ABSTRACT

A DICOM Viewer Plugin to Aid Diagnosis of Ligamentous Injuries of the Occipital Cervical Complex

Jacob Hoffman

Director: Dr. Brian Garner, PhD

Large magnitude deceleration of the human body, such as that experienced in an automobile accident, has the potential to cause whiplash in the neck. The ligaments in the occipital-cervical complex, or OCC, of the neck may be damaged to the extent that a sudden disturbance of the head may cause paralysis or death. The injury often presents itself as bone pulled out by ligaments, and is thus visible in x-rays and CT scans. However, in the case that a ligament is damaged, the only indication of injury in a CT scan is an enlargement of the interval between the base of the skull and the second cervical vertebra, otherwise known as the BDI. The present study investigated the viability of a medical image viewer plugin to aid in the diagnosis of injuries to the Occipital Cervical Complex, or OCC. Comparing the output of the plugin to measurements of the BDI made by a doctor reveals that the plugin can reliably produce a measurement when told the location of the bones involved, though its measurements are on average over a millimeter different than those of the doctor. However, this study reveals that there is a future in applying computer vision to injury detection in CT scans.

APPROVED BY DIRECTOR OF HONORS THESIS:

____

Dr. Brian Garner, Department of Mechanical Engineering

APPROVED BY THE HONORS PROGRAM:

____

Dr. Andrew Wisely, Director

DATE: ________________________

A DICOM VIEWER PLUGIN TO AID DIAGNOSIS OF LIGAMENTOUS INJURIES OF THE OCCIPITAL CERVICAL COMPLEX

A Thesis Submitted to the Faculty of

Baylor University

In Partial Fulfillment of the Requirements for the

Honors Program

By

Jacob Hoffman

Waco, Texas

May 2012

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

ACKNOWLEDGEMENTS

This thesis would have been impossible to complete without the help of many people along the way. Their guidance, inspiration, and support have helped this project come to fruition. I would like to thank Dr. Garner for introducing me to this project, for his encouragement when I felt like I was overloaded, for his advice and direction, for all that he has taught me about computing and for lending me his OpenGL Redbook for over ten months. I would like to thank Dr. Chaput at Scott and White Hospital who first conceived of this project, without whom none of this thesis would exist, and Zach for taking the time to make BDI measurements and helping to anonymize the patients. My Mom, Dad, Luke and Savannah, not to mention our dogs, who always support me and believe in me. Shaynor Newsome at RUF and Jeff Hatton at Redeemer, who helped lead me to Christ over this past year. My friends at Swing Dance, who always somehow brighten my spirit even when I am completely stressed out. Finally I would like to thank God, for bringing me to this great university and giving me so many good friends and opportunities.

CHAPTER ONE

Introduction

Background

Large magnitude deceleration of the human body, such as that experienced in an automobile accident, has the potential to cause whiplash in the neck. The ligaments in the occipital-cervical complex, or OCC, of the neck may be damaged to the extent that a sudden disturbance of the head may cause paralysis or death. Despite the danger, such injuries may not clearly present upon initial medical examination. Two of these types of injuries are known as Atlanto-occipito Dissociations, or AOD, and Atlanto-axial Dissociations, or AAD (Chaput et al., 2011).

The OCC, AOD and AAD

The OCC is understood to be the region of the cervical spine extending from the occiput to the second cervical interspace (Chaput et al., 2011). Bones involved in the OCC are the base of the skull, the Atlas, otherwise known as C1 of the cervical spine, and the Axis, or C2. The base of the skull has a circular opening through which the brain stem descends into the spinal column. The inferior tip of the skull in the sagittal plane is the basion. The Atlas is a circular vertebra that supports the globe of the skull. The Axis resides beneath the Atlas and possesses a protrusion called the dens.

The dens is the pivot point of the Atlas, and is thus inserted through the Atlas' circular opening and aligns with the basion of the skull. The distance between the basion and the superior tip of the dens is called the Basion-Dens Interval, or BDI.

On the lateral mass of the Atlas are two regions that align with the articular surfaces of the occipital condyle. The space between these surfaces is called the Atlanto-Occipital Interval, or AOI (Bertozzi, Rojas, & Martinez, 2009). A third metric, the Lateral Mass Interval or LMI, is defined as the distance between the top of the arch of the Axis to the bottom of the arch of the Atlas measured on the coronal cut transecting the dens at its midpoint (Chaput et al., 2011).

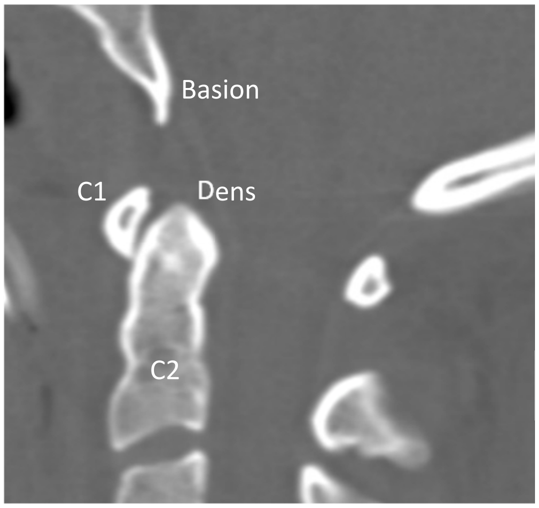
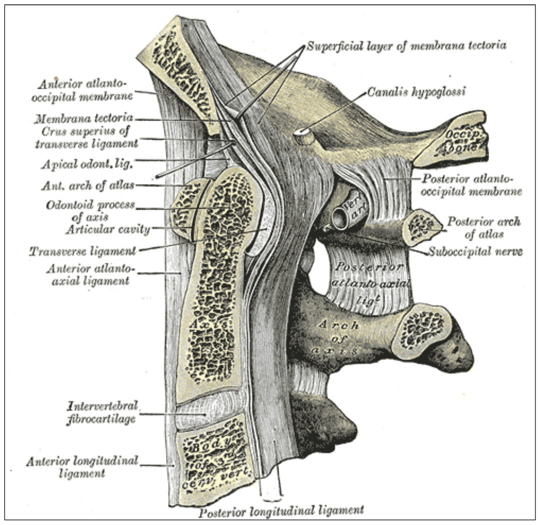


Figure 1.1 A close-up of the OCC. The basion-dens interval is the gap directly between the two features.


Superficial layer of membrana tectoria
Anterior atlanto-occipital membrane
Canalis hypoglossi
Membrana tectoria
Crus superius of transverse ligament
Occip. bone
Apical odont. lig.
Ant. arch of atlas
Posterior atlanto-occipital membrane
Odontoid process of axis
Vert. art.
Articular cavity
Posterior arch of atlas
Transverse ligament
Suboccipital nerve
Anterior atlanto-axial ligament
Posterior atlanto-axial ligt
Arch of axis
Axis
Intervertebral fibrocartilage
Body of 3rd cerv. vert.
Anterior longitudinal ligament
Posterior longitudinal ligament
Figure 1.2 An artist's rendition of the OCC. Note how C1 is a ring that rests on top of C2. (Source: Gray's Anatomy.)

AODs occur when the Atlas and the skull are further away from each other than they should be. AADs occur when the Atlas and the Axis are further away from each other than they should be. Both are due to damage to the ligaments in the neck and cause excessive strain on the brain stem. In this state, these vertebrae are unstable and even a minor jarring, such as stepping off of a sidewalk, can lead to catastrophic failure in the neck and instantly cause quadriparesis or even death.

The BDI and AOI as Indicators of AOD and AAD

Measurements of the BDI, AOI, and LMI have been used to diagnose patients with AOD and AAD. Though the AOI has not been a reliable indicator of OCC injuries, the LMI is useful the BDI has been used to great effect (Chaput, Walgama, Song, Hanson, & Rahm, 2009). BDIs exceeding 10mm or LMIs greater than or equal to 4mm are good indications of an OCC injury, while BDIs > 9.1mm and LMIs > 3.5mm should warrant further analysis with an MRI for viewing ligaments involved (Chaput et al., 2011).

Problem

As opposed to broken bones which are clearly visible in X-rays and CT scans, OCC injuries involve the ligaments of the neck. Since ligaments are soft tissue, they do not appear in X-ray or CT images. However, they are visible in MRIs, and thus diagnosing OCC injuries with MRIs is relatively straightforward. The difficulty arises from the fact that such injuries typically present themselves in victims of automobile accidents, and thus broken bones are of greatest concern and are checked for first with a CT scan.

Upon seeing no broken bones present, a doctor might assume that the patient should be released. An experienced spinal surgeon would be able to spot the telltale signs of an OCC injury in a CT scan with ease, but an emergency room doctor may not. While the measurements of the BDI and the LMI provided by Chaput et al. provide a standard method for recognizing the signs of an OCC injury in a CT scan, it requires an acute knowledge of the specific points from which the measurements are taken. Furthermore,

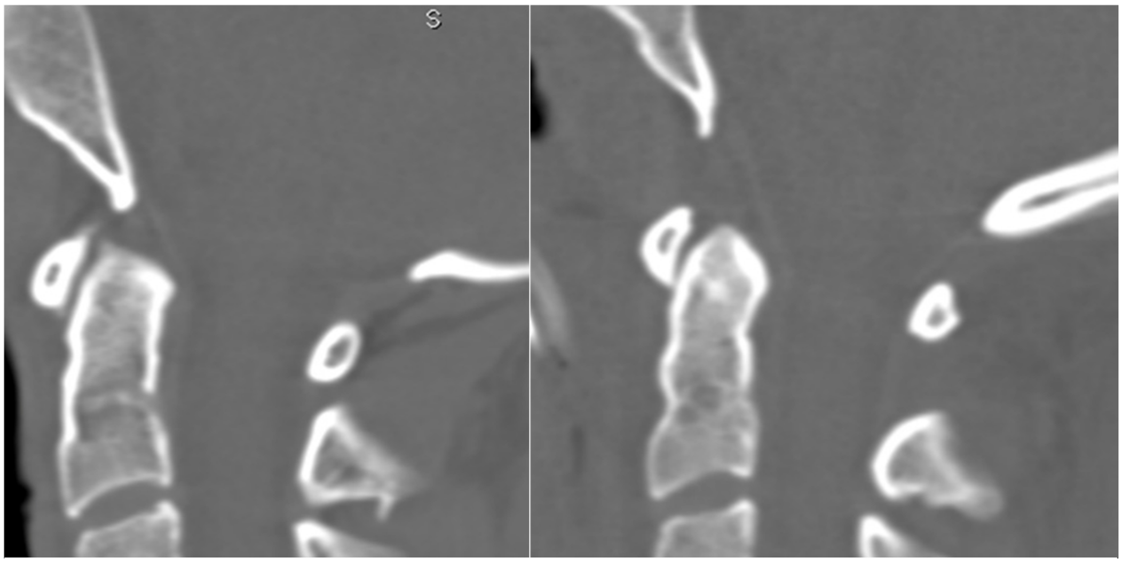


Figure 1.3 A comparison of a healthy patient (left) and an injured patient (right). The difference between life and death in this type of injury is a matter of a few millimeters.

since the difference between healthy and unhealthy is small, errors in measurement selection can undermine the validity of the diagnosis.

On top of this, the values identified by Chaput et al. are from a relatively small sample size, and thus are difficult to generalize to all patients. Dr. Christopher Chaput at Scott & White Hospital in Temple, Texas has expressed desire for an automated method to calculate the intervals that were investigated in his recent study. Since the publication of Dr. Chaput's last research article, he has accumulated CT scans of both healthy and unhealthy OCCs from over eight-hundred patients. Calculating the intervals based off of this larger sample would produce more accurate measurements. However, computing all the data would be an arduous task. An automated method would be able to address this need.

An automated diagnostic tool would also provide doctors with a warning when the CT scan of a patient suffering from OCC injury is loaded. This would eliminate the difficulty of diagnosing OCC injuries with CT scans, saving lives in the process. If the injury can be discovered in time, the patient has a high chance of survival. Very few patients survive untreated OCC injuries (Dominguez, Rahm, & Chaput 2009).

