y,j} \\ \mathbf{R}_{z,j} \end{bmatrix} - \begin{bmatrix} t_{x,i} \\ t_{y,i} \end{bmatrix} \right) + \lambda ||G_i - G_{0,i}||^2 \right) \tag{4.8}$$

where $L(*)$ is a penalty function, $w_{i,j}$ is a weight for point corresponding to trajectory j in image i, λ is a regularization constant, and $G_{0,i}$ is a prior belief on the affine transform derived from the microscope settings. The quantity $w_{i,j}$ is 1 if trajectory j has a correspondence in image i, and 0 otherwise. The objective of Eq. 4.8 is nonconvex in G_i and $\mathbf{R}$. However, if one of these variable is held constant and $L(*)$ is convex, then the objective is convex in the other variable. We solve the problem by holding one of these variables consant, solving the resulting convex problem, and then alternating. This approach is known as sequential convex programming ([BV04]) and works well and efficiently in practice.

Finally, in order to be robust to outliers, we use the Huber penalty ([Hub81, BV04]) for $L(*)$. In most previous work, $L(r) = r^2$, the familiar least squares

spell check

penalty. The Huber penalty uses the L_2 norm for small residuals, and the L_1 norm for large residuals, such that an outlier will not dominate the objective function and unnecessarily distort the final solution. how do you define small?

4.6.1 Iterative Refinement

As a final step towards better alignment, after projection model estimation we perform the following iterative refinement:

1. Remove outliers 2. Replenish missing markers in global trajectories 3. Refit projection maps

Outlier detection is possible after projection model estimation because we have complete projection maps and estimated trajectories. We remove outliers on each trajectory separately since all trajectories did not have an equal number of points. Therefore the projection maps fit the more complete trajectories better, and their statistics should be examined separately. We use a heuristic scheme to detect outliers. Firstly, any point which is greater than twice the diameter of a gold bead away from the predicted center is considered an outlier. Next, for each trajectory we form a vector of residuals and sort it. If all projections had similar quality, we would expect a similar range of residual values for all projections. However, the image quality degrades with tilt angle, so that we would expect to see a steady increase in residual value from lower tilt angle images to higher tilt angle images. By forming the first order difference vector on the sorted vector of residuals, we can easily detect an outlier by looking for a sudden jump in the first order difference value for that projection.

We next replenish missing markers using the obtained projection maps and the NCC score described in Eq. 4.5. For each missing point $x_{i,j} \in \mathbb{R}^2$ of trajectory j in image i, we predict where the point should be in images i and $i-1$ using the projection maps. We then take a patch surrounding the predicted trajectory point in image $i-1$ and use it as a template to find matches with the NCC score in a region of radius twice the gold bead diameter around the predicted point in image i. We take the highest scoring point in this region as long as the score is above 0.5

to be a replenished point.

After one round of outlier detection and missing marker replenishment, we refit the projection maps. We repeat this procedure iteratively until there are no more outliers and no more replenishments are possible.

4.7 Results

which is it?

In this section we evaluate the performance of Raptor. We applied RAPTOR to cryo-ET datasets of the bacteria D. grandis and C. crescentus. Each of these datasets poses unique and different challenges. Most of the datasets were acquired by hand, which results in tilt series with more discontinuous trajectories, or jerky movements across the angular range than in the case of automated acquisition.

Deinoccocus grandis cryo grids are ~~considerably~~ thick, with the large bacteria forming conspicuous meniscus in the amorphous ice. As a result, projections from high angles usually occlude many of the markers clearly visible at the lower angles. The cryo-grids used with these bacteria are also more irregular, uneven in shapes, contrast, and transparency than for other bacteria more commonly studied by cryo-ET. For this reason, the usual marker selection and tracking applications included in other programs tend to fail. Most of C. crescentus and D. grandis datasets were acquired with a total dose of at most 100 e^{-}/A^{2}, resulting in very noisy individual images. Caulobacter cryo-grids are thick and form a meniscus although they are still transparent to the beam. The surface of the grids is not as irregular as with Deinoccocus and the high angle occlusion problem is thus less severe. However, in all cases there are, in high tilt angle views, sets of very useful markers towards the image boundaries which disappear at lower angles.

We first examine its tracking performance, and then compare its resulting final reconstructions with those obtained by manually precision aligned images.

4.7.1 Alignment Results

The robustness of RAPTOR's correspondence is illustrated in Fig. 4.11.

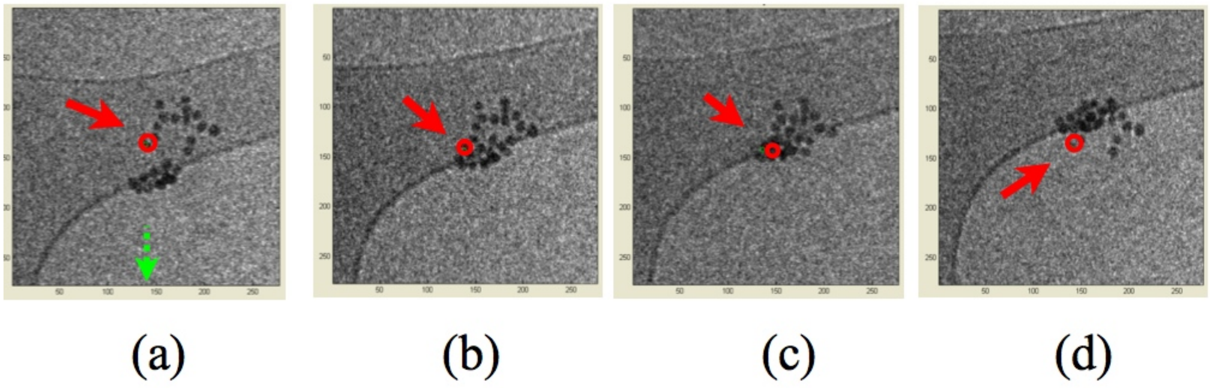


Figure 4.11: Tracking of a single marker through a point cloud. In frame (a), the marker pointed to by the red arrow is headed south. After frames (b) and (c), it emerges from the cloud and is still being tracked correctly.

We illustrate in Fig. 4.12 the overall performance of the algorithm in the alignment of a difficult dataset of C. crescentus. This dataset was acquired with a helium-cooled stage, and as a consequence the relative position of the markers was significantly more variable than is normally the case with liquid nitrogen-cooled stages ([CD05]). Panel (A) of Fig. 4.12 shows the projection acquired at 58 degrees. Panel (B) shows the set of trajectories recovered by RAPTOR. There are complete trajectories across the whole tilt series, there are trajectories that cover the whole angular range of the tilt series but are missing the marker in some images, and there are partial trajectories starting from both ends of the tilt series, disappearing at lower angles. The meniscus formed by the cell and bars of the cry-grid may occlude, at high tilt angles, markers chosen at low angles. At specific angles throughout the tilt series, the overlapped projection of groups of markers which belong to different planes within the cryo-grid may cause gaps in their tracked trajectories. Many gaps also represent locations where the probability of making a correct marker assignment is not sufficiently high. Finally, there are in the high angle projections, many markers with ideal SNR which are left out of the view in the projections at lower angles. All these events are represented in the diagram plotted in Fig. 4.12(B). The physical trajectories of all chosen markers in image space (xy coordinates in pixel number), in the raw data and the aligned data, are shown in Fig. 4.12 (C) and (D). The inset in each panel shows one single trajectory. No pre-alignment of the raw dataset was performed before obtaining the automatically aligned one, as is readily obvious from the cloud shape of marker trajectories in the raw data (Fig. 4.12). A comparison with the results of manual marker selection and tracking shows indistinguishable final tomographic reconstructions, which is the ultimate performance test.

4.7.2 Resolution Results

For all datasets, we also compared visually the quality of the tomographic reconstructions obtained after RAPTOR alignment with those obtained after alignment by an independent standard method. Two such comparisons are shown in Fig. 4.13. Panels (A) and (B) show, side by side, a one-pixel-thick slice through a tomographic reconstruction of a dividing C. crescentus cell obtained after manual alignment and after RAPTOR automatic alignment, respectively. The quality of both is equivalent. Another example is shown in Fig. 4.13, panels (C) and (D). The features observed in a one pixel-thick slice through a tomographic reconstruction are comparable in definition and resolution, whether the dataset was aligned with human intervention or automatically by RAPTOR. A better comparison is established by careful inspection of regular features, as shown in the insets. These reconstructions were binned by two, from an image size of 2048 pixels on edge to 1024, and each box measures 84 pixels on edge.

Fig. 4.14, panels (A) and (B) show the resolution plots for each final reconstruction obtained using cross-validation available in software Electra ([CGS05]). The plots demonstrate that RAPTOR obtains the same resolution as the manual reconstruction.

fix table

Dataset	Tilt range (deg)	Approx.. number markers	Bead diam. (pixels)	SNR (high tilt)	NLOO3D manual (0.3-nm)	NLOO3D RAPTOR (0.3-nm)	NLOO3D manual (0.5-nm)	NLOO3 RAPTO (0.5-nm
Caulo1	-58:2:54	15	10	0.038	14.26	13.81	30.16	24.48
Caulo2	-34:1:36	10	10	0.026	11.02	8.63	15.11	14.25
Caulo3	-62:2:58	15	10	0.007	18.01	16.35	24.82	26.78
Caulo4	-61:2:54	10	10	0.009	25.21	32.54	36.59	41.14
Caulo5	-60:1:60	70	10	0.294	10.35	10.34	13.32	13.18
Caulo6	-68:2:68	20	10	0.005	17.27	18.68	24.09	27.96
Caulo7-He	-54:1.5:51	60	10	0.024	15.44	18.34	27.01	27.89
Caulo8	-45:1:45	20	10	0.066	13.25	13.63	20.01	20.49
Caulo9	-60:1:62	10	8	0.032	7.51	7.66	14.32	16.95
Deino1	-62:2:57	30	10	0.044	19.28	19.64	29.13	30.15
Deino2	-60:1:57	20	7	0.008	14.75	10.56	20.55	15.18
Deino3	-25:1:26	40	10	0.044	21.04	19.38	30.93	27.65
Deino4	-62:2:62	15	10	0.001	-1	-1	-1	-1

Table 4.1: Summary of the datasets tested to assess RAPTORs performance. First columns contain information to present the variety of scenarios where RAPTOR was tested: spacing between projections, SNR of the gold beads and number of markers to track. The last four columns show a quantitative comparison of the resolution obtained using human-directed alignment and the automatic full-precision alignment presented in this paper. The method used for resolution assessment is noise-compensated leave-one-out (NLOO) by [**?**]).(-1) Method fails to align.

4.8 Discussion

RAPTOR has been tried in dozens of datasets in numerous labs, and has achieved reconstructions with precision comparable to that achieved with manual alignment. However, instead of a human taking up to one day to align manually, RAPTOR does so in under one hour on a common desktop computer. Initial results were reported in ([AMC+07]). RAPTOR continues to be used by several ET labs today. However, we have observed limitations in some datasets and discuss them below.

The most prominent limitation in RAPTOR is in the presence of a large number of gold beads (more than 400 per image). In such cases, the LBP marginal correspondence step does not find and resolve clear winners for the correspondence of the majority of markers due to a multitude of competing and confusing choices. For example, in a dataset with 400 markers, good correspondences may be found for 100 of them in a pair of images. Furthermore, in subsequent pairs of images, good correspondences may be found for another set of 100 out of the 400 total markers, resulting in partial trajectories. While the structure from motion can function with many partial trajectories, it performs better when there are more long (or better yet complete) trajectories. Therefore when most of the trajectories are small and partial, the achieved resolution in the reconstruction is limited. In such cases, the user can select fewer markers for tracking out of the original large set that have higher quality. We have found that the selection of the initial markers to be tracked is one of the key determinants to RAPTOR's success. In datasets with fewer than 100 markers, RAPTOR performs well when the user selects all or most of the markers. In datasets with more than 100 markers, the user can choose the 100 best scoring markers in the first image for tracking. While selection of a smaller number of initial markers helps achieve more robust tracking, small partial trajectories are still inevitable since the set of candidate markers may still not be consistent with the set of reference markers, and wrong correspondences may still occur.

huh?

Potential remedies to this limitation can be considered from two possible categories: (1)a better correspondence solution, and (2)a better hollistic solution to correspondence and structure from motion. The first category of approaches keeps the separated approach to alignment that RAPTOR follows as explained in section 4.4. RAPTOR's current solution to correspondence relies primarily on preserving pairwise distances between markers. One could favor preserving more detailed quantities, e.g. angles formed by groups of three markers. Ultimately, the first category of approaches fundamentally favors patterns of markers that are as similar as possible between images. This strategy is ultimately limited by lack of consideration of expected changes in the patterns between images due to projection geometry. The second category considers this information, and is more powerful, but also more complex. As correspondences are found for the first few images, it is possible to obtain rough but useful projection geometry information. This strategy involves running inference on the entire CRF in Fig. 4.5 rather than on two separate CRF's as RAPTOR currently does, and will be discussed further in sections 4.9.1 and 4.9.2.

Another limitation of RAPTOR is datasets with incremental angles of larger than 4 degrees. One assumption in RAPTORs local correspondence is that there is a small incremental angle between images in the tilt series. We have successfully tried datasets with incremental angles ranging from 0.5 to 4 degrees. This accommodates most common tomographic tilt series. However, for bigger angular increments, two problems will arise. First, adjacent images will be less similar. This might cause a decrease in the number of correspondences found between images. Second, it becomes more difficult to have many levels in the multi-level scheme, which might affect the robustness of the method. Although we can address this problem by modifying the global correspondence scheme it has not been necessary for any datasets that were available to us.

Another limitation of RAPTOR is the features used for marker detection. RAPTOR currently uses NCC with an average template shown in Fig. 4.6. Cryo ET images are generally low contrast and do not result in too many false positive from this detection. However, higher contrast plastic section and negative stained images return many more false positives due to more distinct edges present anywhere, thus throwing off the correspondence more. To extend RAPTOR to perform well on plastic sections it will be necessary to design a better marker detection scheme. This can be approached as a classification problem in a supervised learning framework.

4.9 Future Work

merge w 4.8 - seems a little repetitive

RAPTOR is publicly available for download from www-vlsi.stanford.edu/TEM/software.htm, and is in use by numerous labs that do ET. While it is doing well for common datasets, we believe that further enhancements are possible and beneficial which we describe below.

4.9.1 Joint Inference of Correspondence and Projection Model

Our current solution solves the two problems of marker tracking and projection model estimation separately and disjointly. The marker tracking takes no advantage of even partial projection model information and uses very basic geometric context assumptions to make its decisions. In practice this has worked well for most datasets, generally with less than 200 markers to be tracked. However, some datasets have more markers and therefore crowding, as well as local distortions, creating more confusion for the marker tracking. A possible way to overcome this confusion is to introduce partial projection model information inferred from intermediate tracking results. This implies a much more integrated approach to the two main subproblems. While the overall CRF model still applies, the inference method over this large CRF would need to change. There are many possible options to do this, and we consider two here: expectation maximization (EM) and particle filtering.

In an expectation maximization (EM) framework, we would seek to maximize an energy function over all the variables in a sequential projection by projection fashion. At each slice, we would perform correspondence estimation and tracking, which would take into account projection parameters, and could be implemented as a MAP or marginal query. We then would use these results to roughly estimate projection parameters incrementally. These parameters would then be fed back into the current tracking query, and this process would be iterated. Since the parameters get more refined as tracking progresses, the refined parameters could be used to go back to previous projections and do better tracking. This iteration could happen over a number of forward backward passes to increasingly explore the solution space.

huh?

This is not what we talked about. Why do it this way?

A more complicated approach would be a particle filtering framework. In this framework we would maintain an empirical distribution over all the variables using multiple hypotheses for the correspondences, since they present the most uncertainty when there are many markers present. To do this we must generate and score particles over these correspondences and sets of projection parameters that reflect multiple hypotheses for the correspondences. This is in general not trivial, and may require Markov chain Monte Carlo (MCMC) techniques which sample particles closer and closer to our target distribution. By combining this empirical distribution with the estimated distribution of the projection model parameters, we may then have an opportunity to maximize all of Eq. 4.1 and effectively explore a larger portion of the overall space. We could also iterate from there to generate newer particles and refine the empirical distribution as well.

4.9.2 Extension of Correspondence CRF to Multiple Images

Did not follow what would you really do here

In order to maintain robustness while tracking a larger number of markers, it would be useful to extend the correspondence CRF beyond pairs of images. The current global correspondence scheme is good at detecting inconsistencies and rejecting offending markers. However, it does nothing to correct these inconsistencies in a way that could salvage the otherwise useful detected markers. In such a formulation, the variables could be trajectories themselves, and they would take on values of sets of markers in all the images. As in the current case, the most contextual information gain would be obtained from pairwise factors. The pairwise factors in this case would try to find consistency between pairs of trajectory assignments, and may need to consider partially estimated projection. This approach would expand the scope of the context in the CRF and would be an interesting direction for improvement.

4.9.3 Higher Order Factors

The next enhancement to RAPTOR's correspondence CRF is the use of higher order factors, beyond pairwise. RAPTOR currently seeks similar patterns of markers which preserve pairwise distances. New factors could be defined that look for more detailed geometric similarities between patterns. Since more details of the pattern would be considered, the factors would need to be over a larger set of markers. For example a triplet factor would look for assignments that form similar triangles as the group of markers they are being considered for.

Despite all these limitations, RAPTOR has been successful in aligning a large majority of cryo ET datasets with precision by a variety of research facilities, with success rates of greater than 70%. For the most part, this time consuming bottleneck of the cryo ET pipeline shown in Fig. 2.8 has been relieved, and the bottleneck has shifted to downstream portions of this pipeline. In the next chapter we will examine the next bottleneck in this pipeline- segmentation of cell boundaries from reconstructed tomograms.