Another challenge with CT scans is that no two image series are alike. Though measures are taken by technicians to provide consistency through every series, a program cannot be guaranteed to receive each series at the same orientation and contrast setting. Often critical aspects of the OCC are omitted simply because technicians are unaware of the need for their inclusion due to the rarity of such injuries.

Since CT scans are manipulated on what is known as a DICOM viewer, an automated method would need to interact with a CT series through DICOM image software. A variety of DICOM viewers are available, but few of them are open source.

Purpose

The purpose of this study is to create a software module that can warn doctors in the event that a patient displays the characteristics of an OCC injury. A software module such as this can be incorporated as a plugin for existing, commercially available software packages, and has the potential to help prevent the paralysis and deaths of hundreds of individuals each year that would otherwise go untreated. It would also allow for the computation of a more general assessment of what constitutes a healthy OCC. The versatility of the plugin would allow for it to be used in emergency rooms around the nation.

Solution

Plan overview

The product of this research will be a program that will aid doctors in the diagnosis of OCC injuries. It will interact with the DICOM viewer OsiriX, an open source application that allows for programs such as this one to connect to it as a 'plugin.' The program will be able to recognize key features of the OCC without confusing it with other parts of the cervical spine and without being thrown off by variations in hue, noise, or orientation of the CT scan. Using Dr. Chaput's method of identifying OCC injuries, the software will automatically measure the BDI of the patient and warn the user if the measurement falls outside a normal range. The BDI was chosen as the indicator because it is the most straight-forward to measure and provides a good prediction of the health of the patient. The software will be composed of several files of code written primarily in C++ to allow for multi-platform portability. The output of the software will be compared to measurements made at Scott and White Hospital.

Thesis overview

This thesis is organized into four chapters, including this Introduction as Chapter I. Chapter II details the methods used to analyze the image series. Chapter III contains the results of this study. Chapter IV discusses the results and makes recommendations for the future of this project.

CHAPTER TWO

Methods

Algorithms Overview

In order for a computer algorithm to measure the distance between bones, it is necessary for it to be able to distinguish the edges of bones. This chapter provides a detailed description of the process by which the software discerns the dens and the basion of the OCC from the rest of a CT-series.

The initial prototype featured a simple approach for distinguishing bone by using a threshold. *Figure 2.3* depicts the result of applying different thresholds to the same image. The prototype used a threshold of ‘200,’ the result of which can be viewed in *Figure 2.10* B. While it served well for a proof of concept in the early stages of the thesis, it was severely limited in its dependability since it made a gross assumption for all CT-series and was tied to OsiriX’s proprietary representation of image data. In other words, bone may not always appear at the same intensity in all series and if the algorithm were to be ported to a new DICOM viewer, it would need to be reprogramed.

It became evident that a more sophisticated algorithm was needed. The current method is independent of any pre-prescribed pixel value for defining bone. *Figure 2.1* outlines the algorithm that the software uses to determine the edges of bone. It draws from the Canny Edge Detector (Canny 1986), in that it considers both edge direction and magnitude. It differs in that it does not use hysteresis in isolating the edges, but instead applies separate thresholding methods to the gradient magnitude image and gradient direction image to isolate regions of the image that contain an edge.

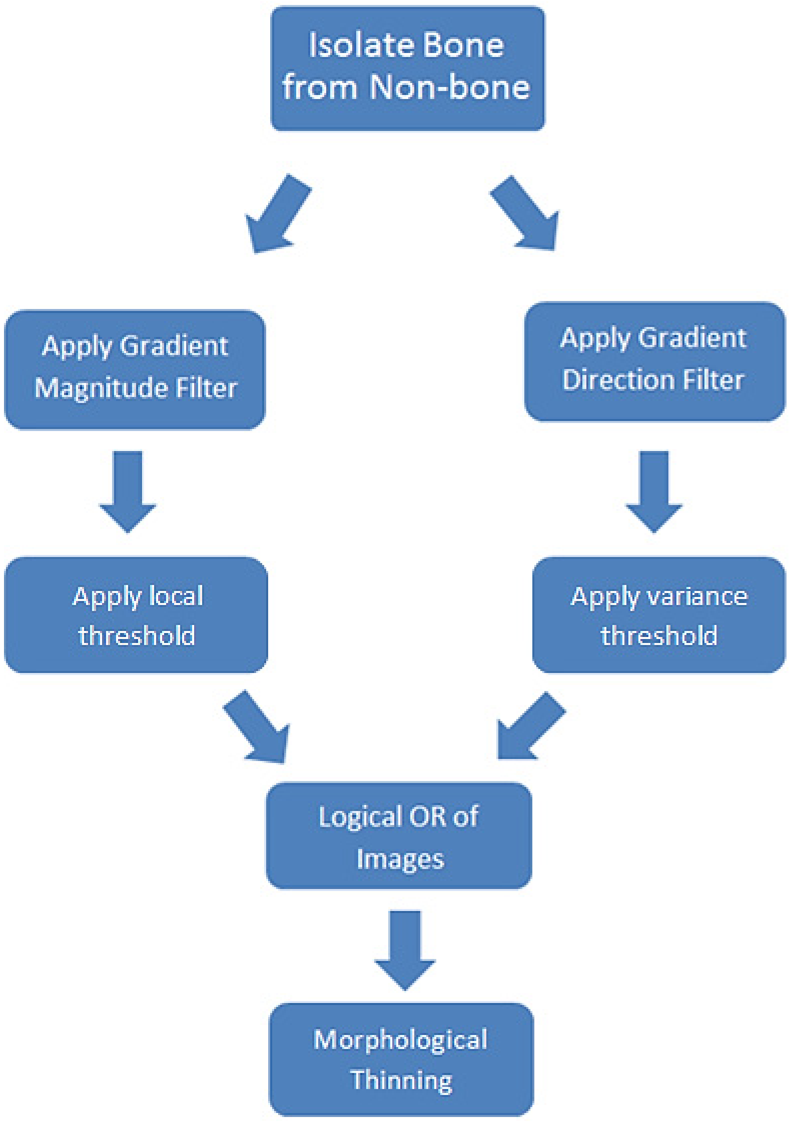

Isolate Bone from Non-bone
Apply Gradient Magnitude Filter
Apply Gradient Direction Filter
Apply local threshold
Apply variance threshold
Logical OR of Images
Morphological Thinning

Figure 2.1 Graphical representation of the edge isolation algorithm. The process ends at morphological thinning. Note that there is a split in the flowchart where the gradient magnitude image and the gradient direction image are treated separately. Doing this assures that if one step destroys part of an edge, the OR of both paths may fix it before the thinning process. Not shown is a smoothing step that is applied after the gradient magnitude filter, before local thresholding is applied. Note that though the gradient magnitude and direction filters are shown being applied simultaneously, the magnitude filter is applied first, followed by the direction filter, since the software implementation does not use multi-threading.

The software begins the process by isolating bone from non-bone. It is a threshold that replaces all pixels with a value beneath the threshold with the value of the threshold. It adapts to each image by selecting a value that excludes as much non-bone intensities as possible without eliminating any bone from the image.

After this, two copies of the resultant image are passed through separate filters. One filter is the gradient magnitude filter. It replaces each pixel in the image with a grayscale value that represents the magnitude of the local gradient in the pixel's 3x3 neighborhood. It is then passed through a mean filter that replaces each pixel in every 3x3 neighborhood with the mean intensity of the neighborhood. This is to reduce the influence of noise on the thresholding that follows.

The other image copy is passed through the gradient direction filter. This replaces each pixel with a value, zero through seven, which represents the compass direction of the gradient in the 3x3 neighborhood of each pixel. The subsequent step thresholds the image in a process this thesis refers to as 'variance thresholding.' Inspired by O'Gorman and Clowes work detecting curves with the Hough Transform (O'Gorman and Clowes, 1973), it assumes that the gradient near an edge has less variance than elsewhere in an image. It converts each pixel in the gradient directions image to a unit vector and sums the pixels in each 3x3 neighborhood. If the summed vectors result in a line that exceeds a specified radius, the neighborhood is considered to have high 'agreement' and thus low variance.

The thresholding of the images containing the gradient magnitude and direction results in binary images where pixels with a value of one are interpreted as near a true edge. This thesis refers to these areas as 'edge-regions.' Once both images are thresholded, they are combined using a logical OR operator. This step functions just like an OR in computer science and produces a binary image whose edge-regions include those from both the gradient magnitude and gradient direction images. The motivation for combining the two images is to prevent the loss of an edge in the thresholding processes; if an edge is severed in the gradient magnitude filter, it may not be in the variance filter, and vice versa.

After combining the two binary images into one, the edge-regions can be reduced to edges via morphological thinning (Changxian 1998). These one-pixel-wide lines are extracted from the exterior in each object in the image. This provides a means for navigating the CT series three-dimensionally to find the best image for measuring the BDI.

The following section describes in detail the operation of each step in the process of isolating edges. The section that follows describes the method by which the ‘best’ view of the BDI is found and calculated.

Algorithms in Detail

Distinguishing Bone from Non-bone

A CT scan produces an image based on the density of the material scanned. Within a CT image, denser material, such as bone, appears with higher intensity (lighter) pixels, whereas less dense material, such as skin and muscles, appears with lower-intensity (darker) pixels. In order for the software to distinguish the difference between bone and non-bone, it must first understand the difference between “light” and “dark.” This subsection describes the approach used in the algorithm to distinguish the light pixels representing bone material from the darker pixels representing non-bone material.

The difficulty in teaching a computer to focus on bone is that there is no affirmative pixel value for bone in any CT scan; bones in one part of the image may appear brighter or darker than other bones in another part of the image. The reasons for these differences are numerous, including density, focus, orientation, etc. Bones between series are not guaranteed to have the same hue either, as differences in the patient, equipment, and technician all contribute to minute differences between series' appearances.

Thus, selecting a specific threshold for all CT scans to help the computer know the difference between bone and non-bone is an unsuitable approach. However it is necessary for the computer to make this distinction, as this prevents the program from taking the gradient of pixels that do not correspond to bone; once the gradient magnitude or gradient direction filter is applied, it is impossible to detect whether a pixel corresponds to bone or otherwise. This is problematic, as a CT scan may contain sharp edges that would be prominent in a gradient magnitude image where there is no bone, such as the back of the neck or the esophagus in *Figure 2.8*. Therefore, a particular value is chosen for each image in order to draw the line between bone and flesh. This is done by considering every pixel's intensity in the image.

First, the number of possible intensities is determined by subtracting the value of the maximum intensity in the image from the value of the minimum intensity. Using this information, a histogram of the image's intensities is constructed. The histogram is converted to a probability density function by dividing each value in the histogram by the total number of possible intensities. This is then converted to a cumulative density function by replacing each value in the probability density function by the cumulative sum. A generic cumulative density function can be seen in *Figure 2.2*.

Next, the threshold is determined from the cumulative density function. This is done by first assuming that a CT-scan will have a bimodal distribution of pixels; a grouping of bright pixels that correspond to bone and a grouping of darker pixels that correspond to everything else. In the cumulative density function of a CT-scan, there will be two rises in the function that occur at the intensities where these groupings lie. To select the threshold, the cumulative density function is compared to that of a hypothetical image with the same range of pixel intensities, but no modality in pixel distribution except for having the same proportion of minimum pixels as the CT-scan.

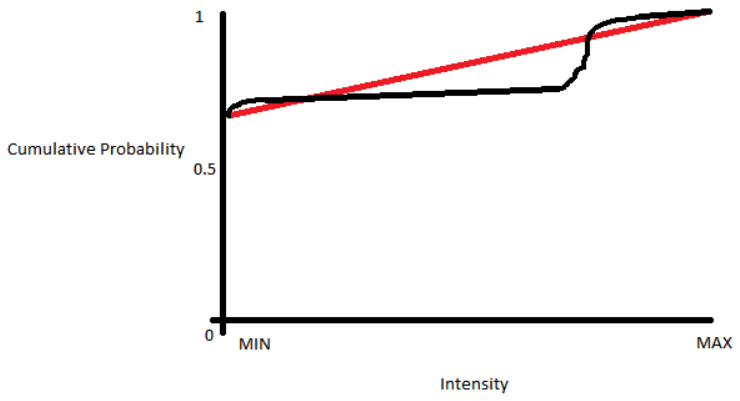


Figure 2.2 A generic representation of the cumulative density function, or just cumulative probability, of a CT-scan image. Global thresholds are determined from this distribution by comparing the CDF of an image's pixels to that of a hypothetical image, shown in red, that has the same minimum and maximum intensities as the CT-scan. In the hypothetical image, each intensity has equal representation, thus giving rise to a flat probability distribution (except for the minimum value), hence the steady slope in the CDF. The threshold is determined by starting at MAX in the spectrum and computing the left-hand delta of each intensity, comparing it to 2x the slope of the hypothetical image's CDF. If the delta of a particular intensity is greater than 2x the slope of the hypothetical CDF, that intensity is made the threshold for the whole image.