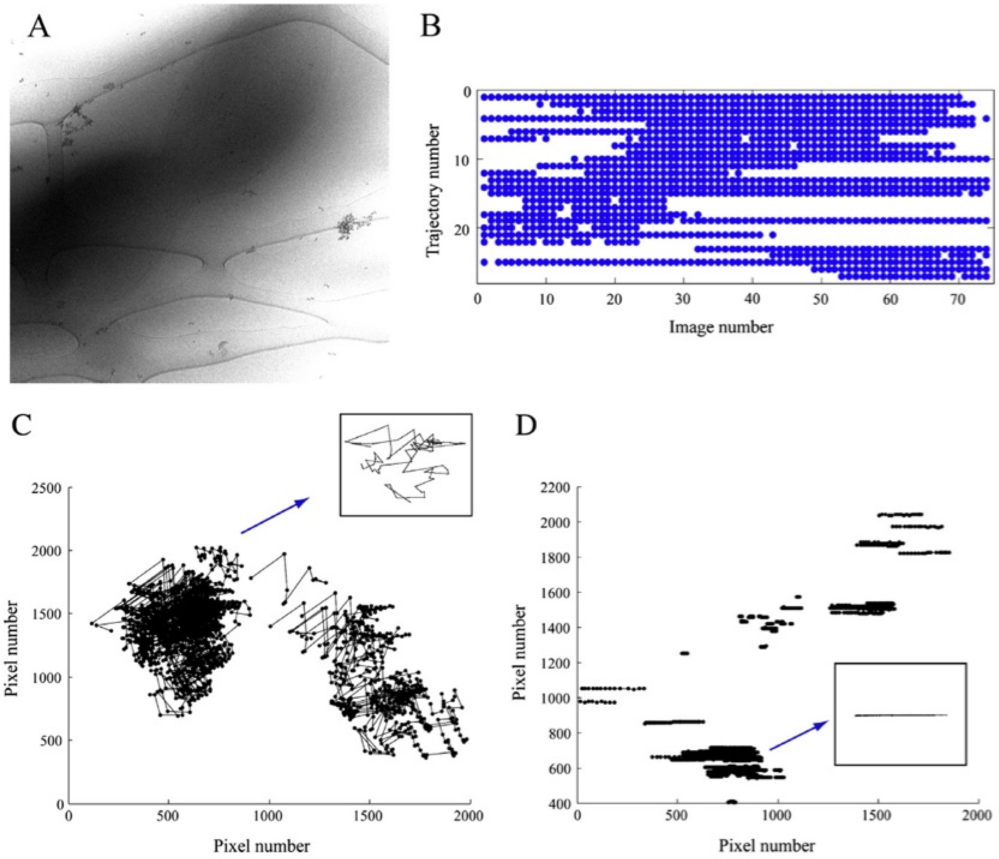


Figure 4.12: RAPTOR performance tracking markers in a Caulobacter crescentus dataset. (A) The -58 deg 2D original projection from the tilt series. The size of the image is 2048 pixels on edge. The cell forms a meniscus of considerable thickness. Both the cell and the grid bars may occlude markers at high tilts. The wider area projected into the images at these angles contains many markers with appropriate SNR, far from the cell, which disappear at lower angles. (B). Trajectories recovered from the tilt series. (C) Overlay of several marker locations in image space (x-y coordinates in pixel number) throughout the raw datasets illustrates the trajectories in the raw data set, (D) trajectories of same markers after RAPTOR automatic alignment. The insets in (C) and (D) show the trajectory of a single marker.

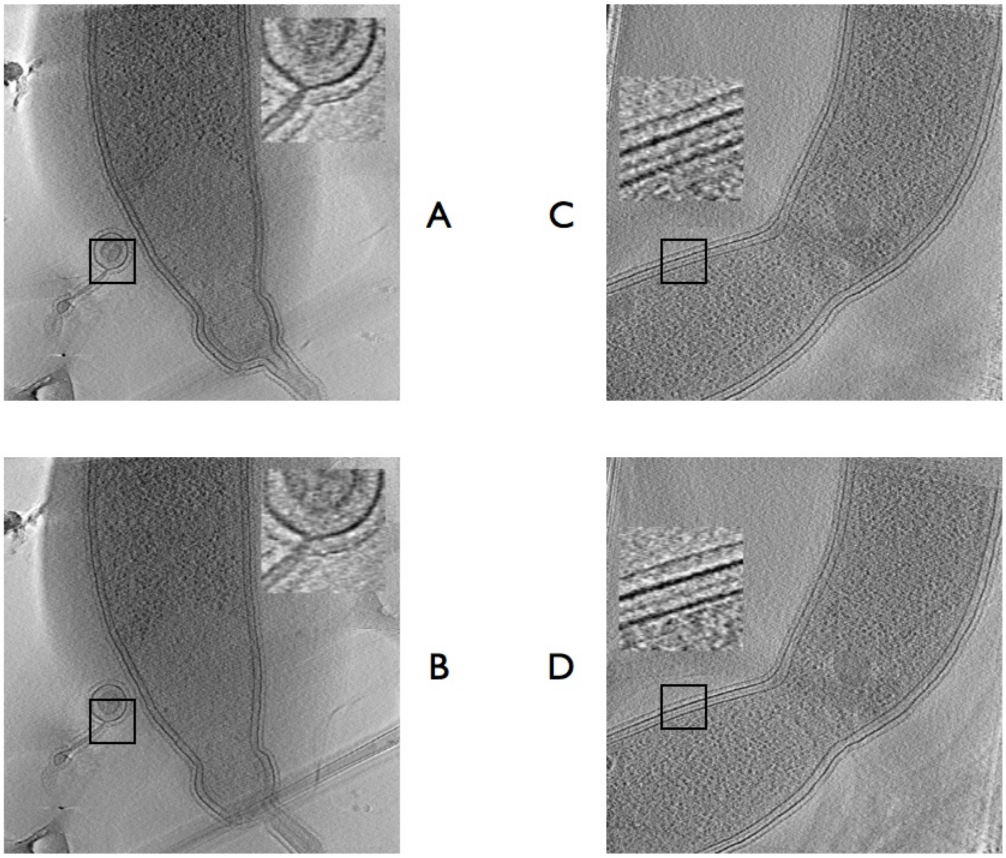


Figure 4.13: Comparison of RAPTOR aligned and manually aligned reconstructions. (A) and (B) - One pixel-thick slices from tomographic reconstructions of a Caulobacter crescentus polar mutant, obtained from a manually aligned dataset and from an automatically aligned dataset, respectively. (C) One pixel-thick slice from a tomographic reconstruction of a dividing Caulobacter crescentus cell after manually aligning the tilt series and (D) After RAPTOR automatic alignment. Reconstruction after marker tracking by hand and human-directed alignment (A and B), and after automatic alignment (C and D), results in equivalent final quality. The images are sections, 900 pixels on edge, from tomographic reconstructions binned from 2048 to 1024 pixels on edge. The boxes shown in the insets measure 84 pixels on edge in the binned images.

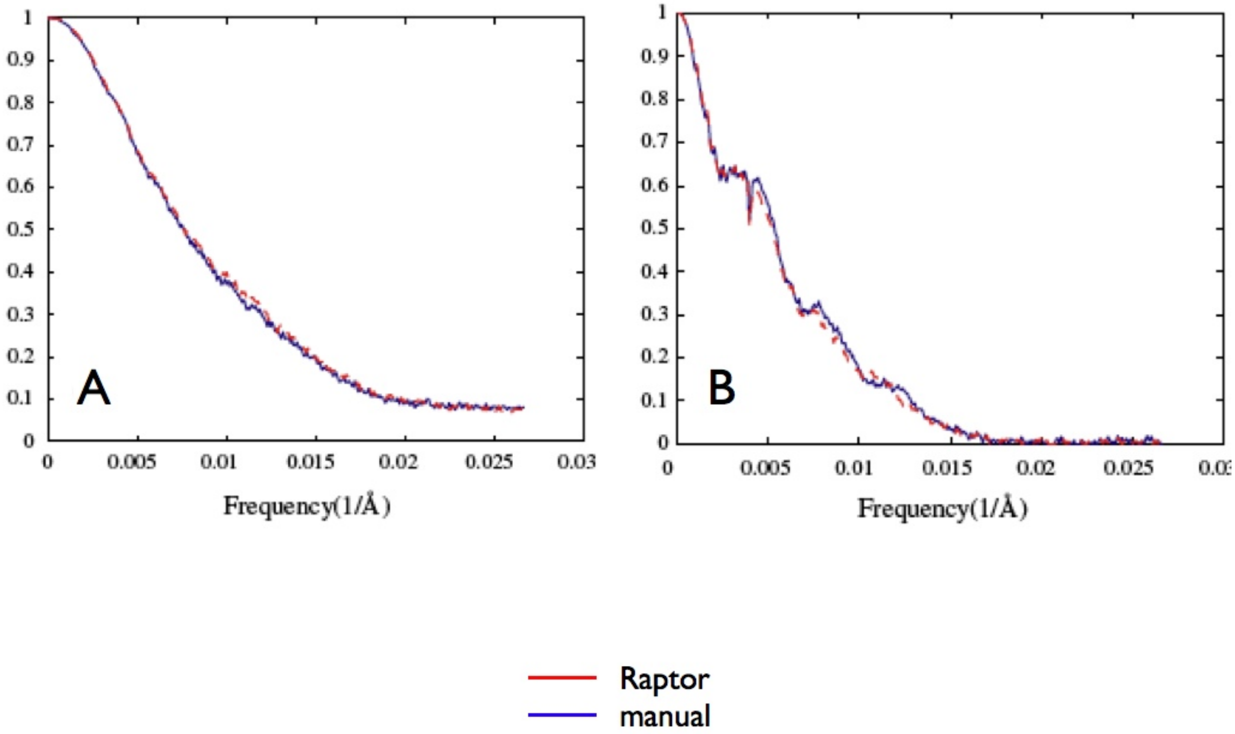


Figure 4.14: Panels (A and B) show quantitative resolution comparisons for tomographic reconstructions of the two datasets above using noise-compensated leave-one-out method of [CGS05] The resolution curves obtained with RAPTOR, in red, and with human-directed alignment, in blue, are equivalent.

Chapter 5

3D Segmentation of Cell Boundaries

Automated precision image registration gives the electron tomography pipeline (Fig. 2.8) high throughput all the way until after volume reconstruction. One of the first tasks after this step is to segment the cell boundary, which ~~is typically~~ currently is a ~~very~~ time consuming, ~~and~~ manual~~ly~~ in~~tensive~~ process. The goal in this chapter is to automatically segment the cell boundary in electron tomograms, thus removing this bottleneck. Once again, we will see how local features contain insufficient information for this recognition task, and how contextual information can be exploited in a joint probabilistic framework (another CRF) to overcome this uncertainty.

5.1 Automatic Cell Boundary Segmentation

The cell boundary is important as a subject of study itself, as well as a reference structure which can be used to navigate from and find other relevant structures inside the cell. Having the segmented boundary thus reduces the size of the volume in which any search for cytoplasmic features (such as ribosomes, filaments, cytoskeletal features) needs to operate. An example of the importance of the cell boundary's surface is found in [CJ05]. In this work, the spatial and temporal behavior of the cell constriction process during division was studied. To analyze the mechanical forces and stresses involved, a quantitative measurement of the cell boundary surface curvature and other geometric quantities was necessary, as shown in in Fig. 5.1.

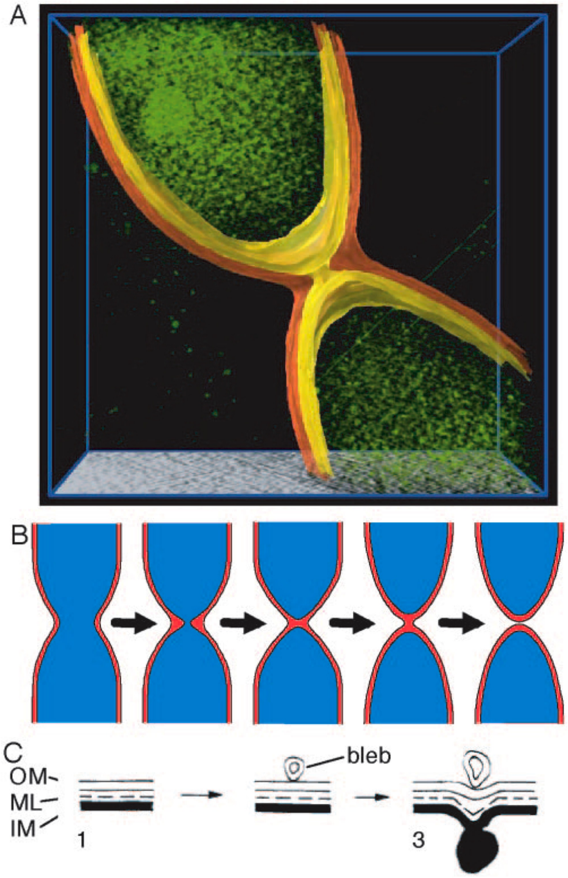


Figure 5.1: Study of the constriction process in cell division (from [CJ05])

In the absence of reliable automatic methods, an expert user today can spend significant amounts of time (ranging from hours to days) manually extracting this boundary by clicking on thousands of points. An adequate solution to automating this task must address the significant challenges of noise and missing wedge artifacts described earlier. Due to these challenges, many existing segmentation and boundary detection techniques that rely on features such as intensity and its gradients are unreliable in these images. Alternatively, we can consider other potential sources

What are you trying to tell me.

of information for boundary classification:1) local appearance features defining a boundary point; 2) local appearance features that are distinct inside and outside the contour and help demarcate the two regions; and 3) shape information of object to be segmented, which can be local, global, or both.

delete?

The key to overcoming these challenges is to adequately specify and model the cell boundary. To this end, we propose an approach called BLASTED (Boundary Localization using Adaptive Shape and TExture Discovery) — an efficient recursive algorithm which extracts the cell boundary in cryo-ET datasets. BLASTED discovers and exploits knowledge about the pattern along the boundary as well as the shape for these types of cells to jointly infer the global set of boundary points and shape under a conditional random field framework which combines and relates all this knowledge.

It is a goal

To understand how to discover the boundary pattern, we examine an example with four patches of the cell boundary obtained from a cross sectional slice, rotated to a common vertical orientation in Fig. 5.2. They have been anisotropically blurred along the tangent to the membrane with a Gaussian kernel. Even after such blurring, their quality is low. As a result, no single one of these patches informs us of the pattern to look for. On the other hand, in the same figure we show the average of ten such patches, which distinctly shows the three layers of the Caulobacter surface — the inner membrane, outer membrane, and S-layer more clearly. This suggests that we can discover the nominal boundary patch texture by non-locally learning from multiple patches distributed throughout the volume. The boundary pattern is one of the most distinguishable features in these images.

The shape of the cells is not generally predictable a priori. However, the cross sections have generally smooth outlines and can be modeled by low order polynomial segments or shapes such as ellipses. Also, the cross sections change slowly throughout most of the cell, such that each inferred cross section can be helpful in guiding an inference of a subsequent cross section. We shall see that these two simple yet specific shape assumptions will aid the inference of the whole cell surface substantially.

We shall see that by combining these two types of cues within an integrated probabilistic framework, BLASTED segmented the cell membrane over an average of 93% of the length of the cell in 19 difficult cryo-ET datasets.

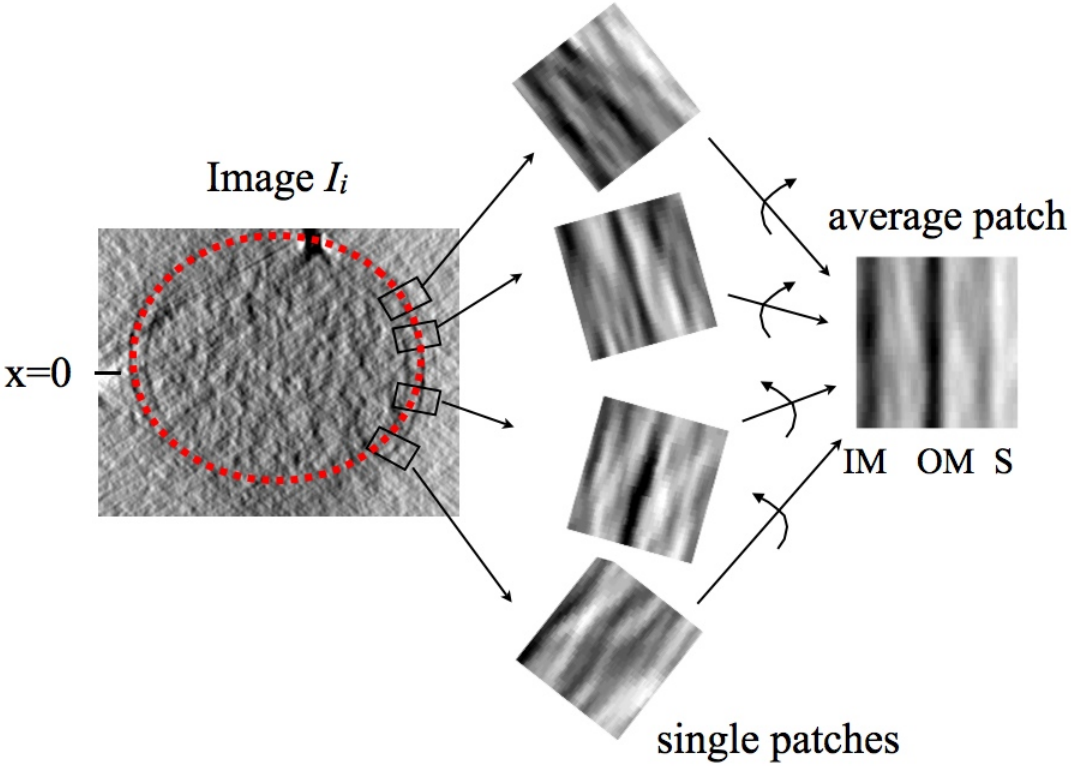


Figure 5.2: Patches obtained along a cell membrane

5.2 Previous Methods

When segmenting the cell boundary, we seek a closed surface which separates the volume into two regions. Broadly, there are two classes of techniques for this task: region classification and boundary classification. In the former, distinct regions having some characteristic are detected, and their boundaries naturally become the output of the segmentation. In the latter, the boundaries between regions are detected directly. Regardless of technique, there are three possible sources of information to guide the segmentation: 1) local appearance features defining a boundary point; 2) local appearance features that are distinct inside and outside the contour and help demarcate the two regions; and 3) shape information of object to be segmented, which can be local, global, or both. Both features and shape information can be 2D or 3D in nature, depending on the method and particular application. Since segmentation is a very developed field that has many applications, an exhaustive survey is beyond the scope of ~~this~~ paper. Instead, we summarize some previous work in these two classes of techniques from the general segmentation community, as well as adaptations to electron tomography below. When evaluating methods for this task, one needs to consider 1) segmentation performance (span and precision of segmented surface), 2) extent of manual supervision required, 3) scalability to large datasets with billions of voxels, and 4) robustness to low signal to noise ratio, false positives and missing data.