The cumulative distribution function would be a straight line beginning at the proportion of minimum pixels and increasing to one. Taking advantage of this, the threshold is chosen to be the brightest intensity whose probability on the cumulative distribution function is increasing with respect to intensity twice as fast as the hypothetical distribution. The reason for doubling the slope of the hypothetical distribution is to push the threshold down to the left to ensure that all pixels that correspond to bone are included. Every pixel whose intensity is less than the threshold is replaced by the value of the threshold. All other pixels are left the same.

The result of using this threshold is to allow the computer to distinguish edges that are associated with bone from the edges of other objects. *Figure 2.13* shows that after applying the filter, the sharp edge produced by the back of the neck has almost completely disappeared from the gradient magnitude image. The next section details how the gradient magnitude is calculated.

Computing the Gradient Magnitude of an Image

The next step in processing the image is to compute its gradient magnitude. The reason for not using the threshold generated in the previous step as an absolute determiner of the edge of bone is because the edge of a bone as perceived by humans does not immediately begin where the hue of flesh ends, but is where the hue of the bone is changing most rapidly near where flesh meets bone. This necessitates an algorithm that produces an image whose pixels represent the magnitude of the gradient in the original image. Calculating the gradient at each pixel only considers the three-by-three neighborhood of each pixel. It is only necessary to consider how the gradient is immediately changing. The process of retrieving the neighborhood is relatively easy:

store the eight surrounding pixels in an eight-membered array, where the elements 0 through 7 are ordered clockwise around the center pixel, starting with the pixel directly above the center and ending with the pixel to the upper left of the center.

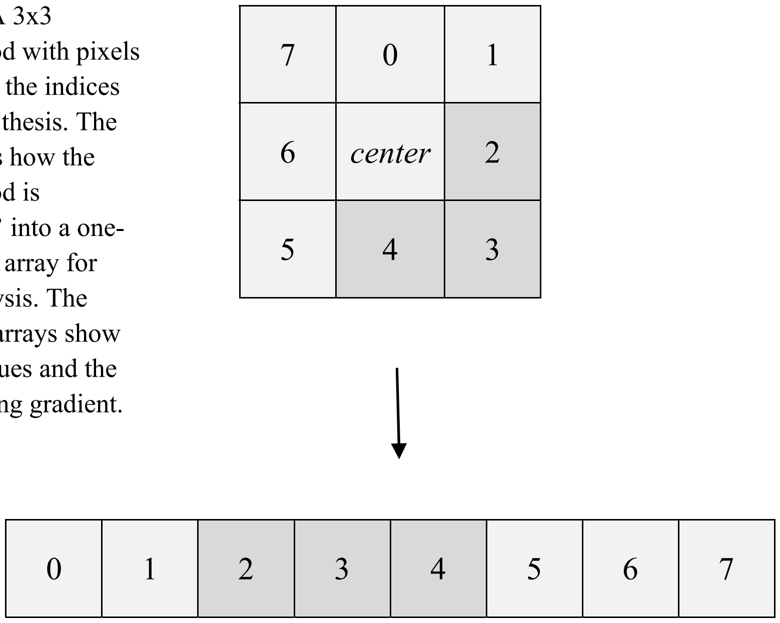


Figure 2.3 A 3x3 neighborhood with pixels labeled with the indices used by this thesis. The arrow shows how the neighborhood is 'unwrapped' into a one-dimensional array for further analysis. The subsequent arrays show example values and the accompanying gradient.

The array contains the intensities of the surrounding pixels. If we assume the dark gray pixels have a value of 20 and the light gray pixels have a value of 100, the array would contain the following data:

100	100	20	20	20	100	100	100

The gradient of the neighborhood is then calculated by subtracting the value of each element from the element to its right in the array, except at member 7 whose value is subtracted from member 0. The result using the previous example would produce the array at the top of the next page.

0	0	-80	0	0	80	0	0

From this point, calculating the gradient magnitude is trivial: simply subtract the minimum member of the gradient array from the maximum member and divide by 2. Thus the pixel *center* will have a value of 80 in the gradient image since it is in the neighborhood of an 80-pixel gradient.

Smoothing the Gradient

After the gradient has been computed, it must be smoothed out to reduce noise. This is done *after* taking the gradient instead of before since smoothing filters remove information from the image, and it is best for complex algorithms to have access to as much information as possible. However, the next step requires the image to be turned into a binary image. Since a yes-no decision is going to be made for every pixel, it is best to make sure that noise is not a factor in making a decision at that step. The smoothing method used is a mean filter on the three-by-three neighborhood of each pixel.

Each pixel in the resulting image is the average of the nine pixels in its three-by-three neighborhood in the original image. Applying the mean filter to the gradient magnitude output makes a significant difference in the output of locally thresholding the image.

Locally Threshold the Gradient Magnitude

After the gradient magnitude image has been created and filtered, a threshold is applied to it to create a binary image where pixels with a value of 1 are near the true edge between bone and non-bone. In order to do this, it is assumed that bright pixels in the gradient image correspond to an edge, since the human eye detects edges where an image is changing rapidly.

Thresholding the gradient magnitude with a global value for the entire image is unsuitable, since a few bones are likely to have some faint edges. Instead, a local threshold is applied to each pixel's 3x3 neighborhood.

The algorithm will set the corresponding pixel in the output image to 1 if the center pixel in the original image is at least brighter than a specified percentage of the difference between the maximum and minimum pixels in its neighborhood. *Figure 2.14* and *Figure 2.15* demonstrate the effects of this form of local thresholding on the gradient magnitude, with *Figure 2.15* showing the output of different percentages. The end product uses a percentage of 50%, so if a pixel is brighter than the average of the maximum and minimum pixels in its neighborhood, it will become a 1.

After this binary image has been created, it is set aside temporarily while the gradient direction of the image is computed and thresholded.

Computing the Gradient Magnitude of the Image

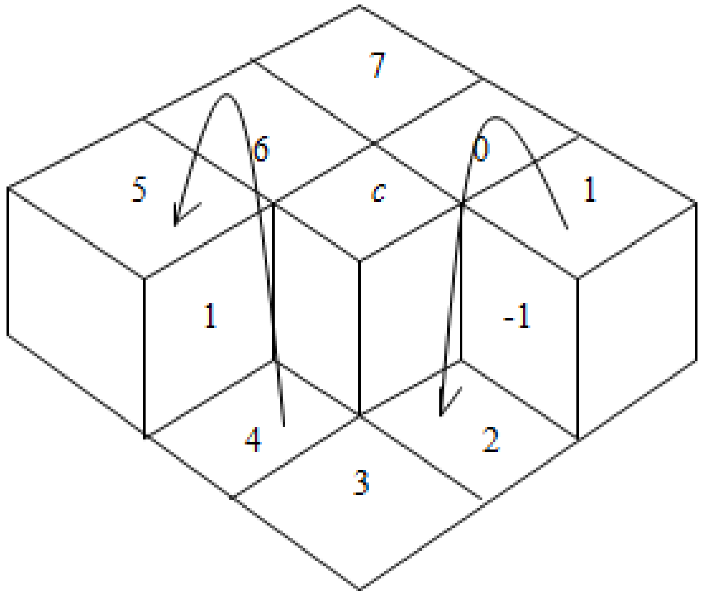
Since a gradient of a function not only has magnitude, but direction, the gradient magnitude function also produces an image of the gradient direction of a CT scan. This information is useful because consistency of the gradient's direction can provide more reliable edge information than the gradient's magnitude in noisy scenes (Law et al. 2006). This thesis refers to the inconsistency of an edge as 'variance.'

Computing the gradient direction of an image is similar to the process of taking the gradient. But before continuing, consider the following neighborhood:

Figure 2.4.a An example 3x3 neighborhood. Light pixels have a value of 1, dark pixels have a value of 0.

7	0	1
6	*c*	2
5	4	3

Imagine that the pixels are blocks whose height is indicated by their intensity. So, let the dark pixels have a height of 0, and let the light pixels have a height of 1. This leads to the following three-dimensional interpretation of the above neighborhood:



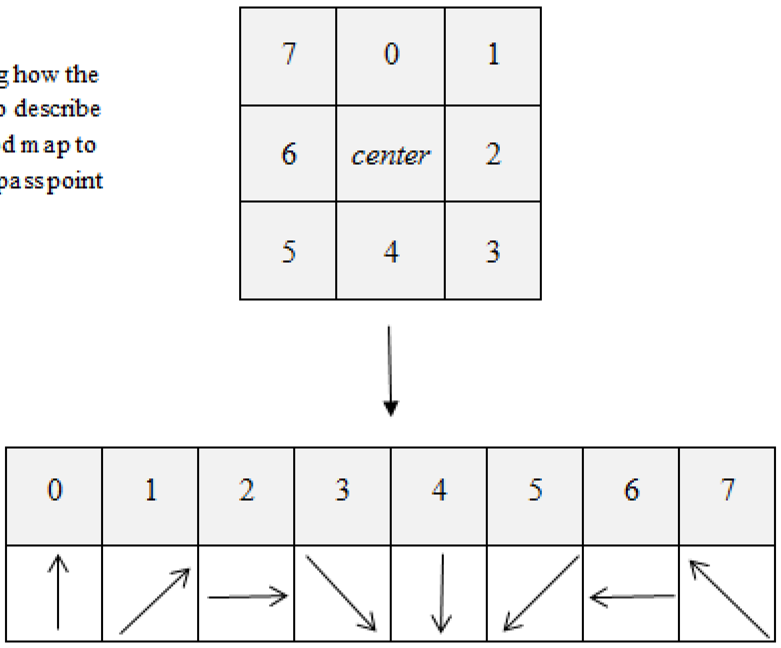
Gradient:

0	0	-1	0	0	1	0	0

Figure 2.4.b The middle image shows a three-dimensional interpretation of the neighborhood in *Figure 2.4.a*. The bottom image is the gradient of the neighborhood, calculated in the same manner as it is in the gradient magnitude filter.

When the gradient is computed using the same method as in the gradient magnitude, the array contains the height of the jump that needs to be made in order to reach the next pixel when moving around *c* in a clockwise manner. Furthermore, notice that the minimum and maximum values in the array correspond to the points where the edge passes through the neighborhood. For consistency, the direction of the edge is considered to always point from the pixel corresponding to the minimum value in the array to the maximum. The direction of the edge is chosen from the eight compass point directions. *Figure 2.5* shows how these directions are mapped to the eight indices ranging from zero to seven used to denote the neighborhood's pixels relative to the center.

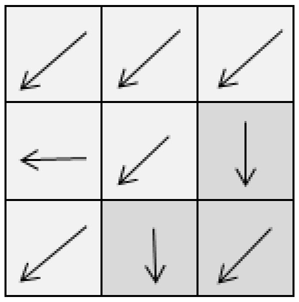
Figure 2.5
Demonstrating how the indices used to describe a neighborhood map to the eight compass point directions.



We store the direction of the edge as a value from *Figure 2.5* directly in the pixel corresponding to *c* in a temporary image. An example of this is shown in *Figure 2.9*, with each direction being assigned to a grayscale value.

When all of the edge directions are computed and stored in the temporary image, each three-by-three neighborhood in the temporary image is scanned. A generic neighborhood might look like this:

Figure 2.6 A generic neighborhood with edge direction arrows overlaid. Note how the arrows do not all point in the sam e direction, but tend point along the direction of the edge's propagation.