Region classification techniques try to identify regions of distinct texture and/or color. Individual pixels might be associated with an appearance model suggesting their class assignment. Pixel pairs in the image are assigned affinity scores that serve to impose soft smoothness constraints on the pixel-to-region assignment. This information can be used to infer boundaries between dissimilar regions. Some methods use energy functions based on graph cuts [KZ03, ZK04, SM00], fuzzy sets [GWBVE08], and separation and deviation functions [Hoc01]. Graph cuts were applied to electron tomography datasets in [FH02]. A different region based method, the watershed transform [VS91, Mey01], is based on intensity and intensity gradients. This method is a region growing technique based on mathematical morphology, What is this and aims to find regions around local maxima in the intensity map. Watershed methods often suffer from the problem of oversegmentation. A 3D watershed method was used in [Vol02] to segment molecular structures like actin filament bundles, as well as Golgi regions in pancreatic cells, which had relatively high quality (well defined boundaries and homogeneous regions) compared to cryo-ET data. When they fail. Region classification techniques have been very successful in many types high quality images.

However, these techniques can have difficulties with cryo-ET cell membrane segmentation, where the texture inside and outside a boundary is usually not consistent and may not be a distinguishing criterion.

Boundary classification approaches seek a boundary based on some feature of the boundary itself, such as intensity gradients, or difference between average intensities of regions separated by that boundary. It is often formulated using an energy function that is then optimized using some type of local search. Methods in this class include active contours, snakes, and level sets [LKGD07, MS89, CV01, CT93, KWT88, Set95]; variants have been developed with both 2D and 3D approaches. An important advantage of having energy functions is the ability to encode shape information, whether it is weak (like smoothness constraints) or strong (adherence to an absolute shape). Some energy based methods have been developed under level set and other frameworks specifically for segmenting subcellular structures in viruses in electron tomography [BSS05], once again on regions with reasonably well defined boundaries and more homogeneity compared to the whole cell cryo-ET. While these methods are known to be subject to local optima and scalability issues, in practice they are used widely with success by carefully choosing starting points, parameters, and preconditioning. A method called watersnakes [NWvdB03] combines watershed and snakes into an edge driven, region growing technique with an energy function, and helps alleviate some of these problems. Recently, [NJ08] used a 3D watersnake method with shape prior on mitochondria and one example of a cell membrane, requiring a rough 3D manual segmentation in the form of hand-labeling multiple slices in the beginning. Two summaries of segmentation methods tried in electron tomography so far can be found in [San07, Fra06].

In images of higher quality, local features based on intensity and intensity gradients such as various edge detection operators [Can86], [Sob70] and many others are used successfully. Many variants of these methods also exist and are summarized in [ZT98, Wah07]. However, in the case of cell membranes in whole cryo-ET datasets, low SNR and missing wedge and other artifacts due to electron scattering and non-linear image formation [LBPE06, FSC97] cause boundaries to be often broken and not contiguous. This causes difficulty for intensity and intensity gradient based local features. Other boundary methods [SB07] can address the edge-fragmentation problem by using a transformation to a space that represents edges compactly and efficiently (such as Hough or Radon space). While they are capable with long straight segments, they have difficulty in segments which are short or have curvature. These methods also return false positives when attempting to detect cell membranes in cryo-ET images.

Experience with the issues described above suggests that boundaries of interest are not adequately described by features based on intensity or texture gradients. An important work in computer vision that addressed this problem for natural images was the Berkeley boundary detector [MFM04], which exploits local orientation as well as texture, intensity, and their variations along the boundary, and uses supervised learning to classify local boundary points. This was an innovative shift from previous boundary recognition techniques, and was designed to detect features locally in natural images which have richer features than cryo-ET datasets. The local features could be then used in a higher level object detection algorithm. Indeed, recent works in object detection by [FFJS08, OPZ08] have used local shape formed by feature points in addition to the features' appearances themselves to achieve better overall detection. Most recently, [HEPK09] used learned global shape and appearance in a probabilistic model to localize landmarks and outlines of objects of a particular class. It is this line of work that forms the basis for our approach.

the point of this is getting lost.

What is the core concept of this line of work

5.3 BLASTED:Joint Discovery of Boundary Points and Shape

The BLASTED algorithm is founded on a holistic description of the cell boundary using its shape and texture. It leverages the idea of discriminatively training boundary point detectors as well as other previous work in boundary recognition to incrementally discover the points and texture of 3D whole cell boundaries, and ultimately the shape of the boundary itself. We accomplish this segmentation by:

1) treating the boundary texture itself as an unknown that is nonlocally learned; 2) proposing candidate points using physical features that depend on estimated shape as well as the learned boundary texture; and 3) robustly classifying candidate points using the learned texture with local and global shape context. In order to keep computational complexity manageable and support large datasets, we partition the 3D inference of the boundary into a recursive 2D slice by slice organization.

As described in section 5.1, the key to overcoming the challenges of noise and missing wedge artifacts in segmentation of cryo ET datasets is to adequately specify and model the cell boundary. The BLASTED framework models the cell boundary by describing the relationships between texture, points and shape of the cell boundary. Physical features of potential boundary points are based on oriented versions of a learned texture, whose orientations depend on the underlying shape. The underlying shape in turn depends on the correct boundary points.

We start by modeling the surface as a set of N slice outlines as depicted in Fig. 5.3. Each outline lies in image $\mathbf{I}_i, i \in \{1 \dots N\}$, which consists of n pixels, and is described by a vector of slice *shape state parameters* $\mathbf{s}_i$. Starting from a hand-labelled outline in one slice, we aim to find all subsequent outlines and a set of points that collectively lie on the cell membrane.

In each image $\mathbf{I}_i$ depicted in Fig. 5.3, we characterize the boundary appearance using a *boundary texture template* $\mathbf{T}_{i,0} \in \mathbb{R}^{q \times r}$; this is an unoriented fixed size $q \times r$ template which resembles an ideal texture on the boundary. This template is a model parameter that is to be learned nonlocally as shown in Fig. 5.2. Oriented versions of this template can be used to help evaluate the appearance of a patch in the image. To this end, we define the *oriented boundary texture template* $\mathbf{T}_{i,\theta}$ to be the boundary texture template $\mathbf{T}_{i,0}$ rotated by an angle θ. We define $M_{i,j} \in \{0, 1\}$ as the indicator variable for whether each pixel $j \in \{1 \dots n\}$ in image $\mathbf{I}_i$ is on the cell boundary. Also, each pixel j in image $\mathbf{I}_i$ has associated with it an orientation $\theta_{i,j}$ (which depends on indicator $M_{i,j}$ and shape $\mathbf{s}_i$). For convenience we denote the set of all pixel orientations in image $\mathbf{I}_i$ as $\boldsymbol{\theta}_i$, and the set of their indicator variables as $\mathbf{M}_i$. The goal of the algorithm is to infer $\mathbf{M}_i$, $\mathbf{s}_i$, and $\boldsymbol{\theta}_i$ from the set of images $\mathbf{I}_i$.

~~We next combine these quantities to construct a detailed model of the cell boundary.~~

5.4 Probabilistic Framework

Our goal is to build a comprehensive holistic model of the cell boundary which combines the appearance, points, and shape information given the evidence — namely $\mathbf{T}_{i,0}$, $\mathbf{M}_i$, $\mathbf{s}_i$, and $\boldsymbol{\theta}_i$ given the set of images $\mathbf{I}_i$. We will combine all these quantities into a CRF. But first we must define two quantities more specifically — boundary appearance $\mathbf{T}_{i,0}$ and boundary slice shape $\mathbf{s}_i$.

5.4.1 Modeling Boundary Appearance

The first component of our membrane description is the underlying boundary pattern. As described in section 5.1, the underlying pattern of the cell membrane is more complicated than a single edge, and is represented by an estimated texture template $\mathbf{T}_{i,0}$. This estimate is not readily obtainable by looking at one point on the boundary, but rather by averaging over multiple such points as depicted in Fig. 5.2. We will see how this average is taken and incrementally improved over time as the algorithm proceeds in section 5.5.3. The template $\mathbf{T}_{i,0}$ will be a parameter in our CRF.

5.4.2 Modeling Boundary Shape

The second component of our membrane description is the shape to which boundary points adhere. We would like to exploit the natural characteristics of cell shapes (e.g. smoothness, slowly changing curvature) as much as possible to make the description as specific and as discriminative as possible. We model the boundary shape as an interrelated sequence of cross sectional shapes as described in section 5.3. At each slice i, we define a vector of shape state parameters $\mathbf{s}_i$, which contains information for slice i and possibly previous slices as sufficient statistics to facilitate prediction of future slice shapes. The first slice is hand labeled and therefore $\mathbf{s}_1$ is given as a starting slice for the algorithm.

We first focus on the shape parameterization for one slice. We consider two types of cross sectional shapes: ellipses and cubic B-splines. Ellipses are very specific but model cross sections of many cells. B-splines are general and are suitable for cells whose cross sections are not elliptical.

Ellipses

A simple shape that models many cells' cross sections well is an ellipse. Ellipses are fully described in two dimensions with only five parameters x_i^e,y_i^e (center), θ_i^e (orientation), and c_i^{e1}, c_i^{e2} (major and minor axes). An ellipse can be represented in two forms — either with these five parameters directly:

$$\mathbf{e}_i^T = \left[\begin{array}{ccccc} x_i^e & y_i^e & \theta_i^e & c_i^{e1} & c_i^{e2} \end{array} \right] \tag{5.1}$$

or as a quadratic polynomial:

$$f(x,y) = ax^2 + by^2 + cxy + dx + fy + g = 0 \tag{5.2}$$

As we shall see, fitting of ellipses to point data is readily and robustly achievable. Therefore, it is desirable to use ellipses to model cell cross sections when we can.

B-splines

When the cross sectional shape is not elliptical, we use a more general shape model — cubic B-splines. Cubic B-splines are ubiquitous in computer aided geometric design (CAGD) and computer graphics. An n^{th} order B-spline curve is a concatenation of multiple n^{th} order polynomial segments. For cubic B-splines, n is 3. Such curves can be represented in two forms — piecewise polynomials (intrinsic), or with a pair of component functions $(x(t), y(t))$ which are functions of a parameter t (parameteric).

In both views, the breakpoints at which the polynomial segments connect are called *knots*. An example B-spline curve is depicted graphically in Fig. 5.4.

In the intrinsic representation, the breakpoints and polynomial coefficients of each segment are specified. In the parametric representation, the curve is represented in terms of its parametric component functions:

$$\begin{aligned} &\mathbf{P}(t) = (x(t), y(t)) \\ &x(t), y(t) \in \mathbb{R} \end{aligned} \tag{5.3}$$

It can be shown that in the parametric representation, each point $\mathbf{P}(t)$ can be expressed as a linear combination of a fixed number N_c of control points $\mathbf{P}_i$ which define the curve:

$$\begin{aligned} &\mathbf{P}(t) = \sum_{k=1}^{N_C} \mathbf{P}_k B_k(t), \\ &\mathbf{P}(t), \mathbf{P}_k \in \mathbb{R}^2 \end{aligned} \tag{5.4}$$

The weights of the control points in this linear combination or functions over t called *B-spline curve basis functions* which are determined by the B-spline order n and its continuity constraints at the knots. The number of control points N_C depends on the number of knots in the curve, as well as the continuity constraints of the curve at the knots. This number determines the complexity of the curve in our formulation. It is lower bounded by a minimum complexity needed to model the shape, and upper bounded by the quality of the data used to fit the curve. Specifically, if N_C is chosen to be too high for a given quality of data, overfitting may occur.

B-splines are very versatile and can model very general curves and surfaces of varying degrees of complexity by merely choosing different values of N_C. In order to avoid overfitting, the complexity of the B-spline must be properly constrained. As we shall see, the control points $\mathbf{P}_{i,k}$ will play a crucial role in this process. Specifically, the surface is modeled as a sequence of B-spline curves with a consistent set of control points across the slices, and the context of control points from previous slices will be used to help find the control points in a slice being inferred. This concept is depicted graphically in Fig. 5.5 (b).

B-splines are also able to model 3D surfaces in an analogous fashion to 2D curves. The parametric representation for a B-spline surface is given by

$$\begin{aligned} &\mathbf{P}(u,v) = \sum_{k=1}^{N_{C,1}} \sum_{l=1}^{N_{C,2}} \mathbf{P}_{k,l} B^s_{k,l}(u,v), \\ &\mathbf{P}(u,v), \mathbf{P}_{k,l} \in \mathbb{R}^3 \end{aligned} \tag{5.5}$$

where the $B^s_{k,l}(u,v)$ are *B-spline surface basis functions*, and are separable into products of B-spline curve basis functions:

$$B^s_{k,l}(u,v) = B_k(u)B_l(v) \tag{5.6}$$

As a result, modeling the surface as a parametric B-spline surface with Eq. 5.5 can be interpreted as being the same as modeling the surface as a collection of parametric B-spline curves with a fixed, consistent set of control points as shown in Fig. 5.5 (b).

For more on B-splines the reader is referred to [PT97, Far02].

3D Shape as the Evolution of a 2D Shape

Having defined the shape parameterization for a cross sectional slice, we are now ready to define the shape state parameters $\mathbf{s}_i$, which contain sufficient statistics to predict future slice shapes.

In the case of ellipses, $\mathbf{s}_i$ is a combination of $\mathbf{e}_i$ and its difference with $\mathbf{e}_{i-1}$:

$$\Delta\mathbf{e}_i = \mathbf{e}_i - \mathbf{e}_{i-1} \tag{5.7}$$

$$\mathbf{s}_i = \begin{bmatrix} \mathbf{e}_i \\ \Delta\mathbf{e}_i \end{bmatrix}, \mathbf{s}_1 = \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{0} \end{bmatrix} \tag{5.8}$$

In the case of B-splines, $\mathbf{s}_i$ is the collection of all previous control points $\mathbf{P}_{1:i,j}$:

$$\mathbf{s}_i = \begin{bmatrix} \mathbf{P}_{1,1}.....\mathbf{P}_{1,j}.....\mathbf{P}_{1,N_c} \\ . \\ . \\ . \\ \mathbf{P}_{i,1}.....\mathbf{P}_{i,j}.....\mathbf{P}_{i,N_c} \end{bmatrix} \tag{5.9}$$

In both these cases, the first slice is hand labeled and $\mathbf{s}_1$ is given as a starting slice for the algorithm. We shall see how these shape states will be used to predict future slice shape parameters.

5.4.3 Combining Shape and Appearance in a Conditional Random Field

Having defined the relevant variables $\mathbf{I}_i$, $\mathbf{M}_i$, $\boldsymbol{\theta}_i$, $\mathbf{s}_i$, and parameter $\mathbf{T}_{i,0}$; we can now formally describe the membrane boundary probabilistically in a CRF. In this framework, the state variables of the model are $\mathbf{M}_i$, $\boldsymbol{\theta}_i$, and $\mathbf{s}_i$. The evidence are the images $\mathbf{I}_i$, and the first outline $\mathbf{s}_1$ which is given since it is hand labelled. The texture templates $\mathbf{T}_{1:N,0}$ are model parameters that are learned incrementally. The CRF for BLASTED is depicted in Fig. 5.6.

We can express the joint conditional probability of the state variables given the observations in this CRF:

$$P(\mathbf{s}_{2:N}, \mathbf{M}_{1:N}, \boldsymbol{\theta}_{1:N} | \mathbf{I}_{1:N}, \mathbf{s}_1; \mathbf{T}_{1:N,0}) =$$
$$\frac{1}{Z} \prod_{i=2:N} \phi_s(\mathbf{s}_i, \mathbf{s}_{i-1}) \prod_{j=1:n} \phi_x(\mathbf{s}_i, M_{i,j}) \phi_a(\mathbf{s}_i, M_{i,j}, \theta_{i,j}) \tag{5.10}$$

where Z is a normalizing constant. The first factor ϕ_s in (5.10) is the global shape transition model, which reflects the compatibility of the 2D shape at slice i with the 2D shape at slice $i-1$. The second and third factors ϕ_x and ϕ_a form the observation model; they reflect the likelihood of seeing the images $\mathbf{I}_i$ with a hypothesis shape $\mathbf{s}_i$ and their pixels classified by $M_{i,j}$. We next describe the different factors in this CRF.

5.4.4 Shape Transition Model — 3D Shape as the Evolution of a 2D Shape

The role of the shape transition model factor is to predict the next curve given the previously inferred curves, which are described by the shape state parameters $\mathbf{s}_i$ defined in section 5.4.2. In the case of ellipses, this means predicting the next set of ellipse parameters from the set of ellipses inferred so far. In the case of B-splines, it means predicting the next set of (consistent) control points from all the control points inferred so far.

For the case of ellipses, we take $\Delta\mathbf{e}_i$ and $\Delta\mathbf{e}_{i-1}$ from (5.7) to be jointly Gaussian. Therefore, the conditional distribution of $\mathbf{e}_i | \mathbf{s}_{i-1}$ is also Gaussian, and can be directly written as:

$$\mathbf{e}_i | \mathbf{s}_{i-1} \backsim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{5.11}$$

have you defined these? I think so but could not find it in a quick scan where

$$\begin{aligned}\boldsymbol{\mu}_i &= E[\mathbf{e}_i|\mathbf{s}_{i-1}] = \mathbf{e}_{i-1} + \hat{\boldsymbol{\Delta}}_{e,i} && (5.12)\\ \boldsymbol{\Sigma}_i &= E[\mathbf{e}_i\mathbf{e}_i^T - E[\mathbf{e}_i]E[\mathbf{e}_i^T]|\mathbf{s}_{i-1}] \\ &= \boldsymbol{\Sigma}_1 - \boldsymbol{\Sigma}_2\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_2^T && (5.13)\end{aligned}$$

in which

$$\begin{aligned}\hat{\boldsymbol{\Delta}}_{e,i} &= E[\boldsymbol{\Delta}\mathbf{e}] + \boldsymbol{\Sigma}_2\boldsymbol{\Sigma}_1^{-1}(\boldsymbol{\Delta}\mathbf{e}_{i-1} - E[\boldsymbol{\Delta}\mathbf{e}]) && (5.14)\\ \boldsymbol{\Sigma}_1 &= E[\boldsymbol{\Delta}\mathbf{e}\boldsymbol{\Delta}\mathbf{e}^T] - E[\boldsymbol{\Delta}\mathbf{e}]E[\boldsymbol{\Delta}\mathbf{e}^T] && (5.15)\\ \boldsymbol{\Sigma}_2 &= E[\boldsymbol{\Delta}\mathbf{e}\boldsymbol{\Delta}\mathbf{e}_{-1}^T] - E[\boldsymbol{\Delta}\mathbf{e}]E[\boldsymbol{\Delta}\mathbf{e}^T] && (5.16)\end{aligned}$$

If previously segmented datasets are available, the values $E[\boldsymbol{\Delta}\mathbf{e}]$, $\boldsymbol{\Sigma}_1$, and $\boldsymbol{\Sigma}_2$ are learned from them, and stored as model coefficients. This is done by gathering statistics on $\boldsymbol{\Delta}\mathbf{e}$ and $\boldsymbol{\Delta}\mathbf{e}_{-1}$ (changes in ellipse parameters between adjacent slices, and differences in these changes between adjacent slices) from the correctly labelled dataset, and forming the empirical mean vector and covariance matrices for use in Eqs. (5.12 -5.16). The shape transition model factor for the case of ellipses for which we have these statistics is a function of $\mathbf{e}_i$ and is taken to be:

$$\phi_s(\mathbf{s}_i, \mathbf{s}_{i-1}) = e^{(\mathbf{e}_i - \boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{e}_i - \boldsymbol{\mu}_i)} \qquad (5.17)$$

When segmenting new datasets for which similar correctly segmented datasets are not available, we have found that simply using the inferred shape from slice $i-1$ to predict the shape in slice i can also work well, and the results can be used as a correctly segmented dataset for learning in a boostrapped fashion. The shape transition model factor in this case is simply:

$$\phi_s(\mathbf{s}_i, \mathbf{s}_{i-1}) = \delta(\mathbf{e}_i - \mathbf{e}_{i-1}) \qquad (5.18)$$

For the case of B-splines, predicting the next shape in slice i is the same as predicting the next set of control points $\mathbf{P}_{i,j}$ for all j. Each control point $\mathbf{P}_{i,j}$ is predicted with the estimate $\hat{\mathbf{P}}_{i,j}$ by fitting another cubic B-spline to the previously obtained control points $\mathbf{P}_{1:i-1,j}$. It is possible to predict each control point in this fashion since there is a consistent set of control points across slices that are corresponding to each other, and is the main reason for this design choice. In essence, the B-spline curve basis functions serve as regressors for predicting the next set of control points from their predecessors. In this case, the shape transition model factor becomes:

$$\phi_s(\mathbf{s}_i, \mathbf{s}_{i-1}) = e^{-\sum_{j=1}^{N_C} ||P_{i,j} - \hat{P}_{i,j}||^2} \tag{5.19}$$

5.4.5 Observation model — shape noise

The second factor ϕ_x in (5.10) quantifies shape noise in the observation model, that is, deviation of points j on the boundary in image $\mathbf{I}_i$ from shape $\mathbf{s}_i$. We model this as a Gaussian rolloff with distance from $\mathbf{s}_i$ for points which are on the membrane, and uniform for other points:

$$\phi_x(\mathbf{s}_i, M_{i,j}) = \begin{cases} e^{-d(x_{i,j}, \mathbf{s}_i)^2/(2\sigma_x^2)} & M_{i,j} = 1 \\ 1 & M_{i,j} = 0 \end{cases} \tag{5.20}$$

where $x_{i,j}$ is the location of pixel j in image $\mathbf{I}_i$, $d(x, \mathbf{s}_i)$ is the Euclidean distance from point x to shape $\mathbf{s}_i$, and σ_x is an input parameter to the algorithm which allows for variation of the locations of $x_{i,j}$ around the shape $\mathbf{s}_i$. This parameter can be interpreted as the standard deviation allowed between points on the boundary and the nominal shape $\mathbf{s}_i$. A smaller value for σ_x penalizes deviations from the shape $\mathbf{s}_i$ more heavily. We found that using a relatively large value of 100 pixels (120 nm) for this parameter worked well across all experiments (all of which had the same magnification) , and used this value for all the results. For comparison, the thickness of the outer membrane which we are trying to localize is roughly 10 nm.

5.4.6 Observation model — physical similarity

The third factor ϕ_a in (5.10) measures physical similarity of a patch centered around $x_{i,j}$ and oriented by $\theta_{i,j}$ in image $\mathbf{I}_i$ with the learned oriented texture patch $\mathbf{T}_{i,\theta_{i,j}}$. One possibility is to define it as the cross correlation of $\mathbf{T}_{i,\theta_{i,j}}$ with the patch, but this may result in many false positives. Our experiments showed that even applying a missing wedge filter to $\mathbf{T}_{i,\theta_{i,j}}$ did not improve cross correlation results or reduce false positives in practice. To address this problem, we treat the physical similarity as a classification problem in a supervised learning framework. In this framework, we consider cross correlation with the oriented texture patch $\mathbf{T}_{i,\theta_{i,j}}$, distance of $x_{i,j}$ from shape $\mathbf{s}_i$, and spatial context of nearby candidate points. We implement ϕ_a as a boosted classifier using AdaBoost ([FHT00, HTF03]), which uses a weighted sum of results from a committee of M weak classifiers. Specifically, each weak classifier $G_m(f), m = 1 \ldots M$ takes a vector of features f, and returns a class label $G \in \{-1, 1\}$ as its decision. The boosted classifier takes a weighted sum of the committees decisions, and uses their sign as a final decision:

$$G(f) = sign(\sum_{m=1}^{M} \alpha_m G_m(f)) \tag{5.21}$$

The weights α_m are calculated during the supervised training of the boosted detector. We modify this usage slightly to obtain a confidence level rather than a hard decision. To this end we can apply the *sigmoid function*:

$$\sigma(v) = \frac{1}{1 + e^{-v}} \tag{5.22}$$

on a real number v to get a score between 0 and 1.

We can now consider getting a confidence level for each pixel being on the boundary. For each pixel $x_{i,j}$ we take a set of features f, and form the weighted sum in ~~to the sum~~ in Eq. 5.21. We than convert this sum to a confidence level that $M_{i,j} = 1$

(that is, that $x_{i,j}$ is on the boundary) using Eq. 5.22. The features for this classifier will be described in section 5.5.1. We can now define the physical similarity factor ϕ_a:

$$\phi_a(\mathbf{s}_i, M_{i,j}, \theta_{i,j}) = \frac{1}{1 + e^{-\sum_{m=1}^{M} \alpha_m G_m(f)}} \tag{5.23}$$

The variables, parameters, structure, and factors for the overall cell boundary CRF have now been defined. We next consider how to run inference over this CRF to solve for the cell boundary.

5.5 Inference in BLASTED

Having defined the CRF for BLASTED, we now seek to perform a MAP inference on this CRF of the unknown variables given the evidence. The steps of this inference are described next, and are outlined graphically in Fig. 5.3. The inference seeks the maximizing values of $\mathbf{s}_i$, $\boldsymbol{\theta}_i$, and $\mathbf{M}_i$ for each slice i, using the learned template $\mathbf{T}_{i-1,0}$ starting from one hand labelled slice:

$$\hat{\mathbf{s}}_i, \hat{\boldsymbol{\theta}}_i, \hat{\mathbf{M}}_i = \arg \max_{\mathbf{s}_i, \boldsymbol{\theta}_i, \mathbf{M}_i} \left(P(\mathbf{s}_i, \boldsymbol{\theta}_i, \mathbf{M}_i | \mathbf{I}_i, \mathbf{s}_{i-1}; \mathbf{T}_{i-1,0})\right) \tag{5.24}$$

The inference takes place sequentially in a slice by slice fashion. As in most sequential inference in temporal models, there is a time update (prediction) step and a measurement update step as described in section 3.3.2.

In the time update step, shape $\hat{\mathbf{s}}_i$ is predicted from shapes inferred so far using the global shape transition model as described in section 5.4.4. In the measurement update step, the algorithm does block coordinate ascent on the observation model. In each iteration, it first uses the fixed shape $\hat{\mathbf{s}}_i$ to seek candidate points using the latest estimate of the boundary template. It then estimates $M_{i,j}$ by maximizing appearance terms of the observation model with a trained boosted classifier. Finally, it fixes $M_{i,j}$, and uses the candidate points for which $M_{i,j} = 1$ to fit the refined shape $\mathbf{s}_i$, thus minimizing the shape noise terms of the observation model. We discuss the details of maximizing the appearance term in section 5.5.1 and shape fitting in section 5.5.2. This completes one iteration of the coordinate ascent and measurement update. To guarantee an ascent direction, the observation model is evaluated to see if it — what does this refer to. has increased in this step, in which case the iteration gets accepted. Otherwise, the iteration terminates. The inferred shape $\mathbf{s}_i$ gets assigned to $\hat{\mathbf{s}}_i$, and the procedure is repeated until the number of outlier points to the inferred ellipse is below some threshold, or a maximum number of iterations is reached. Finally, if the average texture of slice i, $\mathbf{t}_{i,0}$, is sufficiently similar to $\mathbf{T}_{i-1,0}$ (judging by the normalized cross correlation between the two) and it completely lies within the field of view, template $\mathbf{T}_{i,0}$ is updated for the next slice as follows:

$$\begin{aligned} \mathbf{T}_{i,0} &= \alpha * \mathbf{t}_{i,0} + (1-\alpha) * \mathbf{T}_{i-1,0}, i \in 2 \dots N \\ \mathbf{T}_{1,0} &= \mathbf{t}_{1,0} \end{aligned} \tag{5.25}$$

effectively updating the estimate of the boundary texture. More details of the estimation of $\mathbf{T}_{i,0}$ are explained in section 5.5.3

~~We will now describe the details of the measurement update step~~.

5.5.1 Measurement Update — Boundary Point Classification

The boundary point classification happens in two steps: selection of candidate points and classification of the candidate points.

Selection of candidate points for classification

Since most pixels in any image are far from boundaries, it would be desirable to limit the inference to a small subset of the pixels which act as candidate points. In this section we describe how we find candidate points in an image $\mathbf{I}_i$ starting from some hypothesis shape $\hat{\mathbf{s}}_i$ and underlying boundary texture template $\mathbf{T}_{i,0}$. The estimation of $\mathbf{T}_{i,0}$ is described in section 5.5.3.

We begin by sampling uniformly at random a fixed number m (in our experiments 60) of reference points $x^r_{i,k}, k \in 1 \ldots m$ along $\hat{\mathbf{s}}_i$ (like the template estimation). The set of all $x^r_{i,k}$ is called $\mathbf{x}^r_i$ for convenience. The orientation at $x^r_{i,k}$, or $\theta_{i,k}$ is taken to be the tangent of shape $\hat{\mathbf{s}_i}$ at $x^r_{i,k}$. We form the oriented boundary texture template $\mathbf{T}_{i,\theta_{i,k}}$. We then take *interest patches*, $\mathbf{A}_{i,k} \in \mathbb{R}^{u \times v}$ as large fixed size $u \times v$ (in our experiments 101×101 pixels) patches centered around $x^r_{i,k}$ in image $\mathbf{I}_i$. Finally, we run normalized cross correlation of $\mathbf{T}_{i,\theta_{i,k}}$ with all points in $\mathbf{A}_{i,k}$, apply a decay factor based on distance of the point from $\hat{\mathbf{s}}_i$, obtaining a score for each point in $\mathbf{A}_{i,k}$ (this score is explained in more detail ~~more~~ below ~~in Section 5.5.1~~). We keep only the points that score more than a fixed threshold (ncc_{min}, a parameter to the algorithm), resulting in n_i candidates, each one called $x^c_{i,j}, j \in 1 \ldots n_i$. The union of all such points in image $\mathbf{I}_i$ is called $\mathbf{x}^c_i$ for convenience. This operation is depicted graphically in Fig. 5.3. The candidate points are now ready to be considered for classification.

Classification of candidate points with appearance features and local spatial context

It remains to define the potential associated with the $M_{i,j}$ variables. We use a supervised learning framework to learn this potential, so as to try to reduce false positives. To accomplish this, we use local spatial context within the cross correlation result patch.

We expect that cross correlation with a template like ours will produce a map with several local maxima, only one of which is correct. To find the correct maximum, we spatially cluster the thresholded cross correlation results as shown in Fig. 5.7. A small region of image $\mathbf{I}_i$ is shown in Fig. 5.7(a). The thresholded cross correlation yields the interest points $x^c_{i,j}$ in Fig. 5.7(b). The points on the outer membrane (middle line) are the correct ones. We refer to a point's rotated x coordinate as the coordinate along the horizontal axis of its corresponding point in the rotated patch. We spatially cluster the points on their rotated x coordinates into three clusters to get the result in Fig. 5.7(c). We then set up a boosted classifier that uses statistics from a candidate interest point's cluster, as well as the information for the interest point itself, to make classification decisions. Boosted detectors are based on simple classifiers (e.g. a decision tree) whose parameters are iteratively learned from a training set of multiple features [FHT00]. They have been successful at taking many features (often called a "bag of features") and learning through a training procedure how to classify test data which provides those same features. Our classifier uses seven features (f_1 to f_7), and returns a confidence that $M_{i,j} = 1$.

The features f_1 to f_3 are for a given candidate interest point $x^c_{i,j}$. They are intended to measure physical similarity of the interest patch $\mathbf{A}_{i,j}$ to the rotated template $\mathbf{T}_{i,\theta^c_{i,j}}$, as well as provide hints to help eliminate false peaks from this information. Feature f_1 is the cross correlation with $\mathbf{T}_{i,\theta^c_{i,j}}$ attenuated by a Laplacian distance rolloff $\exp(-k_a d(x^c_{i,j}, \mathbf{s}_i))$, where k_a is an input parameter to the algorithm and where $d(x, \mathbf{s}_i)$ is the Euclidean distance from point x to shape $\mathbf{s}_i$:

$$f_1 = NCC(\mathbf{A}_{i,j}, \mathbf{T}_{i,\theta^c_{i,j}}) \exp(-k_a d(x^c_{i,j}, \mathbf{s}_i)) \tag{5.26}$$

The parameter k_a acts as a shape stiffness parameter, as it determines the rate at which the likelihood of a point being on the boundary decreases with distance from a predicted shape. To avoid the inference being driven by shape more than the appearance features, we would like this rate to be small. Our experiments worked with this parameter being mostly a value of 0.008, and only once being increased as high as 0.04, reflecting an effective distance radius of 125 and 25 pixels respectively in a slice. Feature f_2 is a similarly attenuated cross correlation with a cropped version of $\mathbf{T}_{i,\theta^c_{i,j}}$ in which the middle one third of the template has been retained, which has the effect of detecting lines with orientation $\theta^c_{i,j}$. Finally, feature f_3 is the ratio f_1/f_2. This ratio is intended to normalize for different imaging conditions and line strengths, and separate incorrect from correct local maxima from cross correlation of f_1 with the candidate patch.

The features f_4 to f_7 are for the cluster to which a candidate interest point belongs. They are intended to provide local spatial context to help remove any remaining false positives by estimating which of the clusters (if any) are the most likely. In following the bag of features approach, we seek to define multiple measures to do this estimation. We define these features as follows. Feature f_4 is the sum of f_1 (total area) for all the points. Feature f_5 is the mean of f_1 in the cluster. Features f_6 and f_7 are relative peaks of the three clusters. They use the sum of the top 5 values of f_1 in each cluster, called s_5. Specifically, feature f_6 is the ratio of s_5 for an interest point's cluster to the smallest s_5 of the three clusters (so that its minimum value is one). Feature f_7 is the difference between s_5 of the interest point's cluster and the largest s_5 of the three clusters (so that its maximum value is zero).

Using this boosted classifier, we classify the interest points given in Fig. 5.7(c) to get the set in Fig. 5.7(d). Note that while we may misclassify good points as not being on the boundary, we are still left with a sufficient number of good points from which shape can be inferred. This boosted classifier is trained a priori, using a set of manually-labelled boundary points. This manual labeling of points is done per cluster, and is quick. Typically we train with more than 2000 points in less than ten minutes. Training need not be done for each dataset, but only on a few points in similar datasets.

5.5.2 Measurement Update — Shape Fitting

Once the boundary points are classified, they can be used to fit the cross sectional shape. We now describe the fitting methods for ellipses and cubic B-splines.

Ellipses

The first shape we consider is an ellipse. Fitting ellipses requires the determining of the polynomial coefficients, or the ellipse parameters as specified in section 5.4.2. It is more straightforward to fit polynomial coefficients using least squares, but the coefficients must meet certain conditions to satisfy an ellipse, and the constraints on the coefficients are nonconvex. Various approaches have been used to accomplish this, and a direct least squares method is described in [FPF99]. This method formulates an equivalent constrained minimization problem where the constraint is quadratic, and then solves by minimizing the Lagrangian. It is robust and ellipse specific, and works well in the presence of unreliable data.