If we sum up all the unit directions stored in the pixels, we get a line whose magnitude represents the agreement of all the directions stored in the neighborhood. If the resultant line is short, this means that the presence of an edge is weak in this neighborhood. If the line is long, this means that an edge's presence is strong. We can threshold what we consider to be adequate agreement in the neighborhood by specifying a threshold radius:

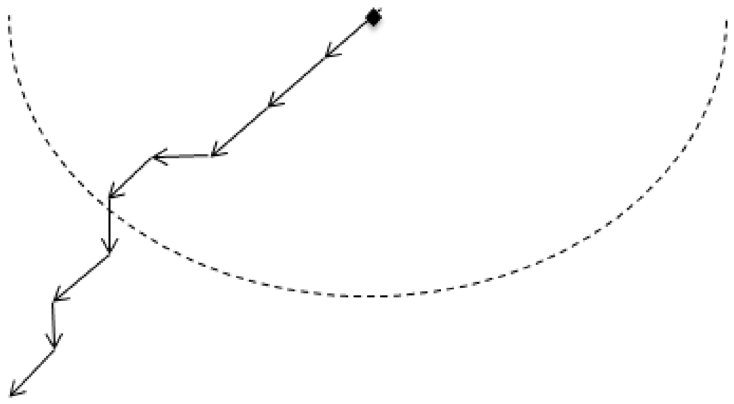


Figure 2.7 The vector sum of the edge directions in *Figure 2.6*. The dashed line represents a threshold. If the sum results in a line that exceeds this radius, the center pixel of the corresponding neighborhood in the output image is changed to a 1. Otherwise it is a 0.

If the summed directions stored in the neighborhood exceed a radius of 7.07, then the pixel in the output image that is in the center of the neighborhood is set to 1, otherwise it is set to 0. *Figures 2.16*, *2.17*, and *2.18* demonstrate the output of variance filtering. Thus pixels of a value of 1 in the binary image are pixels that are near an edge.

Combining the Variance and the Gradient

Now that both the variance and gradient have produced binary images that represent edge regions, it is time to combine them so that any pixels that may represent an edge that were missed in one image will be added to the other. The logical equivalent of this is a logical OR, so we essentially create an image that is the addition of the binary images produced by the variance and gradient. *Figure 2.21* shows the combined output of the variance and gradient filters.

Isolating the Edges of Bone

The product of the previous processes is a binary image where pixels with a value of 1 are pixels that are near a true edge. By assuming that the pixels that represent the true edge are in the middle of these regions, the next step in isolating them is to 'eat away' the outside pixels so that nothing but a line is left (Changxian 1998).

The method for doing this is a continuous process that scans the image multiple times, removing pixels with each pass. The pixels to be removed have a value of 1 and do not have more than two pixels with a value of 1 in its 3x3 neighborhood that are not touching each other. This ensures that pixels that constitute a line are not removed, but all others are. This is applied to every pixel in an image, making a single pass. Since it is unknown how many pixels need to be removed before only one-pixel-thin lines are left, this process typically takes thousands of passes, making it a time-consuming algorithm especially when executed to completion. Instead, the algorithm stops the first time it performs a pass where it removes less than or equal to 1000 pixels. Observe in *Figure 2.24* how there is little difference between running the algorithm to completion and stopping it before removing all eligible pixels.

Once this has been done, the outside edges of the bones must be isolated from the network of lines that appears inside of each bone as a by-product of this process. This is done by tracing the outside of each bone onto an output image, as seen in *Figure 2.25*.

The preceding series of filters combine to form a robust edge detection method, capable of finding edges even in images like *Figure 2.26* where the edge is fuzzy. Now that the algorithms have been explained, the next section will describe how they are applied.

Applying the Algorithms

Now that the algorithmic approach for isolating the edges of bones has been explained, the following section shall describe how the methods are applied to a CT series to measure the BDI. The software can either attempt to automatically identify the basion and the dens, or let the user mark both with a point region of interest, or ROI, with the ROI tool in OsiriX. If no ROI has been placed on a series, the automatic method is used.

In general, the 'best' view of C2 is the cross-section where the top of the dens is higher than the top of the dens in the adjacent slices in a CT series. The 'best' view of the basion similarly is the cross-section where it is higher than the adjacent slices in the series. Whether the user marks C2 and the basion or not, the software checks adjacent images to find the optimal points to measure the BDI. The basion and the dens do not necessarily have to lie in the same image, as in *Figure 2.30*. When the BDI is measured, it does not factor depth in the CT series, since measurements made by doctors are done in the plane of the image (Chaput et al. 2011).

If the BDI exceeds 10mm, the user is warned that the patient is likely suffering from an injury. The output is temporarily written onto the output image itself, along with the measurement value. The procedure for testing the software consists of a trial series of CT series that were never used to configure the algorithms. The series are evaluated by the software and the output measurements are compared with measurements from Scott and White Hospital.

Identifying C2

Once a CT-scan series is loaded, the program begins its analysis with the image at the middle of the series. It checks to see if the image contains the slice of C2 at the middle of the neck. If it does not contain the cross-section of C2, the program will continue checking images by radiating outwards from the middle of the series. If no proper image is found after checking 10 images, the program assumes that this series is not the sagittal view of the cervical spine.

If the program finds C2, it then traces to the top of C2 in that image and checks if the top of C2 is lower in both adjacent images. If this is not the case, the program will move to the image in the direction where C2's top is increasing in height. Otherwise, the program is ready to begin searching for the basion.

They method by which C2 is identified uses a hard-coded set of descriptors that apply to each shape in the image. The values for these descriptors were set through trial and error.

The descriptors used to identify C2 are as follows:

- Height of the object must be greater than its width.
- Height of the object must be less than three times the width.
- If one were to draw a rectangle around the object that fully enclosed all its parts, the area of this rectangle must be greater than 2,500 pixels and less than 20,000 pixels.
- The perimeter of the object's edge must be at least 250 pixels long and at most 2,000 pixels long.
- The object's upper left corner must not be in the furthest left 10% of the image.
- The object's upper left corner must not be in the furthest right 30% of the image.
- The object's upper left corner must not be in the upper 10% of the image.
- The object's upper left corner must not be in the lower 50% of the image.

All of these items must be met by an object for it to be considered a possible candidate for the trace of C2. If the object passes the test, the program moves to the object's center and traces to the right until it reaches the edge. It then moves up the edge to the top of C2. If the adjacent cross-sections of C2 have lower tips than the current one, then this coordinate of the tip is recorded for later use in measuring the BDI.

Identifying the Basion

The basion, like C2, has a set of descriptors that define the computer's understanding of its shape. Once C2 has been found, the program then searches for the basion in that same image. It then checks the adjacent images to see if the coordinate at the bottom of the basion is higher than the one in the current image. If the basion is higher in adjacent images, the plugin will continue searching for the basion until the highest instance is found. Typically the basion and dens reside in the same image, but in some cases they can be in separate ones. *Figure 2.30* contains an example of when the basion and dens are in separate images. If the basion cannot be found, an error is reported to the user, as in *Figure 2.29*, where the plugin was run on a non-sagittal view of the cervical spine.

The descriptors used to define the basion are as follows:

- If one were to draw a rectangle around the object that fully encloses the shape, the area of more than 100 square pixels.
- The centroid, or the average of the edge coordinates, of the object must be in the left 60% of the image.
- The centroid of the object must not be in the left 10% of the image.
- The bottom of the object must be higher than the top of C2.
- The object's width must be less than 1.5 times the object's height.
- If the area of the object is greater than 400 pixels:
    - The object's centroid must be higher than the center of the imaginary rectangle enclosing the object.

    - The object's centroid must be at least the distance from the left edge of the image equal to the distance from C2's centroid to the left edge minus the width of C2.
- If the area of the object is less than or equal to 400 pixels:
    - The object's centroid must not be in the left 20% of the image.
    - The object's centroid must be at least the distance from the left edge of the image equal to the distance from C2's centroid to the left edge minus the width of C2.
    - The distance between the object's centroid and the top of the image must be less than the distance from C2's centroid to the top minus the height of C2.

If all of these requirements are met, the object is considered the basion. If the bottom of the basion is higher than the bottom of the cross-sections of the basion in the adjacent images in the series, this coordinate is taken down for measuring the BDI.

Measuring the BDI

The BDI is measured as the two-dimensional distance between the top of C2 and the bottom of the basion. Though the three-dimensional distance can be calculated, the study done by Dr. Chaput does not account for depth in his measurements. Thus the BDI is measured only in 2D and displayed to the user as in *Figure 2.28*.

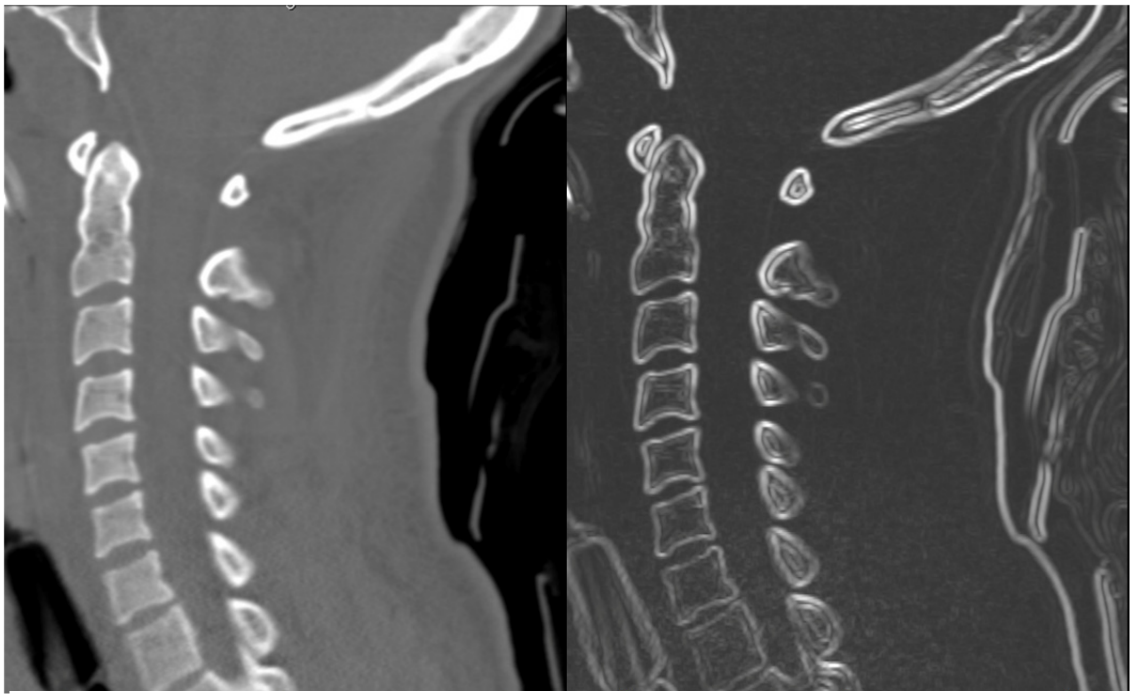


Figure 2.8 Gradient magnitude of an image. The original is on the left, the output is on the right. The output has been scaled to enhance contrast. Note in the output image the areas of greatest intensity are located where edges in the original image are most distinct.