B-Splines

When the cross sections are non-elliptical, we use cubic B-splines with a parametric representation as described in section 5.4.2. Such a representation has a linear combination of control points (Eq. 5.4). The number of control points N_C depends on the number of piecewise polynomial segments in the curve.; ~~Therefore~~ highly complex curves can be modeled by choosing a sufficiently large N_C. However, too large a choice for N_C can result in overfitting. Therefore, the choice of N_C represents an inherent bias-variance tradeoff.

In order to fit a B-spline curve to the boundary points, we model the boundary points as points along the B-spline with additive shape noise:

$$\begin{aligned} &\mathbf{x}_{i,j}(t_j) = \mathbf{P}(t_j) + \boldsymbol{\epsilon}_j = \sum_{k=1}^{N_C} \mathbf{P}_k B_k(t_j) + \boldsymbol{\epsilon}_j \\ &\mathbf{x}_{i,j}(t_j), \mathbf{P}(t), \mathbf{P}_k, \boldsymbol{\epsilon}_j \in \mathbb{R}^2 \end{aligned} \tag{5.27}$$

When fitting the B-spline, our aim is to minimize $\boldsymbol{\epsilon}_j$. At the same time, we aim to constrain the B-spline appropriately to match the inherent complexity of the shape to be fitted and prevent overfitting. An intuitive way to constrain (relax) the B-spline is to remove (add) control points. While this approach is common and makes sense for curve generation, it represents a combinatorial problem in the case of curve fitting. Alternatively, we can maintain a fixed number of control points N_C with consistently placed knots (and therefore constant basis functions $B_k(t)$), and constrain them through regularization. The control points $\mathbf{P}_k$ provide a geometrically meaningful quantity on which regularization can be carried out.

With these goals in mind, we define the following minimization problem to perform the fit of the B-spline curve:

$$\underset{\mathbf{P}_k, t_j}{\text{minimize}} \quad \sum_{k=1}^{N_C} \sum_{j=1}^{M} ||\mathbf{P}_k B_k(t_j) - \mathbf{x}_{i,j}||^2 + \lambda \sum_{k=1}^{N_C} (||\Delta(\mathbf{P}_k - \mathbf{P}_{(k-1)})||^2 + c||\Delta \mathbf{P}_k||^2) \tag{5.28}$$

where $\Delta\mathbf{P}_k = \mathbf{P}_{i,k} - \mathbf{P}_{i-1,k}$ is the difference vector between two corresponding control points from slice i and slice $i-1$. Therefore in this minimization there is a data term and a regularization term to constrain the shape. The regularization is a smoothing term over $\Delta\mathbf{P}_k$ and $\Delta(\mathbf{P}_k - \mathbf{P}_{(k-1)})$and will be explained shortly. When the values of t_j are known, both terms are quadratic over $\mathbf{P}_k$. In this case the objective function is an L_2 regularized least squares objective and therefore convex in $\mathbf{P}_k$.

However, unfortunately t_j is not known, and the B-spline basis functions $B_k(t)$ are nonconvex in t. Therefore the overall objective function is in general nonconvex. It is searched using block coordinate descent, in which the values t_j are fixed and P_k are optimized, and then P_k are fixed and t_j are optimized. This procedure is continued iteratively until convergence. The initial guesses for t_j are obtained through the chord length method ([PT97]). The first step in the iteration is carried out by standard least squares. The second step in the iteration is carried out with nonlinear least squares using the B-spline basis functions and their derivatives, which are precomputed.

The regularization term has a constant λ and involves a smoothing term $\sum_{k=1}^{N_C}(||\Delta(\mathbf{P}_k - \mathbf{P}_{(k-1)})||^2 + c||\Delta\mathbf{P}_k||)^2$. The goal of this term is to penalize deviations of individual control points from their predicted locations, as well as deviations of the overall set of control points from the predicted set of control points.; ~~Therefore~~ there is regularization of control points and pairwise differences of control points between slices. Both types of regularization are useful because of cases of incomplete fitting data. If fitting data is present near one control point but not near another, regularization on individual control points may cause one control point to move from its predicted value while the control point without data will tend to stay fixed. The net effect is a shape distortion. By penalizing changes in pairwise distances between control points, a degree of rigidity is maintained, and control points are encouraged to move together. Together, these two types of regularization make the fitting robust to missing data and outliers in the classified boundary points.

What about C?

An important parameter for this fitting is the regularization constant λ. While it is possible to make this constant an input parameter to the algorithm, a fixed value for λ would cause a variable strength of the regularization term since the strength of the data term varies with the data at each slice. Instead, it would be preferable to provide the algorithm with a more intrinsic parameter that only reflects the complexity of the target shape, and have the algorithm calculate the corresponding value of λ if possible. A good choice for such parameter is the degrees of freedom (DOF) of the projection arising from this fitting. When solving the fitting problem of Eq. (5.28), we are effectively projecting the data points $\mathbf{x}_{i,j}$ to points $\mathbf{Y}_{i,j}$ in a restricted subspace of reachable B-splines. The degrees of freedom of this projection operation can be shown to be

$$DOF == tr(\mathbf{S}_{\lambda,\mathbf{Y}}) = tr(\mathbf{B}(\mathbf{B}^T\mathbf{B}+\lambda\mathbf{C})^{-1}\mathbf{B}^T) \tag{5.29}$$

where $\mathbf{B}$ is data dependent matrix, and $\mathbf{C}$ is a constant matrix. The derivation of Eq. (5.29) is based on the theory of smoothing splines and is explained in Appendix A. [NOTE: ADD APPENDIX A] *

Once the user specifies a target DOF, the algorithm can numerically calculate λ for each slice using Eq. (5.29), since DOF is monotonic with λ. The question of target DOF specification remains for the user. As described previously, this question is a bias variance tradeoff. Too small a target DOF constrains the shape too much and underfitting may occur. On the other hand, too large a value for target DOF could result in overfitting. To give the user a rough idea of a good range for target DOF, the algorithm fits a cubic B-spline with varying number of control points on the first hand labeled slice. The target DOF would be twice this number (each control point is in $\mathbb{R}^2$). It then calculates mean fitting error and the variance of the signed curvature plot for the B-spline ([Far02]). The signed curvature of a parametric curve in $\mathbb{R}^2$ is defined by:

$$\kappa(t) = \frac{x''(t)y'(t) - y''(t)x'(t)}{(x'(t)^2 + y'(t)^2)^{3/2}} \tag{5.30}$$

As more and more control points are allowed, the mean fitting error decreases. At the same time, the signed curvature plot tends to have more variance due to overfitting. The resulting plot for an example initial curve is shown in Fig. 5.8, and shows a clear range of number of target DOF for which neither underfitting nor overfitting occur. The user can then experiment with several target DOF values from this range.

5.5.3 Nonlocal Learning of the Underlying Boundary Pattern with the Texture Template

The unoriented boundary texture template $\mathbf{T}_{i,0}$ is a necessary parameter for inferring the CRF, and is learned incrementally as follows. The first shape in slice 1 ($\mathbf{s}_1$) is fit to the hand labeled points in that slice. Then $\mathbf{T}_{1,0}$ is just the average of the $q \times r$ patches centered around the hand labelled points after being rotated to a common orientation.

To obtain subsequent boundary texture templates, we first sample uniformly at random a fixed number m (usually 60) points $x^s_{i,k}, k \in 1 \dots m$ along the currently inferred shape $\mathbf{e}_i$ in image $\mathbf{I}_i$, and take an oriented $q \times r$ patch $\mathbf{a}_{i,k}$ around each sampled point. For convenience, we use $\mathbf{a}_i$ to refer to the set of all $\mathbf{a}_{i,k}$ in image $\mathbf{I}_i$, and $\mathbf{x}^s_i$ to refer to the set of all sampled points $x^s_{i,k}$ in $\mathbf{I}_i$. The orientation of each patch $\mathbf{a}_{i,k}$ is $\theta^s_{i,k}$, the tangent to $\mathbf{s}_i$ at point $x^s_{i,k}$. We define the current slice average texture $\mathbf{t}_{i,0} \in \mathbb{R}^{q \times r}$ as the average of all of the m patches which are sufficiently similar to the latest estimate of the texture template $\mathbf{T}_{i-1,0}$ (judged by normalized cross correlation) in the current image after rotating them to a common orientation (as shown in Fig. 5.2). The similarity condition is to prevent patches from regions affected by the missing wedge from adding more noise than signal to

what
sets
threshold

the current average. If $\mathbf{t}_{i,0}$ is sufficiently similar to $\mathbf{T}_{i-1,0}$ (judged by the normalized cross correlation of the two), the unoriented texture template $\mathbf{T}_{i,0}$ is updated to be a weighted average of the previous template $\mathbf{T}_{i-1,0}$ with the current slice average texture $\mathbf{t}_{i,0}$ according to (5.31) repeated below:

$$\begin{aligned} &\mathbf{T}_{i,0} = \alpha * \mathbf{t}_{i,0} + (1-\alpha) * \mathbf{T}_{i-1,0}, i \in 2 \dots N \\ &\mathbf{T}_{1,0} = \mathbf{t}_{1,0} \end{aligned} \tag{5.31}$$

Otherwise, we just carry forward the previous template, and assign $\mathbf{T}_{i,0} = \mathbf{T}_{i-1,0}$. This operation is depicted graphically in Fig. 5.3.

Here α is an input parameter to the algorithm. This parameter is the relative weight of the texture inferred at the current slice compared to the texture inferred so far from all the slices, and determines the memory of the estimate. The effect of $\mathbf{t}_{i,0}$, the current average texture i, decays exponentially with distance, at a rate of $(1-\alpha)^{(n-i)}$. A small value means a long memory, and less effect of the current slice on the estimate of texture. A value of one indicates no memory, and bases the texture estimate on the most recent slice only. In the interest of robustness, we preferred to have an estimate with a long memory, and chose a relatively small default value of $\alpha = 0.01$.

The above method dynamically adapts the template to changes in boundary texture, but it does assume that the nominal texture is constant in a given slice and slowly varying across slices. In some cases this may not hold, so we also speculatively maintain alternative templates with virtually no history, and very little nonlocal learning.

These alternative templates are clustered subaverage patches using k-means clustering, where k=2. The clustering is done while averaging the patches in only the last slice to get $\mathbf{t}_{i-1,0}$. The clustered subaverages are called $\mathbf{t}^1_{i-1,0}$ and $\mathbf{t}^2_{i-1,0}$, such that $\mathbf{t}_{i-1,0}$ is a weighted average of the two. When the template $\mathbf{T}_{i-1,0}$ does not return any points above a minimum normalized cross correlation threshold (ncc_{min}, a parameter input to the algorithm), it is assumed that the current patch being studied is sufficiently different than the normal template, and instead use $\mathbf{t}^1_{i-1,0}$ and

Table 5.1: Parameters for BLASTED

Parameter	Description	Range	Default
σ_x	Eq. (5.20)	100 pixels	100 pixels
α	Eq. (5.31)	0.01,0.1	0.01
k_a	Eq. (5.26)	0.008-0.04 pixels^{-1}	0.008 pixels^{-1}
q, r	template size	21x71 pixels	21x71 pixels
u, v	candidate region size	101x101 pixels	101x101 pixels
ncc_{min}	NCC threshold	0.6	0.6
max_iter	coordinate ascent	5	5
max_outliers	coordinate ascent	0.5% of points	0.5% of points
max_outlier_err	coordinate ascent	10 pixels	10 pixels

$\mathbf{t}^2_{i-1,0}$, keeping the result which has the highest score.

The estimated templates $\mathbf{T}_{i,0}$, $\mathbf{t}^1_{i,0}$, or $\mathbf{t}^2_{i,0}$ can now be used together with rough estimates for shape $\hat{\mathbf{e}}_{i+1}$ to find candidate points in the next image $\mathbf{I}_{i+1}$ and to classify whether they are on the boundary or not.

Summary of Key Parameters

The key parameters for this inference are summarized in table 5.1. For all our experiments, the slices were taken to be separated by 5 pixels. All parameters were held constant, except for k_a (the shape stiffness parameter), which was slightly varied for 3 out of 20 tried datasets, and α (the texture template adaptation rate parameter) which was changed for one dataset whose boundary texture varied relatively rapidly within the dataset.

5.6 Results

The 3D cryo-ET data used for this work consists of a selected subset of 20 intact Caulobacter crescentus data sets obtained in the course of ongoing research on cell division mutants, PopZ localization [BCZ+08] and polar organization in bacteria. We have included bacteria spanning a range of cell shapes and sizes: C.c. daughter cells (swarmer) are quite small while pre-divisional cells are twice as long and a bit thicker in diameter; dividing cells and mutants with cell division defects can be quite long, extending well beyond a field of view at 25Kx on a 2Kx2K CCD (2.5 μm by side). As a consequence there is a range of thickness in the amorphous ice meniscus embedding these cells, especially at high tilt angles during the acquisition of tilt series. We have thus included for this work data sets acquired at 1 deg angular steps, for smaller and more transparent cells, and data sets acquired at 2 deg angular steps for the larger ones. We aimed as much as possible to take the tilt axis to be close to and parallel to the length of the cell. Cryo-grids are prepared by cryo-plunging aliquots of cell cultures and stored in liquid nitrogen. All data sets were acquired on a JEOL3100 electron microscope equipped with a FEG electron source operating at 300 kV, an Omega energy filter, a Gatan 795 2Kx2K CCD camera, and cryo-transfer stage. Tilt series were recorded with the program Serial-EM [Mas05] adapted to Jeol microscopes using a magnification of 25Kx at the CCD, giving a pixel size of 1.2 nm at the specimen. Underfocus values ranged between 9 $\mu m\pm$ 0.5 um to 14 $\mu m\pm$ 0.5 um, and energy filter widths were typically around 22 eV ± 2 eV. Angular steps of 1 deg and 2 deg were used with angular ranges between 62 deg to 65 deg. Data acquisition and 3D reconstruction details are exactly as published previously in [CBD+08].

5.6.1 Segmented Datasets

A segmentation was considered successful if it localized the membrane within an accuracy better than its thickness (10 nm, or 8.3 pixels) over large portions (at least 70%) of the segmentable slices. A slice was considered segmentable if the entire cross section of the cell was in the field of view, and the cell was not obstructed. This means that the segmentable portion of the cell should be well within the volume box window. By this measure, BLASTED succeeded in segmenting for 19 of the 20 attempted datasets. The failing dataset had sufficiently lower signal to noise ratio than the others such that tracking could not recover enough candidate points to go through more than 20% of the slices. None of the data was prefiltered. Run times were from two to three hours using unoptimized prototype Matlab code on a workstation with two 2.33 GHz Intel Xeon CPU's using less than 1 GB of memory

We show visualizations of the inferred surfaces for four datasets (Caulo6, Caulo13, Caulo8, and Caulo21) in Figs. 5.9–5.12. All of these panels show the surface formed by the inferred outlines. These datasets represent mutant strains of a Caulobacter during division, and offer interesting shapes for segmentation. In each case, a lateral and cross sectional slice are shown with the inferred surface. For each example, we show a few slices from the volume with their inferred boundary points and outline. The points and outlines appear on the outer membrane as desired. A good example of BLASTED's precision and selectivity is seen in Caulo6 (Fig. 5.9). Here, the carbon grid comes up to and along the membrane at the neck of the division (Fig. 5.9(a)), but does not distract the segmentation despite being a much stronger and more visible edge than the membrane. The wandering shape of the membrane in this area is precisely tracked. Another example of BLASTED's robustness is seen in the case of Caulo13, where a carbon grid severely obstructs the cell membrane in some slices (Fig. 5.10(d)), but does not prevent the segmentation from continuing successfully past the carbon grid to the unobstructed regions (Fig. 5.10(e)). This is possible thanks to global shape context and shape based local features. This situation also occurs in Caulo8 although it is not as severe (Fig. 5.11(e)). A contour based method that uses intensity and edges to score contours would incorrectly respond to the carbon grid's edges and infer the wrong shape for the cross section, and thus fail shortly thereafter in these cases. The zoomed in patches in these figures illustrate the challenges in detecting local features. The low contrast and signal to noise and artifacts make it very difficult to rely on intensity and intensity changes alone to make inferences about being a potential membrane point.

We show a visualization for the inferred surface of the dataset Caulo21, which cannot be modeled by an elliptical cross section. We therefore used a cubic B-spline cross sectional shape model, and chose to segment in the lateral direction (orthogonal to the direction used for the previous three datasets). This strategy was possible because the B-spline can model more general shapes. As a result, we segmented two separate surfaces, one being more complex than the other. For the less complex (top) surface we used a target DOF of 8, and for the more complex (bottom with bump) surface we used a target DOF of 14. The choices for these values were made after viewing a curve like that in Fig. (5.8 and choosing one value with little experimentation.

Even with specialized membrane features and higher level shape information, the quality of points along the membrane is not always consistent. To quantitate the accuracy of the contours produced by BLASTED, we hand label every 40th slice in all datasets, and fit an ellipse to each of these to get the ground truth. We then measure the average distance from each point on the ground truth ellipse to the inferred ellipse in the region that points were found. This measure is based on the Chamfer distance transform [Bor88] and is plotted for the dataset Caulo3 in Fig. 5.13. The mean and standard deviation of this measure are listed for 19 datasets in Table 5.2. We do a similar comparison between two attempts by different users at hand labeling the same dataset (Caulo3) to illustrate the intrinsic uncertainty of the ground truth itself in segmentation of the surface. The error incurred by BLASTED is comparable to this uncertainty, indicating that in the range that is being tracked, the achieved accuracy of the inferred shape is as good as hand labeling.

As a second metric of performance, we define the tracking range for a dataset as the ratio of the number of slices for which the distance between ground truth and inferred ellipses is less than 11 nm to the number of complete slices spanning the cell. When using a fixed template from the first slice ($\alpha = 0$), the average range achieved was 89%. Using a value of $\alpha = 0.01$ improved this figure to 93%. Common reasons for failure were weak physical features in whole quadrants of the slice, resulting in distorted ellipses, as well as deviation of the cross sectional shape from the assumed shape model (ellipse or pair of quadratic segments). All experiments used multiple alternative templates as described in Section 5.5.3, which was responsible for improvements of up to 10% tracking range in individual datasets (depending on intra-slice boundary texture variability). The overall results are shown in Table 5.2.

Table 5.2: Tracking range, accuracy for 19 datasets (all parameters default unless otherwise specified)

Dataset	No Slices	Initial Slice	%tracked	mean/std err(nm)	parameters
Caulo1	1040	700	100	3.89/1.78	
Caulo2	1700	500	91.7	6.36/2.53	$k_a = 0.016$
Caulo3	1860	1000	93.6	2.54/1.32	
Caulo4	1545	600	100	4.15/1.78	
Caulo5	700	150	91.43	3.82/1.77	
Caulo6	1800	300	90	2.62/1.06	$k_a = 0.04$
Caulo7	750	250	100	3.42/1.84	
Caulo8	1700	500	97.9	3.30/1.58	
Caulo9	1365	500	100	4.30/2.26	
Caulo10	1700	1000	90.6	3.66/1.49	
Caulo11	1150	500	94.9	5.21/2.41	$k_a = 0.04$
Caulo12	1600	800	97.5	3.11/1.16	
Caulo13	1800	850	92.2	6.06/1.49	
Caulo14	920	300	73.4	3.47/2.22	
Caulo15	1400	800	94.3	5.52/2.04	
Caulo16	1200	350	90.8	3.13/3.04	
Caulo17	210	150	100	7.06/1.85	
Caulo18	650	400	100	4.89/1.2	$\alpha = 0.1$
Caulo19	700	500	85.7	7.92/4.28	
Average	1255		93.9	4.44/1.95	

5.6.2 Comparison with Other Methods

Despite the increasing importance of cryo-ET, to the best of our knowledge, there have been no standard benchmark cryo-ET datasets with ground truth segmentation made publicly available for evaluation and comparison of different algorithms. In absence of such a baseline, we compare performance of BLASTED on one slice of the dataset Caulo1 against several popular boundary recovery techniques: active contours (snakes) and level sets, as well as a state of the art segmentation method developed for electron tomography.

We attempt the segmentation of the outer membrane on the image shown in

Fig. 5.14(a). We start with two popular boundary based techniques which utilize a rough shape initialization: snakes and level sets. For both methods, we use the same initial contour shown in Fig. 5.14(b), which is close to the correct answer. The result for snakes is shown in Fig. 5.14(b), and is obtained with the implementation of [BS95]. Due to intensity gradients being weak features for indicating the membrane, as the snake evolves, large portions of the contour get driven far from the boundary. Level sets results are shown in Fig. 5.14(c,d). We used the method of [CV01], which seeks a contour that separates regions of maximally different intensity. The parameter μ determines length (stiffness) of the recovered contour. We used the implementation of [Der09], and two values of $\mu = 0.07, 0.5$ to illustrate intermediate and high stiffness. When $\mu = 0.07$, the contour evolves to a good boundary where the physical boundary is clear, and to an unrealistic shape elsewhere. To enforce a smooth shape, we increase the stiffness. But this constrains the contour evolution and makes it less responsive to the physical features. While the recovered contour is close to the shape of the correct solution in this case, a closer look reveals errors near the correct boundary (shown in inset). Since recovering the volume involves traversing hundreds of slices, precise localization of the correct boundary in each slice is crucial. Despite the use of global shape context by these methods, the local physical features are not strong enough to drive towards a precision segmentation.

We next used a publicly available state of the art segmentation method developed specifically for electron tomography — the 3D watershed technique of [Vol02], which has been also released recently in a visualization and segmentation toolbox for electron tomography [PMF08]. We show the direct result of applying this method to a volume 100 slices thick (120 nm) containing the slice of Caulo1 in Fig. **??**(e). We take the highest scoring points of the watershed output (8% of all nonzero scoring points), and plot them overlaid with the image in Fig. 5.14(f). The watershed method is considered to be less prone to local optima than level set methods, and this intrinsic 3D version is expected to result in fewer false positives than a 2D version. Although good points on the target membrane were detected, a large number of sizable local maxima in the intensity map that are not related to the cell membrane appear as false positives. Despite the more powerful 3D local features really?

provided by this method, the limitations of using intensity and intensity peaks as the sole physical attribute of a boundary point for membrane detection remained apparent. Furthermore, since this method is based on morphology and has no closed form energy function, it is difficult to incorporate higher level information such as shape in a structured way to improve the inference.

To explore the role of more advanced local features including local texture and orientation, we applied the Berkeley boundary detector in Fig. 5.14(g). We used the implementation of [FMM05] and kept the points whose probability of boundary $P_b > 0.1$. The edges found on the membrane are quite good, but there still are substantial false positives. While the better local features helped, it is clear that higher level shape context is necessary.

As a reasonably simple attempt at using global shape context to clean up the false positives from better local features, we applied a Laplacian exponential decay from an initial shape prior to the scores obtained from watershed and Berkeley boundary detector. This behaves as a hard shape prior. We used the same initial contour from Fig. 5.14(b), and a decay rate of 0.008 pixels^{-1}, the same decay used by BLASTED. We then applied this decay factor to the watershed result from Fig. 5.14(e) to get the new result in Fig. 5.14(h). We then kept the same proportion (top 8%) of the points with new scores, and plotted them overlaid with the image in Fig. 5.14(i). Similarly applying the decay to the Berkeley boundary detector result yields the points in Fig. 5.14(j). This shape prior helped clean up the false positives in both cases, with the best final result being that of the Berkeley boundary detector. However, a careful look at this result reveals still a number of false positives which are not close to the membrane, and this would make it difficult to obtain a precise localization of the membrane, and therefore the tracking through hundreds of slices. This does suggest however that combining higher level shape with richer local features can lead us closer to a precision localization of the membrane.

Finally, the recovered contour using BLASTED is shown in Fig. 5.14(k); it is on the cell outer membrane as desired (shown in inset). In BLASTED, shape context is not used to clean up false positives after local feature detection, but rather as an integral part of the local feature itself. These shape dependent local features which are targeted to membrane detection, as well as their integration with the global shape information in a structured probabilistic model were able to drive the inference to the correct solution. The output of BLASTED from this slice is now ready to be used to estimate the starting contour for the subsequent slice.

5.7 Discussion

The results shown for BLASTED show that by holistically treating the cell boundary in terms of its shape, underlying texture, and local orientation, it is possible to achieve robust boundary detection automatically. BLASTED was tried in over 25 datasets, 19 of which had ground truth for quantitative comparison. While it succeeded in segmenting the cell boundary in the majority of these datasets, there were failures at some point in each dataset, and in some cases very early in the segmentation. The failures were due to insufficient candidate points being detected. This would happen under two general scenarios. In the first one, the boundary was just too faint and weak to be detected, and the cross correlation score was too low. In the second scenario, spacing between the various layers of the membrane had changed such that the underlying boundary texture template was no longer suitable for detecting candidate points. This is common in particular datasets of certain mutants in which membrane spacing was variable. Both these scenarios suggest that a more robust detector for candidate points based on better local features would be beneficial.

The majority of datasets available to us had smooth boundary shapes, such that elliptical or B-spline cross sections with low curvature were sufficient models of the cross sectional shapes. In the cases where B-splines were used, a short experimentation step in the beginning to choose the degrees of freedom of the shape was sufficient to properly constrain the B-spline model. However, in very few cases the shape had very local instances of high curvature (shape deviations) in which case a single global complexity measure such as degrees of freedom was insufficient to capture such detail. It should be noted that the B-spline model we used is arbitrarily restricted to have fixed control point spacing in order to allow control of the overall complexity of the shape by the single number (degrees of freedom). In cases where a few localized details such as high curvature points or corners exist, it would be beneficial to focus an accordingly concentrated modeling effort, such as adding dedicated control points or searching over a wider range of orientations of the boundary texture template.

Really I did not know this

5.7.1 Future Work

While BLASTED has demonstrated a robust segmentation capability for whole cells, it represents a beginning. There is much further work that can be done, both in terms of developing the method as well as using the method to create and expand a knowledge base.

Feature Improvements

As mentioned above, the primary reasons for BLASTED failures were inability to detect candidate points due to physical features that were either too weak or too different from the texture template. The boosted detector used for boundary point classification currently uses 7 features which were designed manually, and were justified by human reasoning. More exploration is needed to identify other features, and preferably a good set of basis functions for boundary features (which could also lead to bases for cryo ET features in general). An example of such a set of features is the set of Gabor features for natural images. Naturally this requires the existence of a fairly large and diverse set of successfully segmented data to be used as training examples in a supervised learning approach.

3D Shape Phenotyping with Electron Tomography

With the ability to segment cells quickly, it is possible to create a larger knowledge base of data from which more shape details can be learned. Such a trend has recently begun in 2D light microscopy images through the automated segmentation and classification work in CellProfiler ([CJL+06]). Using the CellProfiler package, researchers have been able to study 2D shape phenotypes through many experiments involving varying gene expression and automatic segmentation. Extending this concept to 3D electron tomograms could potentially prove fruitful due to the higher resolution of electron microscopy and richer information available from 3D shape.

Not sure what you
mean here

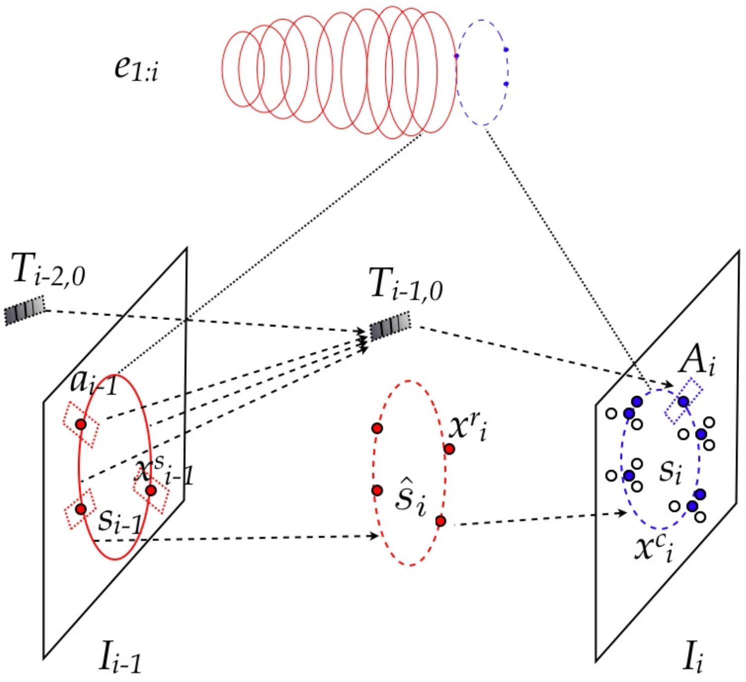


Figure 5.3: Algorithm overview and physical model of the cell: a sequence of outlines and points. Starting with outline $\mathbf{s}_{i-1}$ we predict $\hat{\mathbf{s}}_i$ in image $\mathbf{I}_i$, and sample reference points $\mathbf{x}_i^r$ (red points) along it. For each such point, we seek candidate points, $\mathbf{x}_i^c$ (all blue points, empty as well as solid) inside local regions A_i around $\mathbf{x}_i^r$. The inference then classifies which of these candidate points are on the boundary (solid blue points) and fits the next outline $\mathbf{s}_i$ to them. At each step, the template $\mathbf{T}_{i-1}$ is updated by registering and averaging the patches $\mathbf{a}_{i-1}$, and then weighted averaging with the previous template $\mathbf{T}_{i-1}$. This template is used for the finding and scoring of candidate points in image $\mathbf{I}_i$.

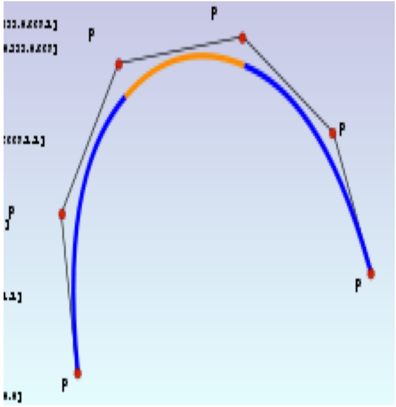


Figure 5.4: An n^{th} order B-spline curve is represented in parametric form as a linear combination of control points $\mathbf{P}_j, j = 1...N$.

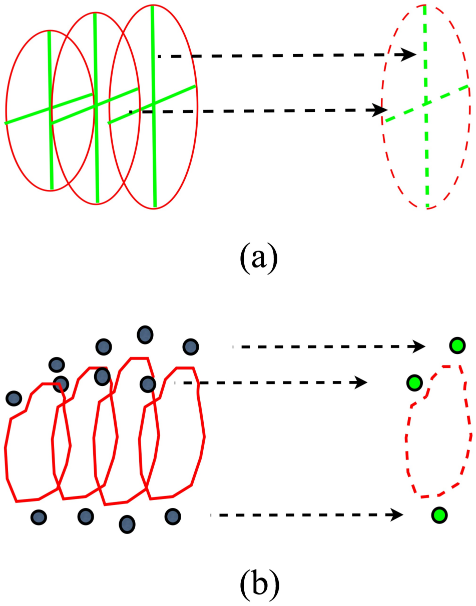


Figure 5.5: A surface is modeled as a sequence of curves, which could be ellipses (a) or B-splines (b).

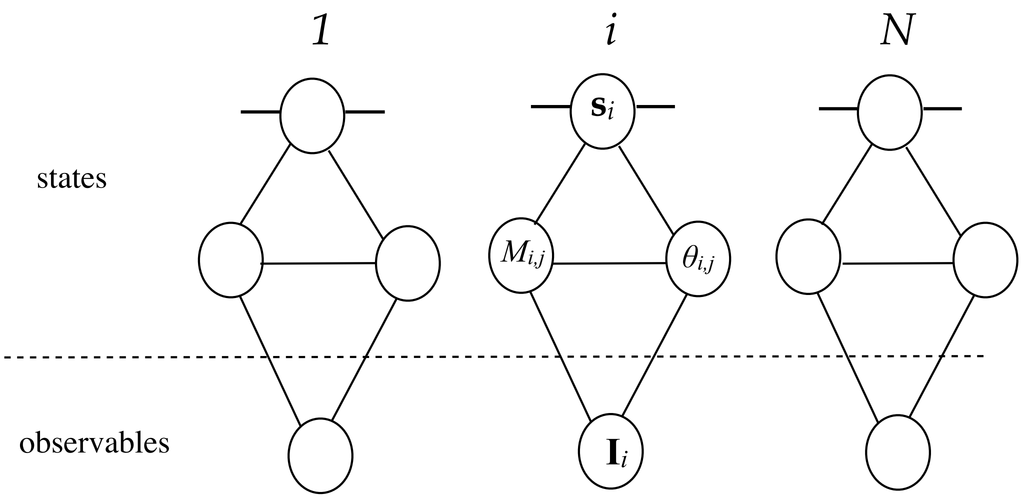


Figure 5.6: Conditional random field (CRF) for BLASTED

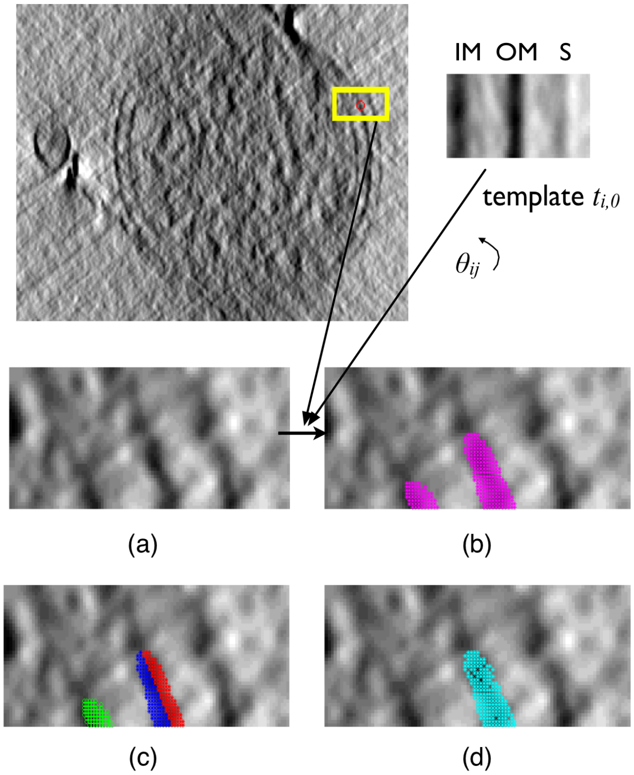


Figure 5.7: Clusters of points found for one reference point $x^r_{i,j}$ using k-means, k=3. (a) patch of I_i, (b) points for which cross correlation is higher than threshold (c) points from (b) put through k-means clustering,k=3 (d) points from (c) that are classified as membrane points by the boosted classifier

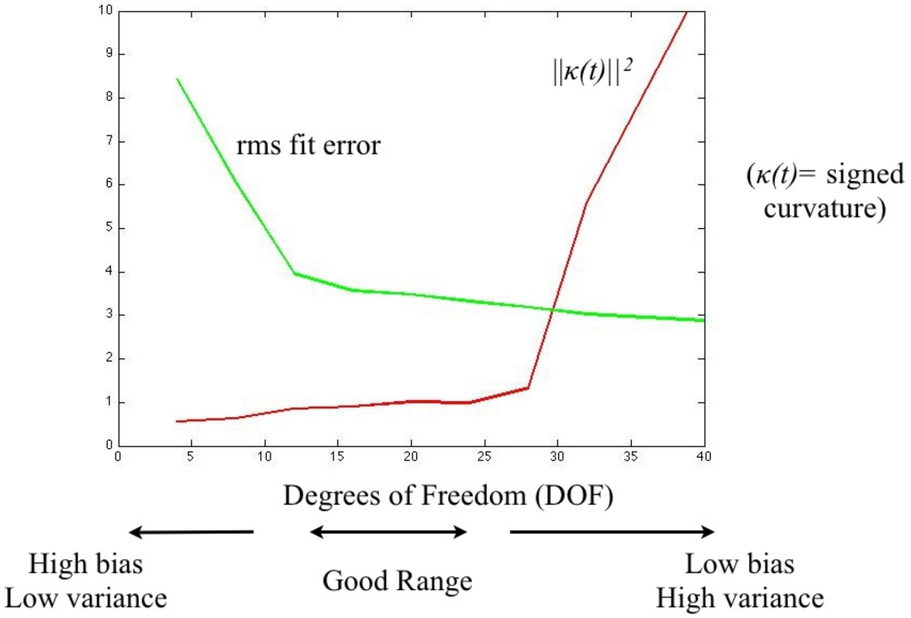


Figure 5.8: Bias variance tradeoff for varying target degrees of freedom (DOF), where target DOF is twice the number of control points used to fit a cubic B-spline to the points provided by the user in the first hand labeled slice.