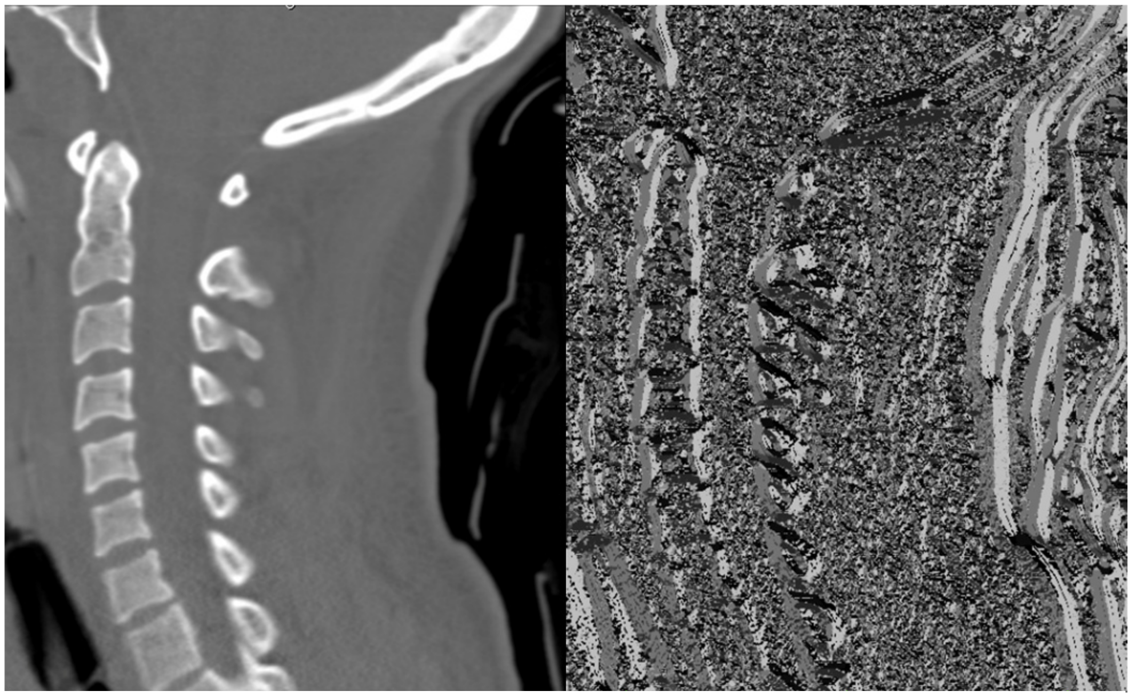
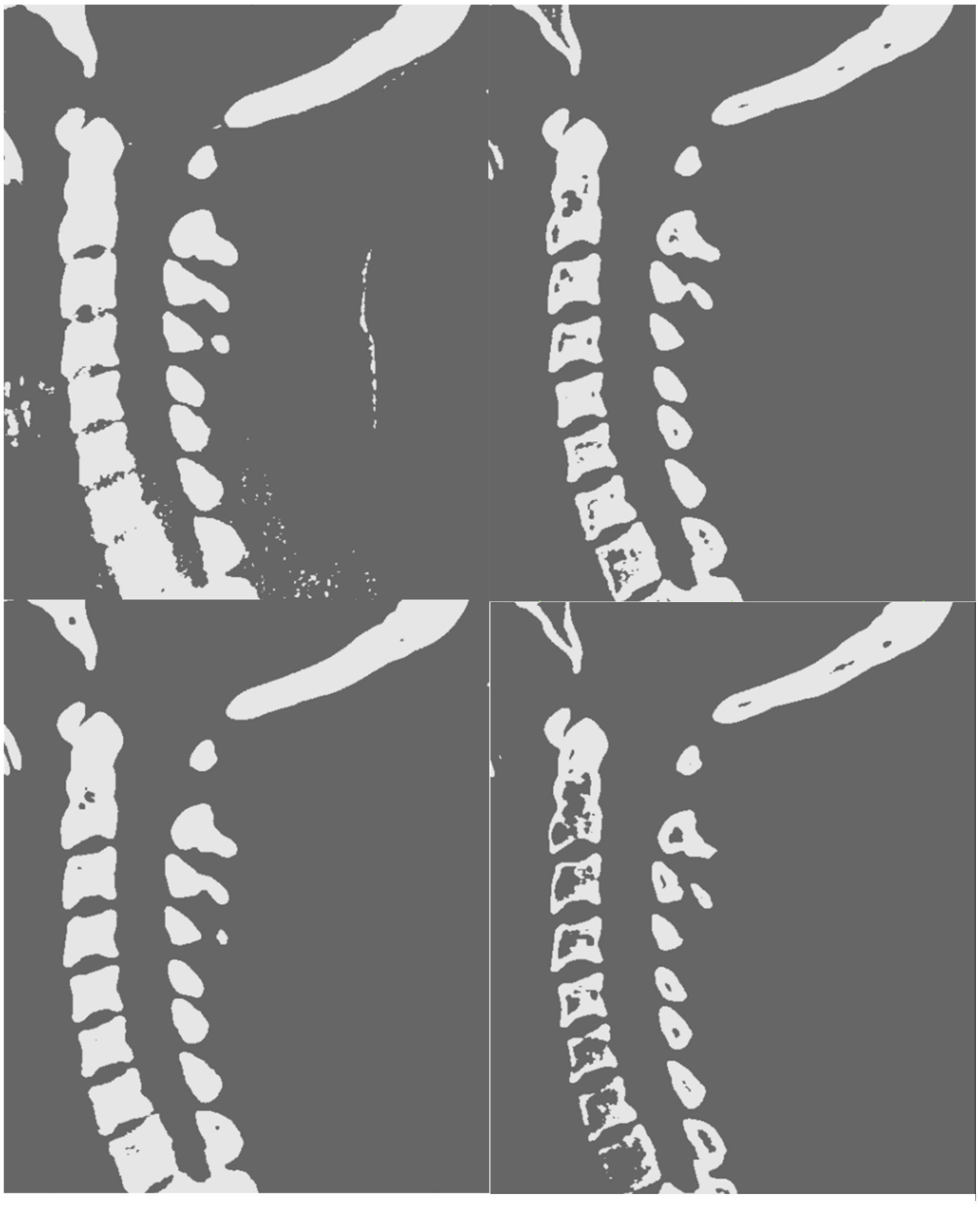


Figure 2.9 Gradient directions of an image. Eight grayscale values represent the eight compass directions that the gradient can 'flow' towards. Note that pixels not associated with an edge are chaotic with respect to their neighbors, whereas pixels located on an edge tend to agree with their neighbors on the direction of the local gradient.



A B
C D
Figure 2.10 Thresholding at arbitrary values. **A** was thresholded at 100, **B** was thresholded at 200. **C** was thresholded at 300, and **D** was thresholded at 400. Note that pixels brighter than the threshold become a 1, and all others become a 0. See that as the threshold increases, more bone data is removed, with image D actually losing entire parts of the spine. The first version of the plugin used 200 as a universal threshold for finding the edges of bone. Since OsiriX stores pixel intensity on an arbitrary scale, a more sophisticated method of edge detection was necessary.

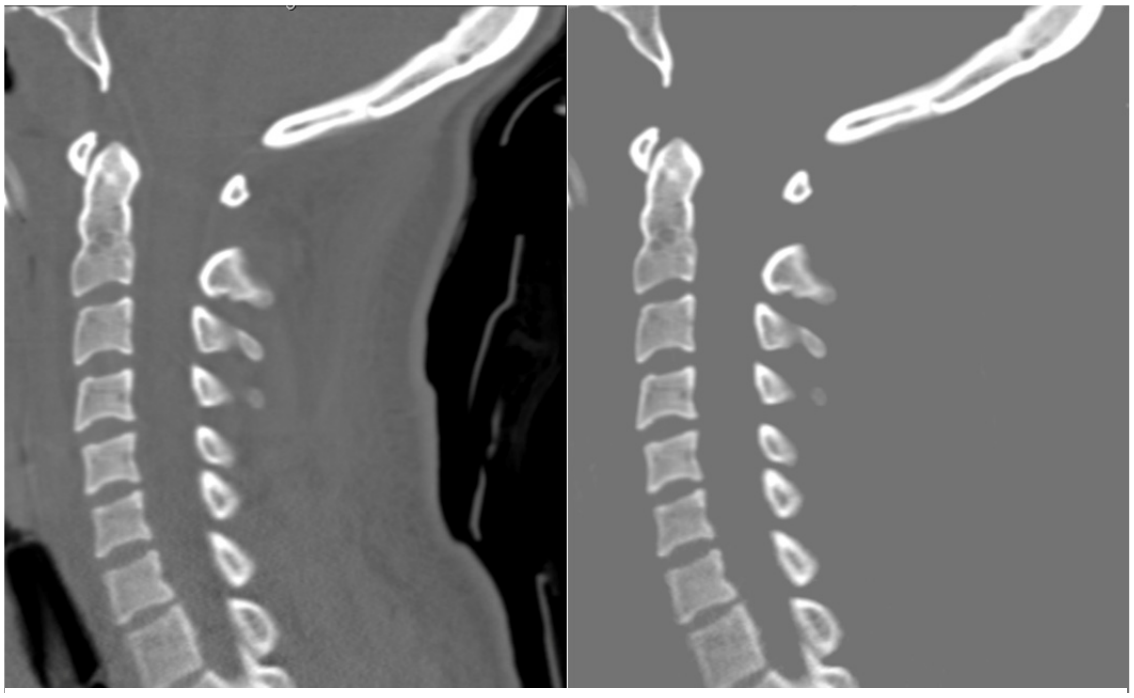


Figure 2.11 Result of Adaptive Global Thresholding. See how the algorithm chooses a value that eliminates all non-bone related pixels. Said pixels are replaced by the threshold value, so that there will not be a sharp edge added to bone where the true edge does not exist.

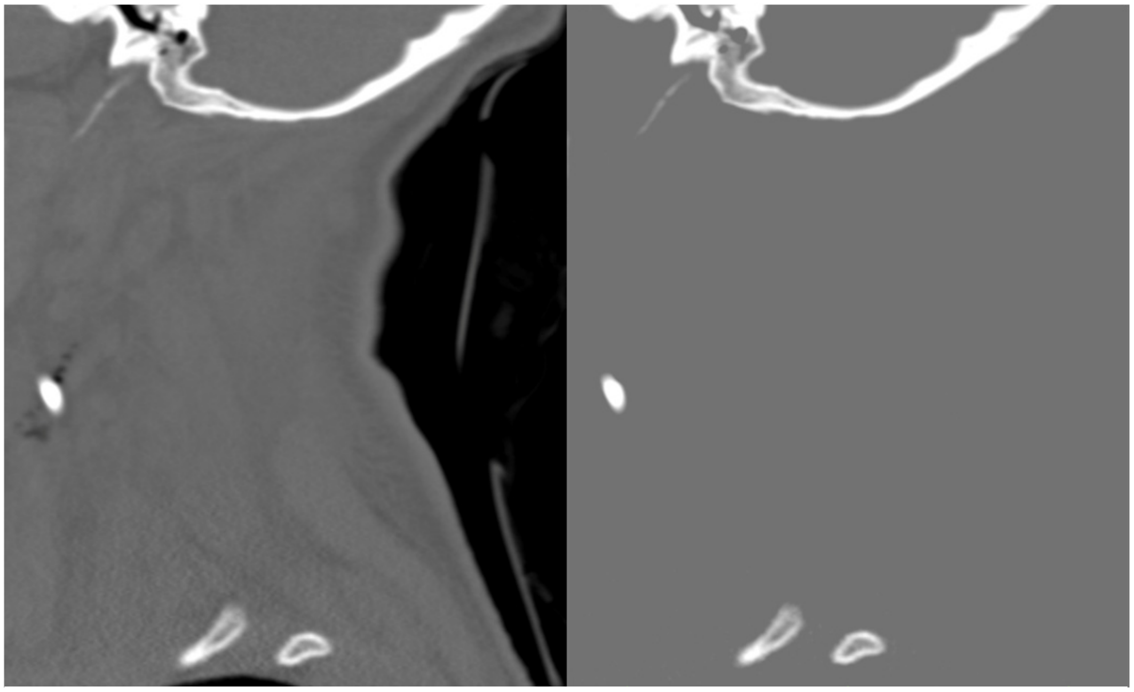


Figure 2.12 Another example of Adaptive Global Thresholding. Note that even with significantly fewer bones in the image, the algorithm still selects a good threshold as long as the original image has a strong bimodal intensity distribution.

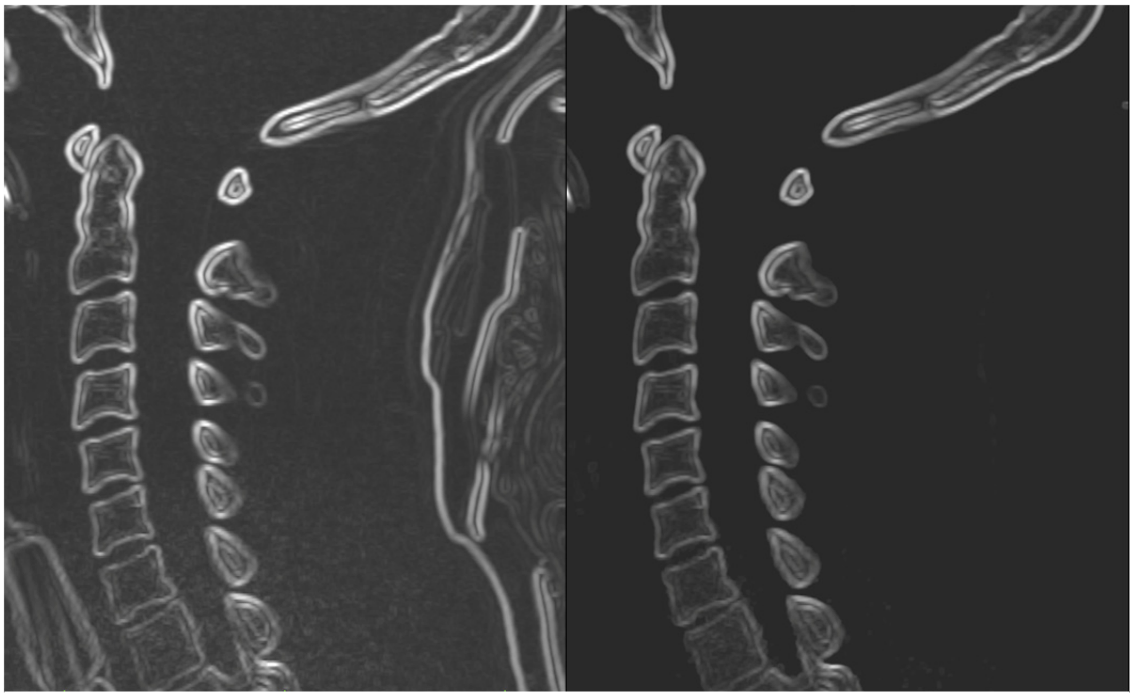


Figure 2.13 Effect of Adaptive Global Thresholding on the gradient operator. See how applying the global threshold helps the computer see the difference between the stark edges of the back of the neck and the esophagus and that of bone.

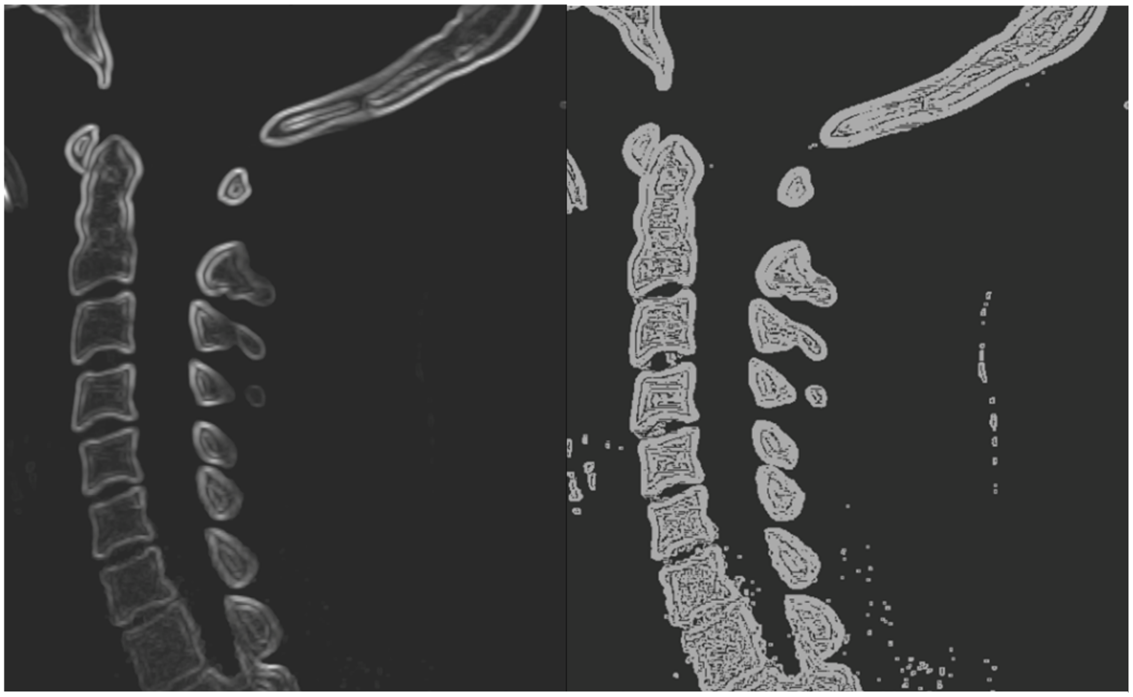
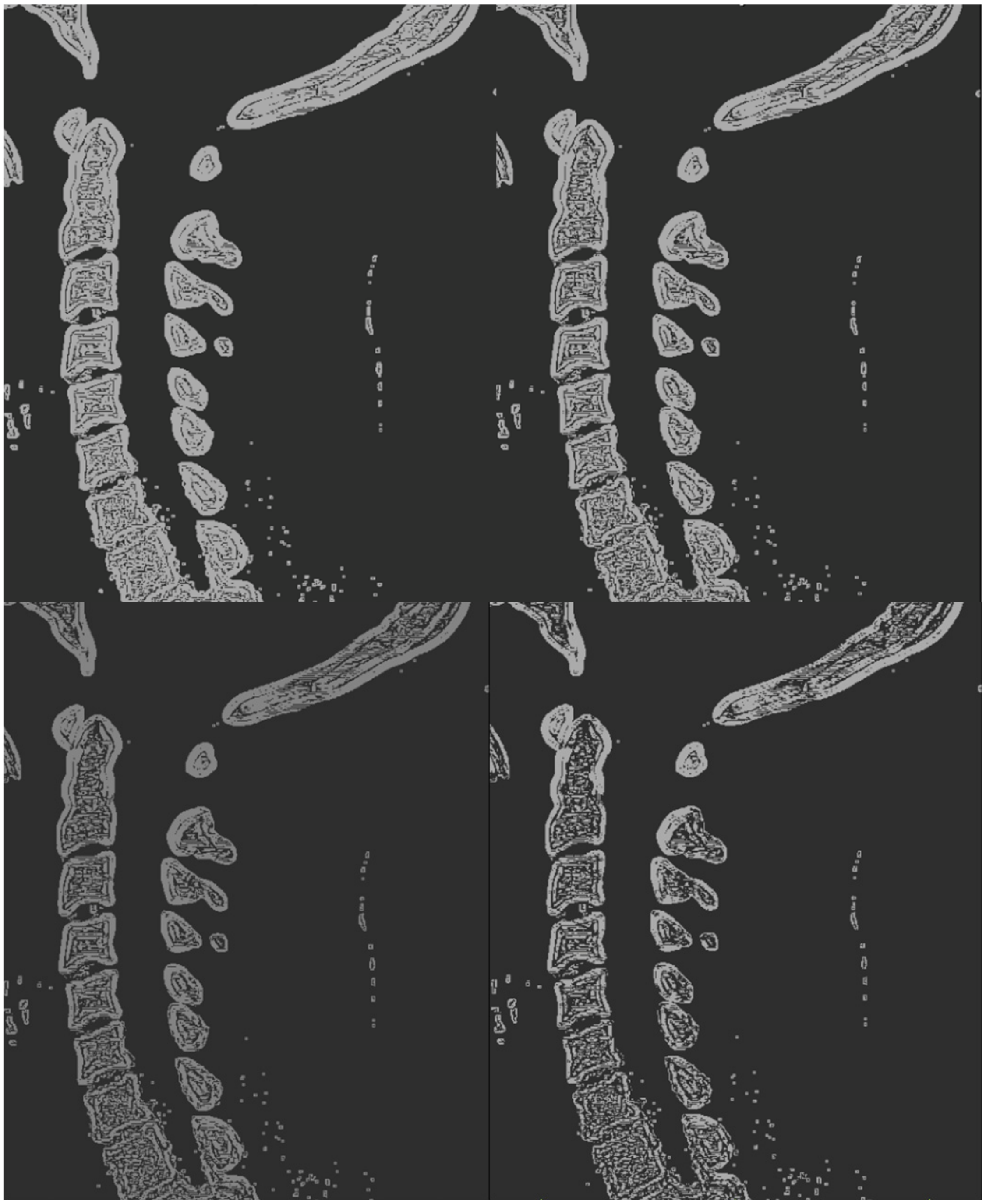


Figure 2.14 Local thresholding of the gradient. The output image retains any pixel that is brighter than 30% of the pixels in each 3x3 neighborhood.



A B
C D

Figure 2.15 Local thresholding of the gradient after global thresholding. Each image retains pixels from the gradient that are brighter than a particular percentage of the pixels in each 3x3 neighborhood. **A**'s percentage is 30%, **B**'s is 40%. **C** is set at 50%, and **D** is at 60%. The 50% used in **C** was used as the metric in the plugin.

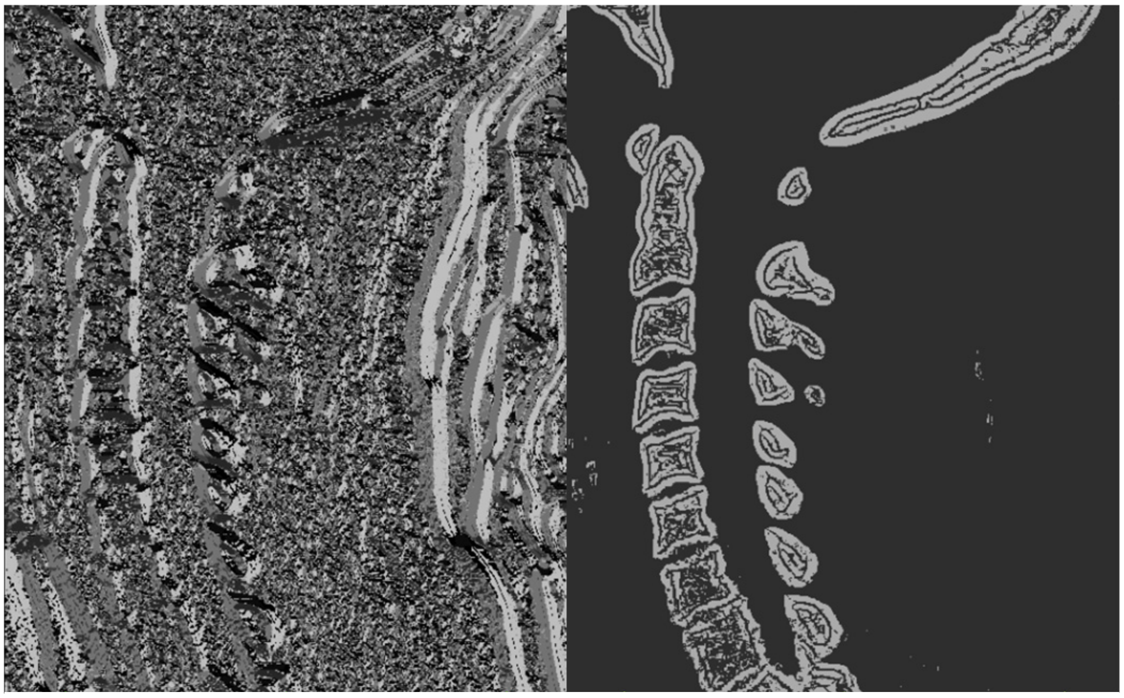


Figure 2.16 The result of variance thresholding. The output image retains pixels whose 3x3 neighborhood in the gradient direction image has 6 pixels of the same value. Note that the output had also been passed through the global threshold to remove the back of the neck.