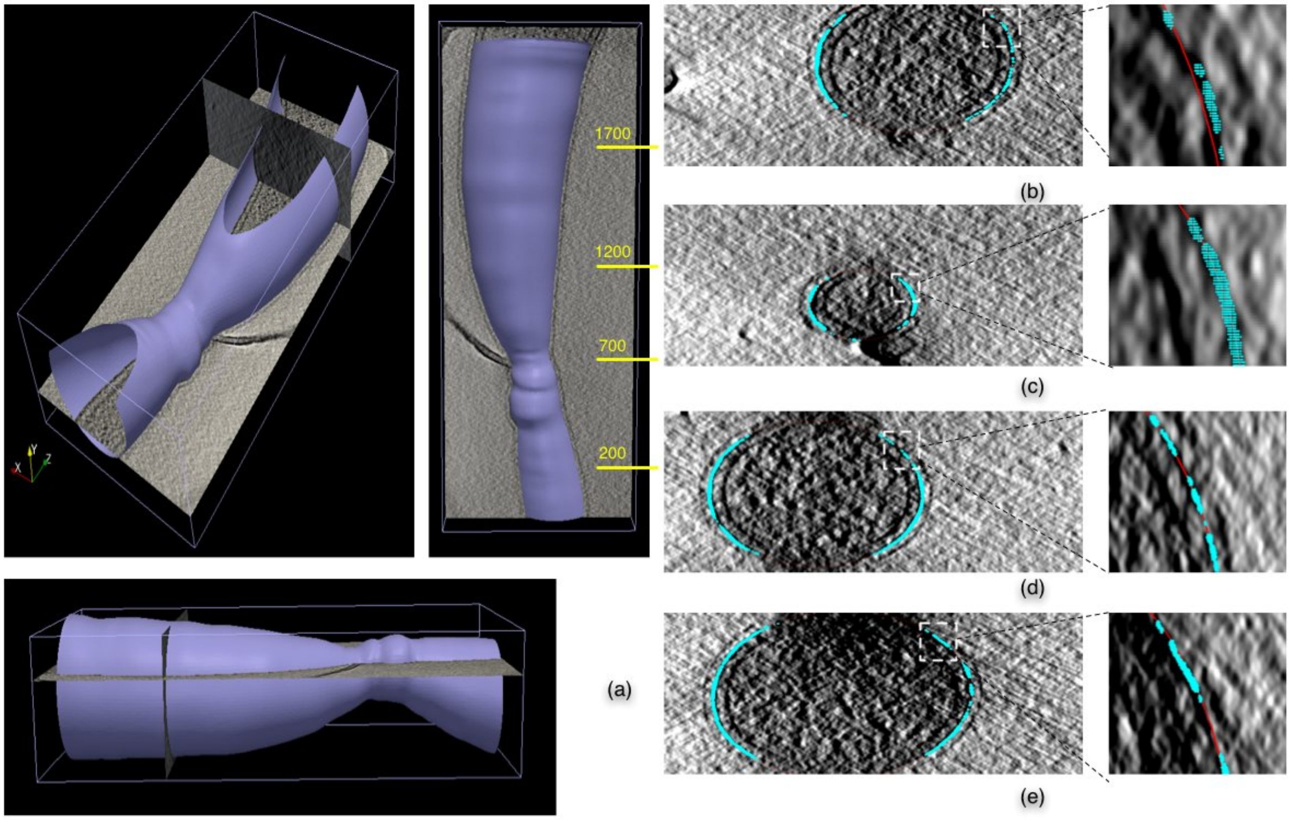


Figure 5.9: Segmentation of Caulo6 — a dividing Caulobacter cell from hand labeled slice 400: a) rendered surface from 3 views, partly cut away for visualization; b) slice 200; c) slice 700 (note carbon grid obstructing bottom part); d) slice 1200; e) slice 1700.

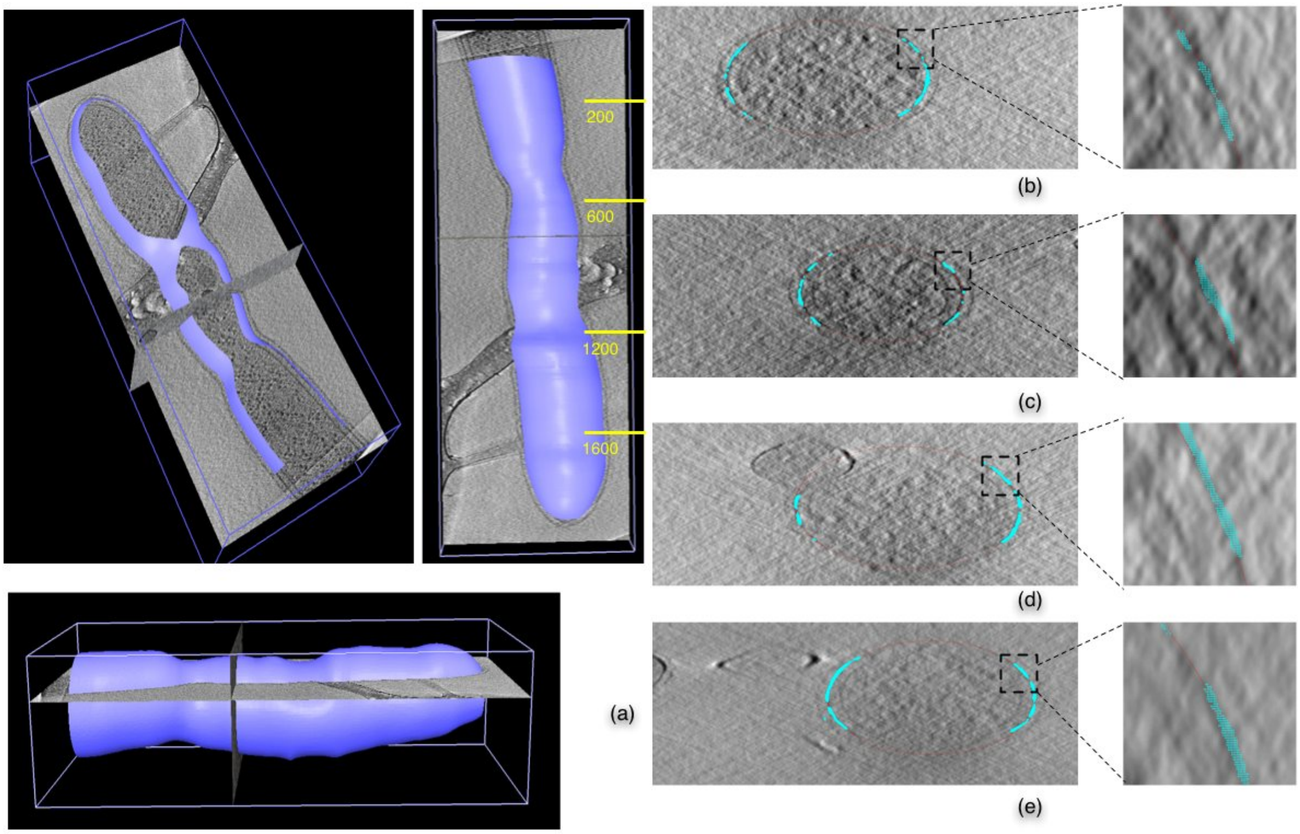


Figure 5.10: Segmentation of Caulo13 — a dividing Caulobacter cell from hand labeled slice 1000: a) rendered surface from 3 views, partly cut away for visualization; b) slice 200; c) slice 600; d) slice 1200 (note carbon grid obstructing top left part); e) slice 1600.

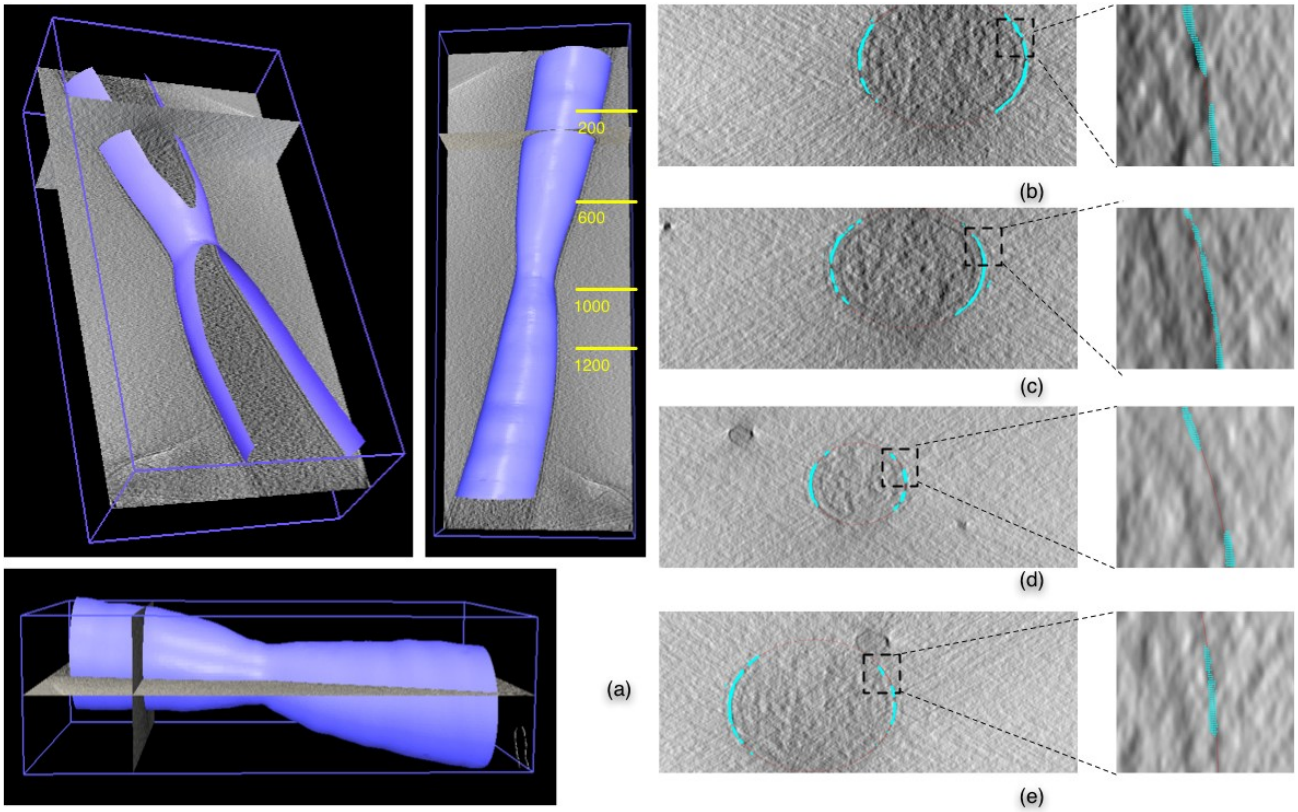


Figure 5.11: Segmentation of Caulo8 — a dividing Caulobacter cell from hand labeled slice 1000: a) rendered surface from 3 views, partly cut away for visualization; b) slice 200; c); slice 600; d) slice 1000; e) slice 1200.

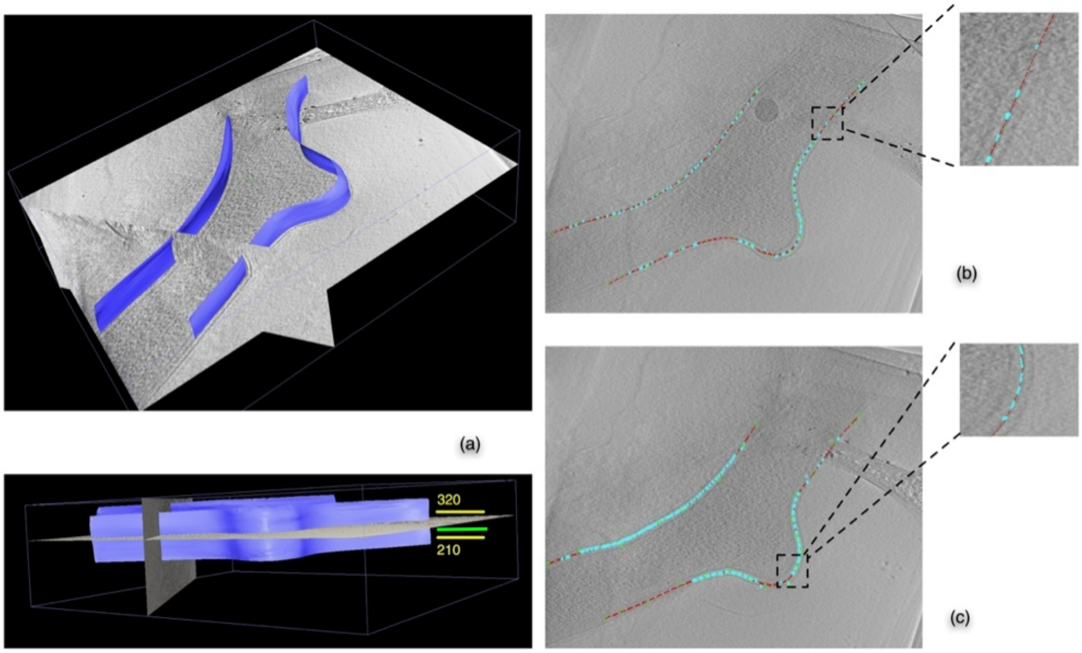


Figure 5.12: Segmentation of Caulo21 — a dividing Caulobacter cell from hand labeled slice 240: a) rendered surface from 2 views; b) slice 320; c) slice 210.

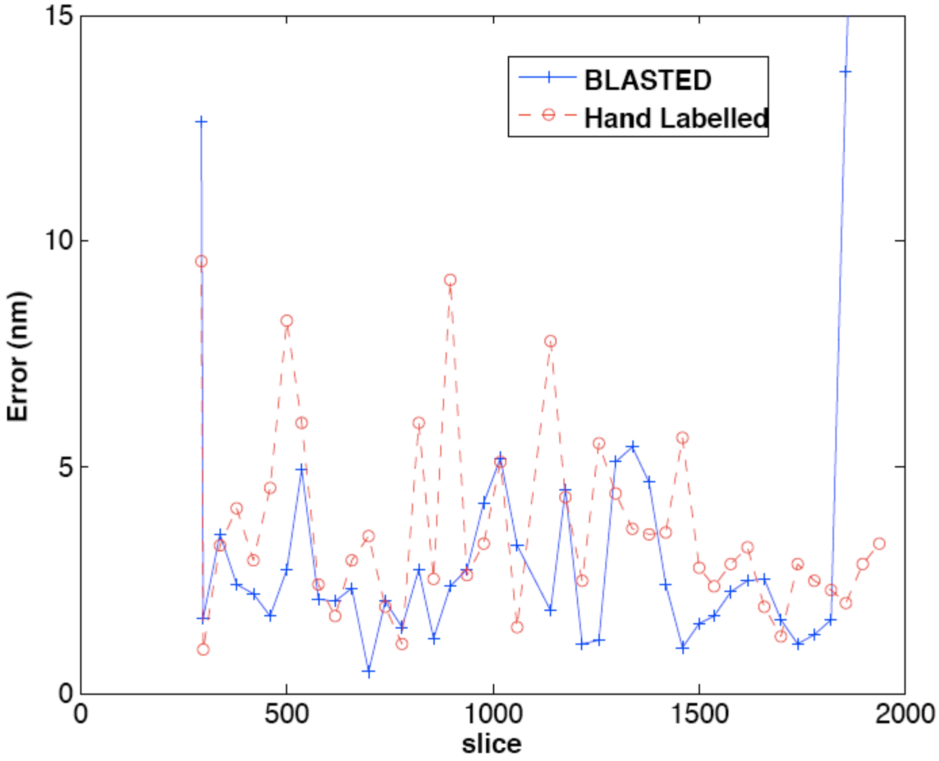


Figure 5.13: Chamfer measure between ground truth and inferred ellipses from BLASTED, as well as for two attempts by different users at ground truth for the dataset Caulo3. Tracking with BLASTED is lost below slice 300 and above slice 1880, when the distance grows very large.

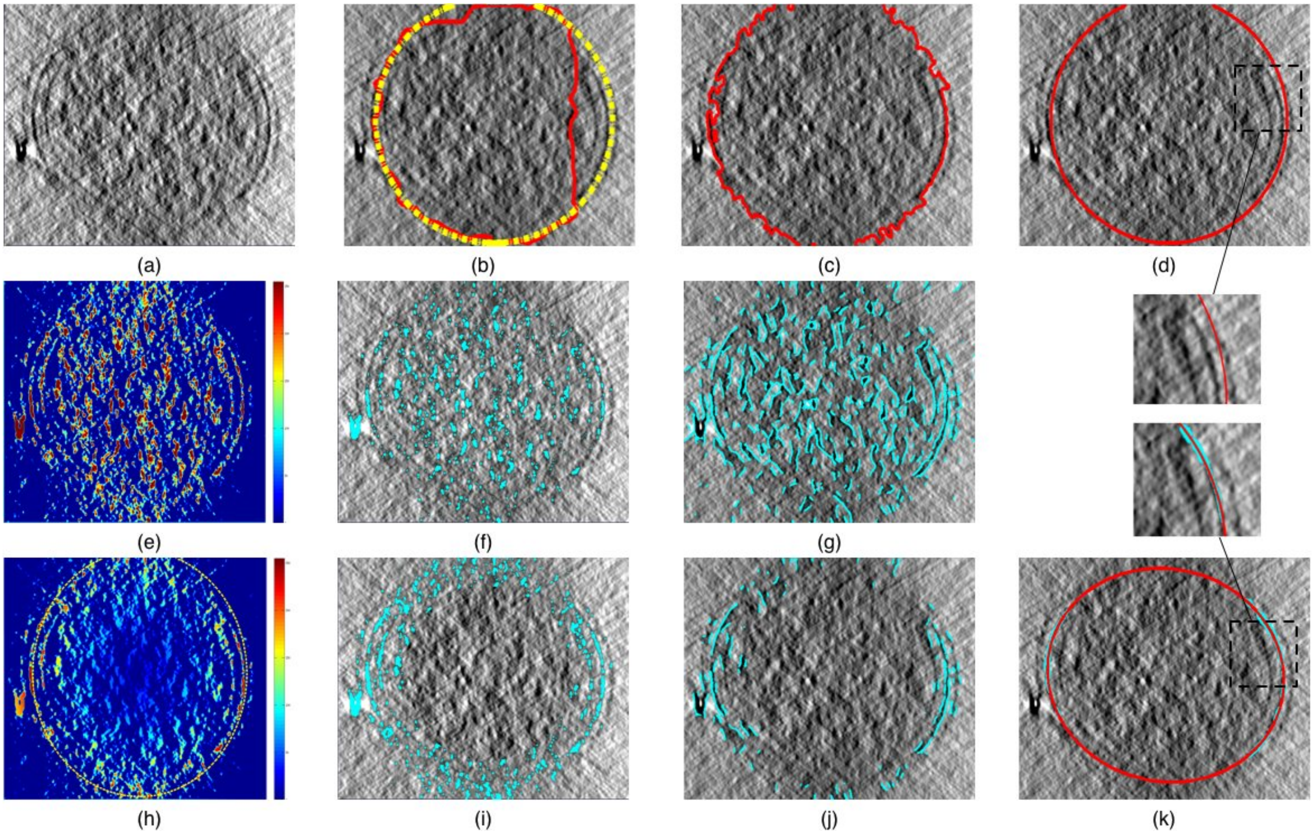


Figure 5.14: Results of different segmentation algorithms on slice of Caulo1: (a) original slice of Caulo1 to be segmented; (b) snakes (initial contour is yellow, final contour is red), (c) Chan-Vese level sets with $\mu = 0.07$; (d) Chan-Vese level sets with $\mu = 0.5$; (e) 3D Watershed segmentation (output map); (f) the top 8% scoring points from 3D Watershed overlaid with Caulo1 slice; (g) Berkeley boundary detector: points whose $P_b > 0.1$; (h) 3D Watershed segmentation with Laplacian decay of rate 0.008 $pixels^{-1}$ applied (output map); (i) the top 8% scoring points from decayed 3D Watershed overlaid with Caulo1 slice; (j) Berkeley boundary detector after Laplacian decay applied: points whose $P_b >$ 0.1; (k) BLASTED (same initial contour as (b,h,i,j), same decay rate as (h,i,j)).

Chapter 6

Conclusions

6.1 Conclusions

Cryogenic electron tomography has emerged as an important modality for imaging of cells. In order for it to fully realize its promise of high resolution 3D imaging of cells and subcellular structures, its pipeline needs to achieve high throughput. The bottlenecks have been a number of labeling tasks in images which contain unreliable local features and require substantial manual intervention. To overcome this uncertainty, strengthen the local features, and ultimately make better inferences for labeling, we used geometric contextual information- the mutual spatial relationships between the local features (This concept is inspired by recent works in computer vision to achieve higher level scene understanding and object detection.) We encoded these relationships in probabilistic frameworks using graphical models, which have the added advantage of available efficient approximate inference algorithms. We have shown that using such frameworks it is possible to automate two time consuming tasks in the ET pipeline- 2D image stack precision alignment and 3D cell boundary segmentation.

Bibliography

[ACM$^+$10] F. Amat, L.R. Comolli, F. Moussavi, K.H. Downing, and M. Horowitz. Subtomogram alignment by adaptive fourier coefficient thresholding. *J. Struct. Biol.*, Accepted, 2010.

[AM79] Brian D.O. Anderson and John B. Moore. *Optimal Filtering.* Prentice-Hall, Inc., 1979.

[AMC$^+$07] Fernando Amat, Farshid Moussavi, Luis R Comolli, Gal Elidan, Kenneth H Downing, and Mark Horowitz. Markov random field based automatic image alignment for electron tomography. *J. Struct. Biol.*, 161(3):260–275, July 2007.

[AMG02] M. Sanjeev Arulampalam, Simon Maskell, and Neil Gordon. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

[Bau02] W. Baumeister. Electron tomography: towards visualizing the molecular organization of the cytoplasm. *Curr. Opin. Struct. Biol.*, 12:679–684, 2002.

[BCZ$^+$08] Grant R. Bowman, Luis R. Comolli, Jian Zhu, Michael Eckart, Marcelle Koenig, Kenneth H. Downing, W.E. Moerner, Thomas Earnest, and Lucy Shapiro. A polymeric protein anchors the chromosomal origin/parb complex at a bacterial cell pole. *Cell*, 134(6):945 – 955, 2008.

[BHE01] Sami Brandt, Jukka Heikkonen, and Peter Engelhardt. Multiphase method for automatic alignment of transmission electron microscope images using markers. *Journal of Structural Biology*, 133(1):10 – 22, 2001.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[Bor88] G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(6):849–865, 1988.

[BPS$^+$06] Eric Betzig, George H. Patterson, Rachid Sougrat, O. Wolf Lindwasser, Scott Olenych, Juan S. Bonifacino, Michael W. Davidson, Jennifer Lippincott-Schwartz, and Harald F. Hess. Imaging Intracellular Fluorescent Proteins at Nanometer Resolution. *Science*, 313(5793):1642–1645, 2006.

[Bra95] Ronald N. Bracewell. *Two-dimensional imaging*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995.

[BS95] C. Bregler and M. Slaney. Snakes technical report, 1995.

[BSL$^+$08] A. Bartesaghi, P. Sprechmann, J. Liu, G. Randall, G. Sapiro, and S. Subramaniam. Classification and 3d averaging with missing wedge correction in biological electron tomography. *Journal of Structural Biology*, 162(3):436 – 450, 2008.

[BSS05] Alberto Bartesaghi, Guillermo Sapiro, and Sriram Subramaniam. An energy-based three-dimensional segmentation approach for the quantitative interpretation of electron tomograms. *IEEE Transactions on Image Processing*, 14(9):1314–1323, 2005.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.

[BZ06] Sami S. Brandt and Ulrike Ziese. Robust alignment of transmission electron microscope tilt series. In *ICPR (4)*, pages 683–686, 2006.

[Can86] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, 1986.

[CBD$^+$08] Luis R Comolli, Brett J Baker, Kenneth H Downing, Cristina E Siegerist, and Jillian F Banfield. Three-dimensional analysis of the structure and ecology of a novel, ultra-small archaeon. *ISME J*, 3(2):159–167, 2008.

[CD05] Luis R. Comolli and Kenneth H. Downing. Dose tolerance at helium and nitrogen temperatures for whole cell electron tomography. *Journal of Structural Biology*, 152(3):149 – 156, 2005.

[CGS05] Giovanni Cardone, Kay Grünewald, and Alasdair C. Steven. A resolution criterion for electron tomography based on cross-validation. *Journal of Structural Biology*, 151(2):117 – 129, 2005.

[CJ05] Luis R. Comolli and Ellen M. Judd. Distinct constrictive processes, separated in time and space, divide caulobacter inner and outer membranes. *Journal of Bacteriology*, 187(20):6874–6882, October 2005.

[CJL$^+$06] Anne Carpenter, Thouis Jones, Michael Lamprecht, Colin Clarke, In Kang, Ola Friman, David Guertin, Joo Chang, Robert Lindquist, Jason Moffat, Polina Golland, and David Sabatini. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 7(10):R100+, October 2006.

[CT93] Tim Cootes and C.J. Taylor. Active shape model search using local grey-level models: A quantitative evaluation. In *4th British Machine Vision Conference*, pages 639–648. BMVA Press, 1993.

[CV01] T. F. Chan and L. A. Vese. Active contours without edges. *IEEE Trans. Image Process.*, 10(2):266–277, 2001.

[DDFG01] Arnaud Doucet, Nando De Freitas, and Neil Gordon, editors. *Sequential Monte Carlo methods in practice.* Springer, 2001.

[Der09] Konstantinos G. Derpanis. York university center for vision research software page, February 2009.

[Far02] Gerald Farin. *Curves and surfaces for CAGD: a practical guide.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[FFJS08] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(1):36–51, 2008.

[FH02] Achilleas S. Frangakis and Reiner Hegerl. Segmentation of two- and three-dimensional data from electron microscopy using eigenvector analysis. *Journal of Structural Biology*, 138(1-2):105 – 113, 2002.

[FHT00] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Ann. Stat.*, 28(2):337–374, 2000.

[FMM05] Charless C. Fowlkes, David R. Martin, and Jitendra Malik. The berkeley segmentation engine (bse), May 2005.

[For73] G.D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, March 1973.

[FPF99] Andrew W. Fitzgibbon, Maurizio Pilu, and Robert B. Fisher. Direct least square fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):476–480, 1999.

[Fra06] Joachim Frank. *Electron Tomography.* Springer-Verlag, 2006.

[FSC97] J.J. Fernandez, J.R. Sanjurjo, and J.M. Carazo. A spectral estimation approach to constrast transfer function detection in electron microscopy. *Ultramicroscopy*, 68:267–295(29), 1997.

[GCJ08] Lu Gan, Songye Chen, and Grant J J. Jensen. Molecular organization of gram-negative peptidoglycan. *Proc. Natl. Acad. Sci. Unit. States Am.*, 105(48):18953–18957, November 2008.

[GWBVE08] E. Garduño, M. Wong-Barnum, N. Volkmann, and M. Ellisman. Segmentation of electron tomographic data sets using fuzzy set theory principles. *J. Struct. Biol.*, 162(3):368–379, 2008.

[HB07] J. Bernard Heymann and David M. Belnap. Bsoft: Image processing and molecular modeling for electron microscopy. *Journal of Structural Biology*, 157(1):3 – 18, 2007. Software tools for macromolecular microscopy.

[HEPK09] G. Heitz, G. Elidan, B. Packer, and D. Koller. Shape-based object localization for descriptive classification. *Int. J. Comput. Vis.*, 84(1):40–62, August 2009.

[Her95] G. T. Herman. Image reconstruction from projections. *Real-Time Imaging*, 1(1):3 – 18, 1995.

[Hoc01] Dorit S. Hochbaum. An efficient algorithm for image segmentation, markov random fields and related problems. *J. ACM*, 48(4):686–701, 2001.

[HTF03] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning.* Springer, corrected edition, July 2003.

[Hub81] Peter J. Huber. *Robust statistics / Peter J. Huber.* Wiley, New York :, 1981.

[JBH$^+$09] Won-Ki Jeong, Johanna Beyer, Markus Hadwiger, Amelio Vazquez, Hanspeter Pfister, and Ross T. Whitaker. Scalable and interactive segmentation and visualization of neural processes in em datasets. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1505–1514, 2009.

[Kal60] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[KF09] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, 2009.

[KK03] A.J. Koster and J. Klumperman. Electron microscopy in cell biology: integrating structure and function. *Nat. Rev. Mol. Cell. Biol.*, 4:6–10, 2003.

[KMFaL01] Frank Kschischang, Senior Member, Brendan J. Frey, and Hans andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47:498–519, 2001.

[KMM96] James R. Kremer, David N. Mastronarde, and J. Richard McIntosh. Computer visualization of three-dimensional image data using imod. *Journal of Structural Biology*, 116(1):71 – 76, 1996.

[KS01] A. Kak and M. Slaney. *Principles of computerized tomographic imaging.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.

[KWT88] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *Int. J. Comput. Vis.*, V1(4):321–331, January 1988.

[KZ03] Junhwan Kim and Ramin Zabih. A segmentation algorithm for contrast-enhanced images. *IEEE International Conference on Computer Vision*, 1:502–509, 2003.

[LBPE06] A. Lawrence, J.C. Bouwer, G. Perkins, and M.H. Ellisman. Transform-based backprojection for volume reconstruction of large format electron microscope tilt series. *J. Struct. Biol.*, 154(2):144–67, 2006.

[LKGD07] C. Li, C. Kao, J. Gore, and Z. Ding. Implicit active contours driven by local binary fitting energy. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn.*, pages 1–7, Washington, DC, USA, 2007. IEEE Computer Society.

[LMP01] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[Mas05] David N. Mastronarde. Automated electron microscope tomography using robust prediction of specimen movements. *Journal of Structural Biology*, 152(1):36 – 51, 2005.

[Mey01] Fernand Meyer. An overview of morphological segmentation. *IJPRAI*, 15(7):1089–1118, 2001.

[MFM04] David R. Martin, Charless C. Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):530–549, 2004.

[MHA$^+$09] Farshid Moussavi, Geremy Heitz, Fernando Amat, Luis R. Comolli, Daphne Koller, and Mark Horowitz. 3d segmentation of cell boundaries from whole cell cryogenic electron tomography volumes. *Journal of Structural Biology*, In Press, Corrected Proof:–, 2009.

[MLJ06] Gavin E. Murphy, Jared R. Leadbetter, and Grant J. Jensen. In situ structure of the complete treponema primitia flagellar motor. *Nature*, 442(7106):1062–1064, 2006.

[MS89] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Comm. Pure. Appl. Math.*, 42(5):577–685, 1989.

[Nat01] Frank Natterer. *The Mathematics of Computerized Tomography.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.

[NJ08] H.T. Nguyen and Q.A. Ji. Shape-driven three-dimensional watersnake segmentation of biological membranes in electron tomography. 27(5):616–628, May 2008.

[NWvdB03] Hieu Tat Nguyen, Marcel Worring, and Rein van den Boomgaard. Watersnakes: Energy-driven watershed segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(3):330–342, 2003.

[OFK$^+$06] J. Ortiz, F. Foerster, J. Kuerner, A. Linaroudis, and W. Baumeister. Mapping 70s ribosomes in intact cells by cryoelectron tomography and pattern recognition. *J. Struct. Biol.*, 156(2):334 – 341, 2006.

[OPZ08] Andreas Opelt, Axel Pinz, and Andrew Zisserman. Learning an alphabet of shape and appearance for multi-class object detection. *Int. J. Comput. Vision*, 80(1):16–44, 2008.

[PMF08] Sabine Pruggnaller, Matthias Mayr, and Achilleas S. Frangakis. A visualization and segmentation toolbox for electron microscopy. *Journal of Structural Biology*, 164(1):161 – 165, 2008.

[PT97] Les Piegl and Wayne Tiller. *The NURBS book (2nd ed.).* Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[Rad17] J. Radon. Über die bestimmung von funktionen durch ihre integralwerte längs gewisser mannigfaltigkeiten. *Ber. Saechs. Akad. Wiss.*, 69:262–278, 1917.

[Rad86] Johann Radon. On the determination of functions from their integral values along certain manifolds. *Medical Imaging, IEEE Transactions on*, 5(4):170 –176, dec. 1986.

[RBZ06] Michael J Rust, Mark Bates, and Xiaowei Zhuang. Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (storm). *Nat Meth*, 3(10):793–796, 2006.

[RHMM99] D. Ress, M.L. Harlow, R.M. Marshall, and U.J. McMahan. Computer visualization of three-dimensional image data using imod. *Journal of Electron Microscopy*, 48(3):277–287, 1999.

[San07] Kristian Sandberg. Methods for image segmentation in cellular tomography. In J. Richard McIntosh, editor, *Cellular Electron Microscopy*, volume 79 of *Methods in Cell Biology*, pages 769 – 798. Academic Press, 2007.

[SB07] Kristian Sandberg and Moorea Brega. Segmentation of thin structures in electron micrographs using orientation fields. *J. Struct. Biol.*, 157(2):403 – 415, 2007.

[Set95] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. In *Proc. Nat. Acad. Sci*, pages 1591–1595, 1995.

[SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[SMVC09] Sjors H. W. Scheres, Roberto Melero, Mikel Valle, and Jose-Maria Carazo. Averaging of electron subtomograms and random conical tilt reconstructions through likelihood optimization. 17(12):1563–1572, 2009.

[Sob70] Irwin Edward Sobel. *Camera models and machine perception.* PhD thesis, Stanford University, Stanford, CA, USA, 1970.

[Vol02] Niels Volkmann. A novel three-dimensional variant of the watershed transform for segmentation of electron density maps. *Journal of Structural Biology*, 138(1-2):123 – 129, 2002.

[VS91] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(6):583–598, Jun 1991.

[Wah07] Benjamin Wah. *Wiley Encyclopedia Of Computer Science And Engineering.* John Wiley & Sons, 2007.

[YFW05] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Trans. Inform. Theor.*, 51:2282–2313, 2005.

[ZK04] Ramin Zabih and Vladimir Kolmogorov. Spatially coherent clustering using graph cuts. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn.*, pages 437–444, 2004.

[ZKB+07] Shawn Q. Zheng, Bettina Keszthelyi, Eric Branlund, John M. Lyle, Michael B. Braunfeld, John W. Sedat, and David A. Agard. Ucsf tomography: an integrated software suite for real-time electron microscopic tomographic data collection, alignment, and reconstruction. *J Struct Biol*, 157(1):138–147, Jan 2007.

[ZT98] Djemel Ziou and Salvatore Tabbone. Edge detection techniques - an overview. *Int. J. Pattern Recogn. Image Anal.*, 8:537–559, 1998.