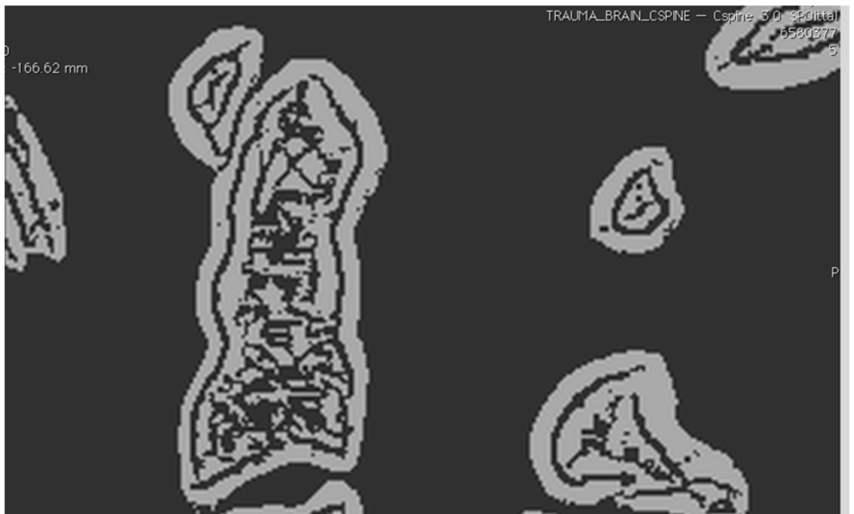
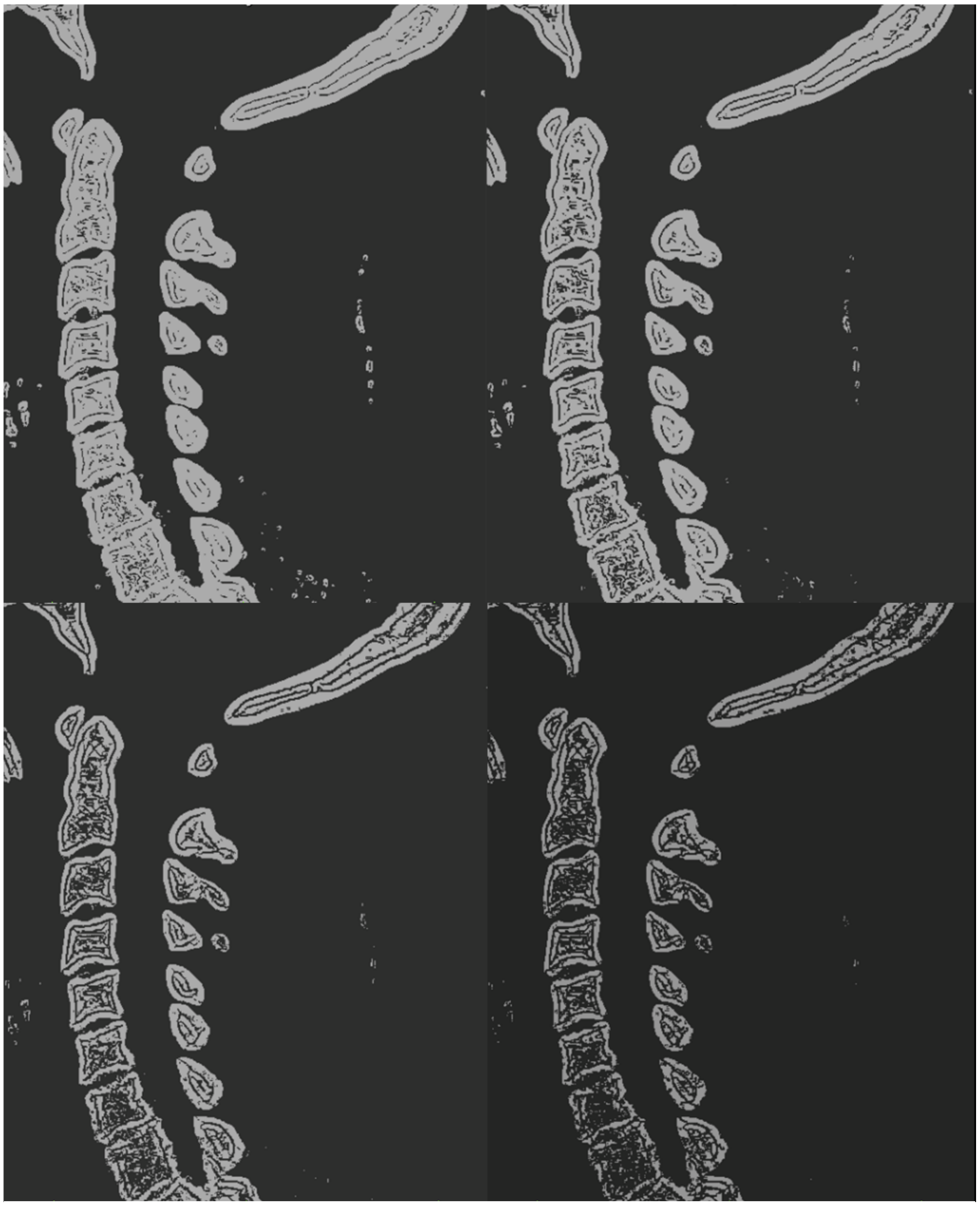


Figure 2.17 A close-up of the output in *Figure 2.16*. See how a solid region forms around the edges of each bone. The center of this region is the location of the true edge as perceived by the human eye.



A B
C D
Figure 2.18 Variance thresholding at increasing radii. The radius in **A** is 3. The radius in **B** is 4. The radius in **C** is 6. The radius in **D** is 7. The term ‘radius’ refers to the sum of the gradient direction unit vectors in each 3x3 neighborhood. Notice that the difference between C1 and C2 becomes evident to the computer in **C**, but in **D** bones are becoming fragmented.

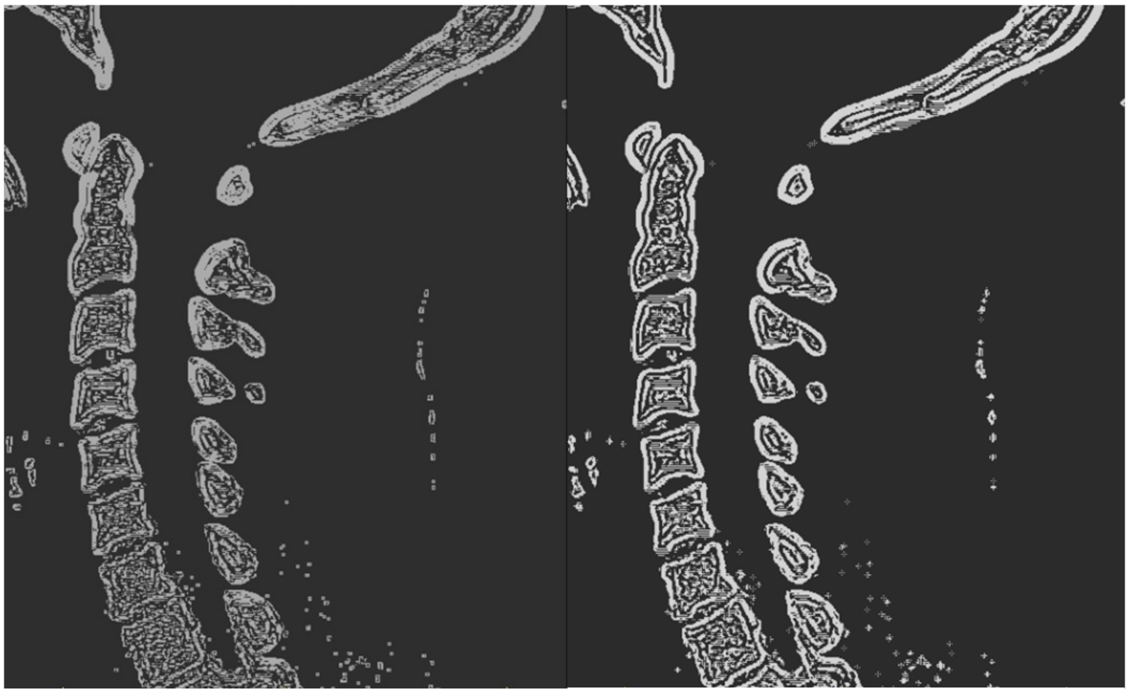
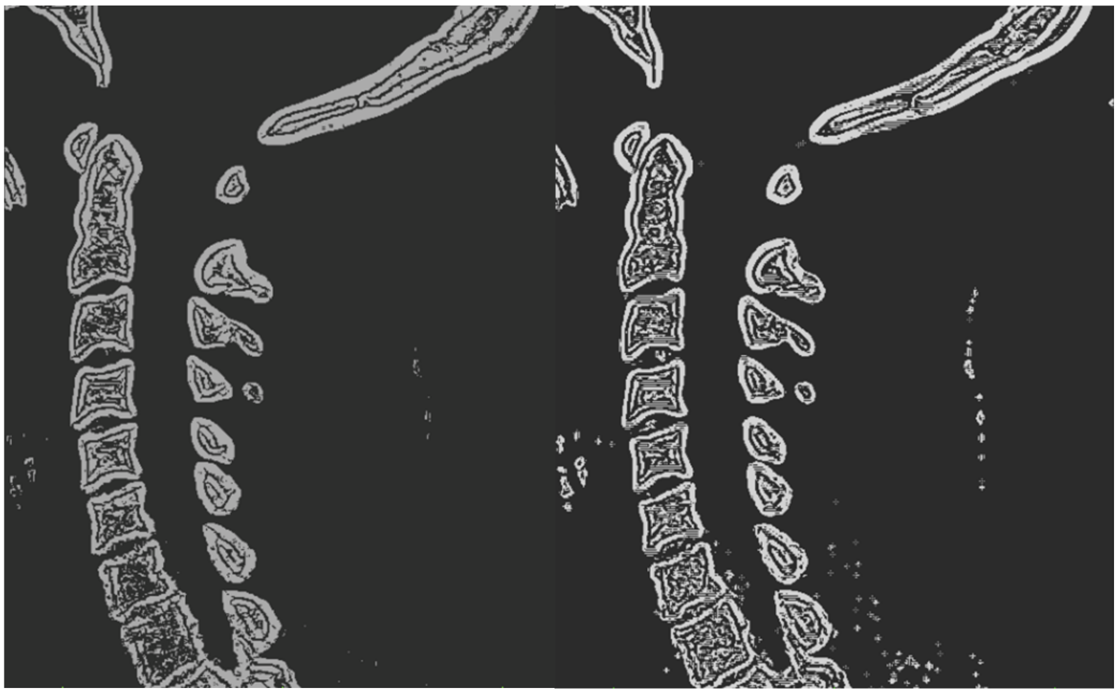


Figure 2.19 Effect of the mean filter. The image on the left is the product of the thresholded gradient without any prior smoothing. The image on the right was passed through a mean filter once before going through the gradient operator. Note how the difference between C1 and C2 is more noticeable after the filter has been applied.



A B

Figure 2.20 Comparison of edge-region detection by means of variance analysis in **A** and by gradient magnitude in **B**.

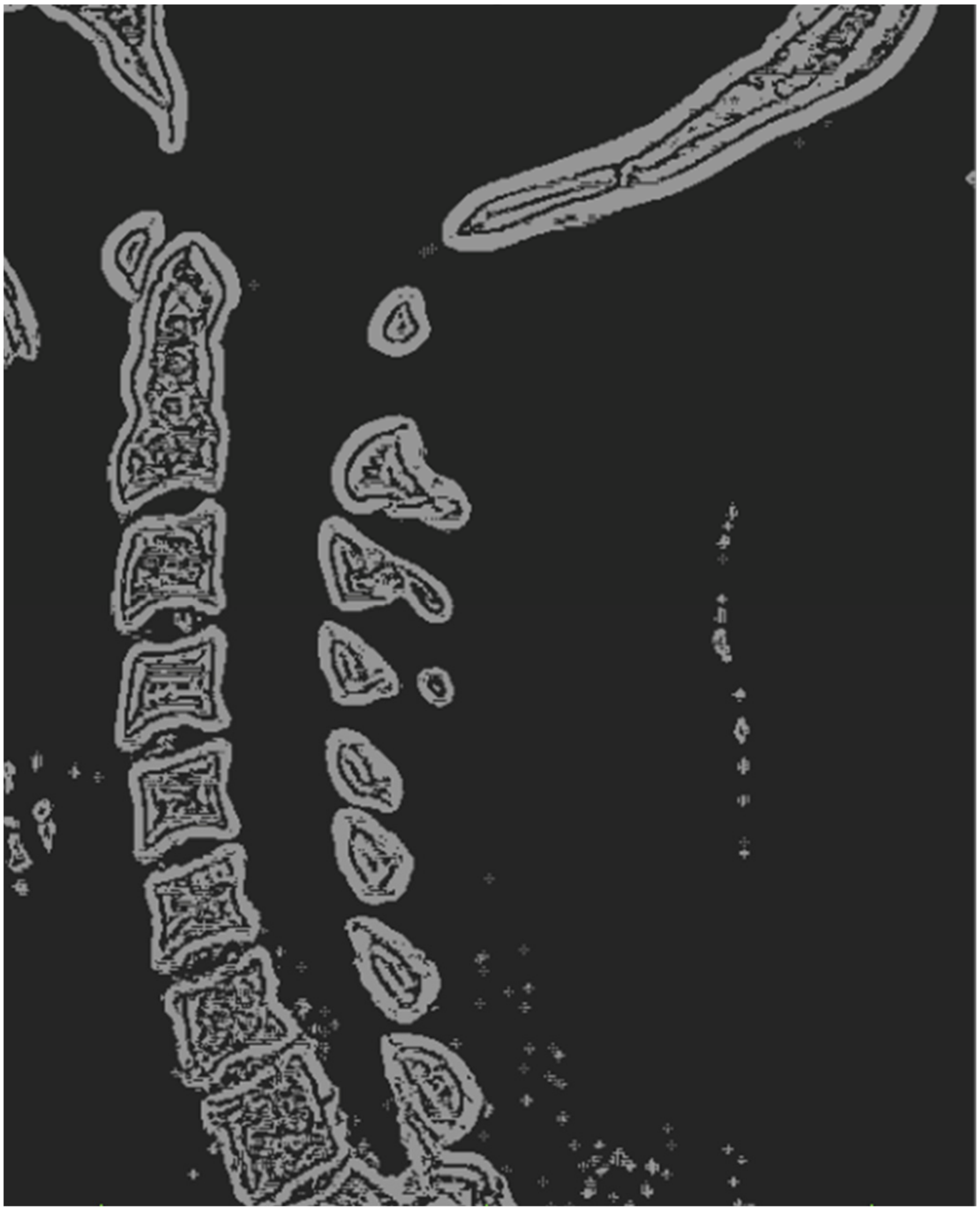
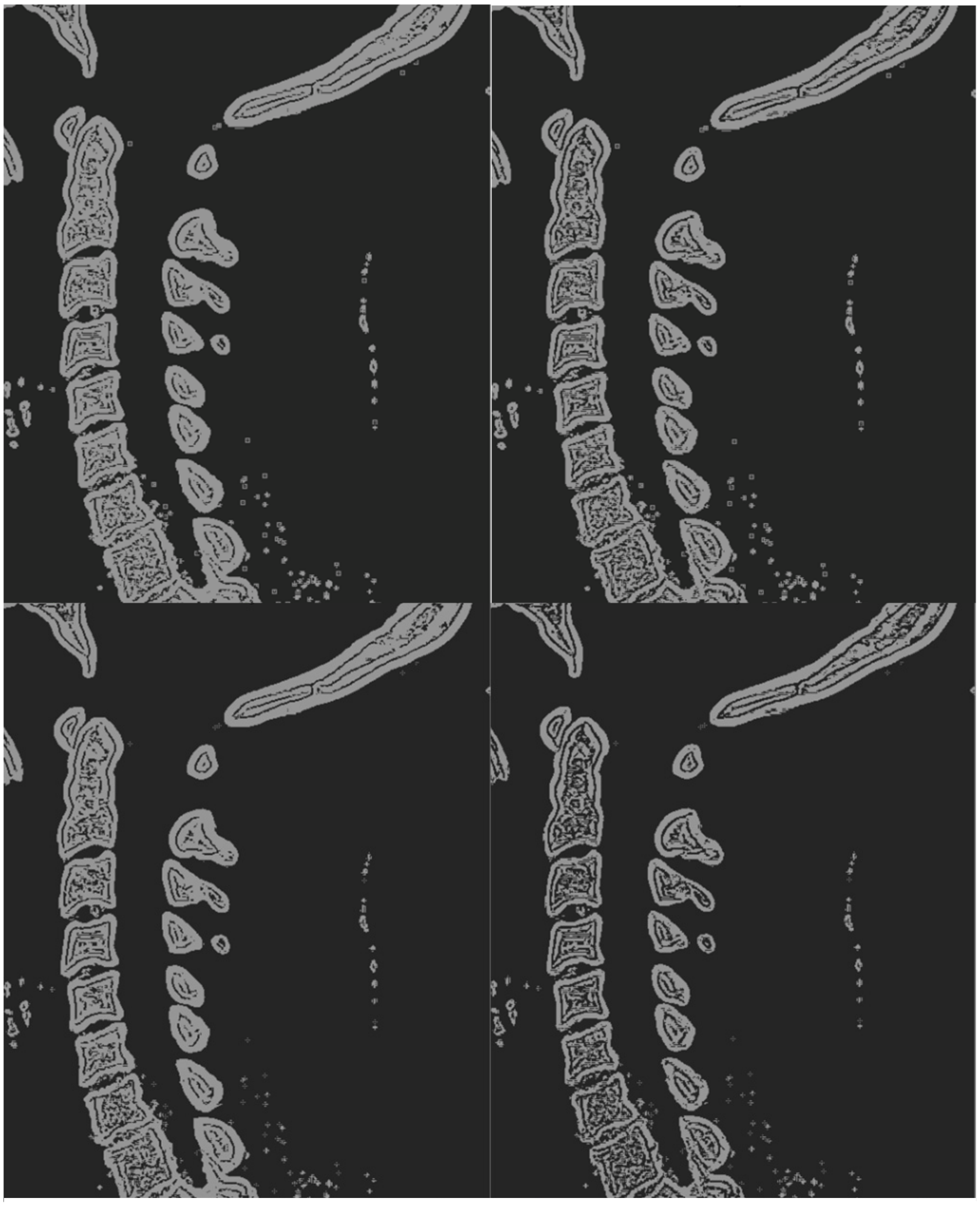


Figure 2.21 The result of combining *Figure 2.20* **A** and *Figure 2.20* **B**. They are combined by an OR operator. The purpose of combining the two images is to reduce the chances of fragmenting an edge. This helps to prevent the edge-narrowing step from destroying an object’s outline.



A B
C D

Figure 2.22 Illustrating the effects of adjusting the parameters of the gradient-based edge finder and the variance-based edge finder. In **A** the gradient was locally thresholded at 40% and the variance was thresholded at a radius of 5. **B** was at 40% and 7. **C** was at 50% and 5. **D** was at 50% and 7. Image **D** is identical to Figure 2.14, and its parameters are used in the final product.

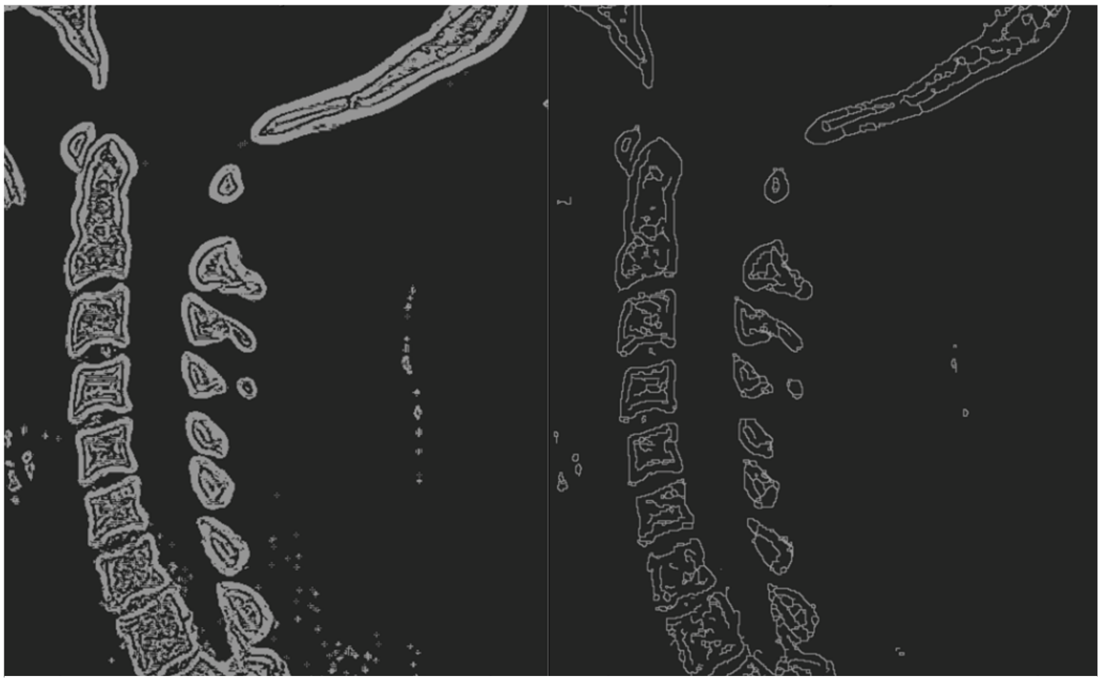
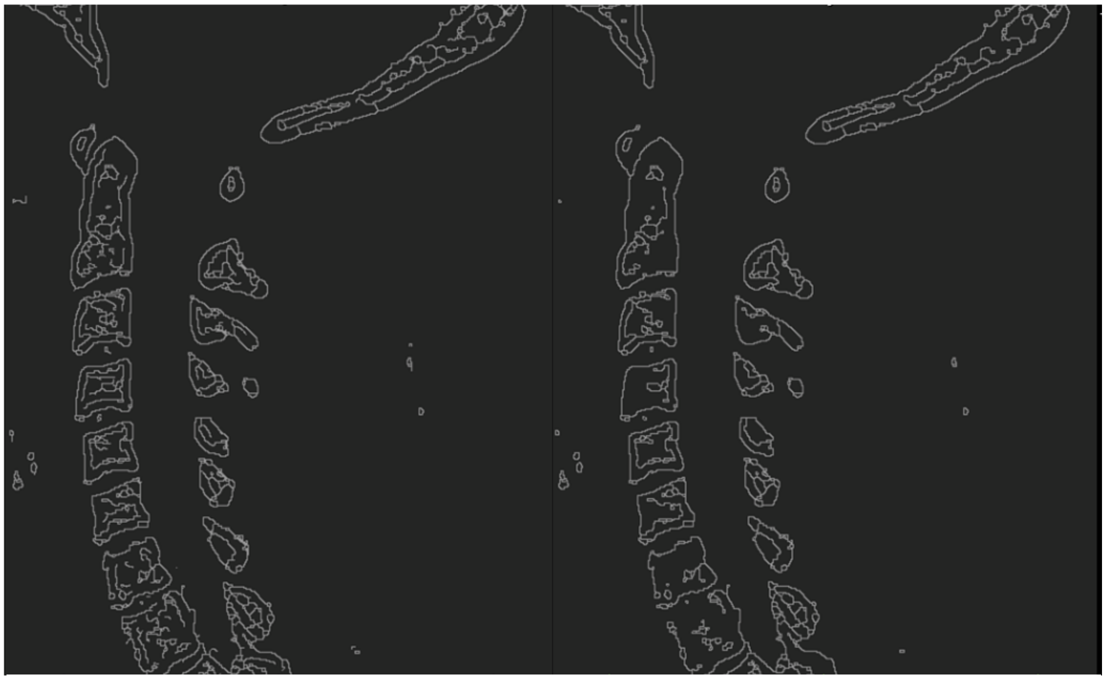


Figure 2.23 An example of thinning the edge regions to a single, one-pixel thick edge. The image on the left is identical to *Figure 2.21*, and the image on the right is the product of the thinning process. Note how closed loops are preserved, whereas line segments are eaten away. The plugin does not fully thin the picture, that is, it does not continue to remove pixels until there are only closed loops.



A B

Figure 2.24 Demonstrating the difference between partial thinning (**A**) and full thinning (**B**). Note that there is little to no difference between the outline of the dens and the basion in **A** and in **B**, however **B** takes a full second to produce whereas **A** takes a fraction of a second. The process in **A** ends when less than 1000 pixels are removed in a pass.

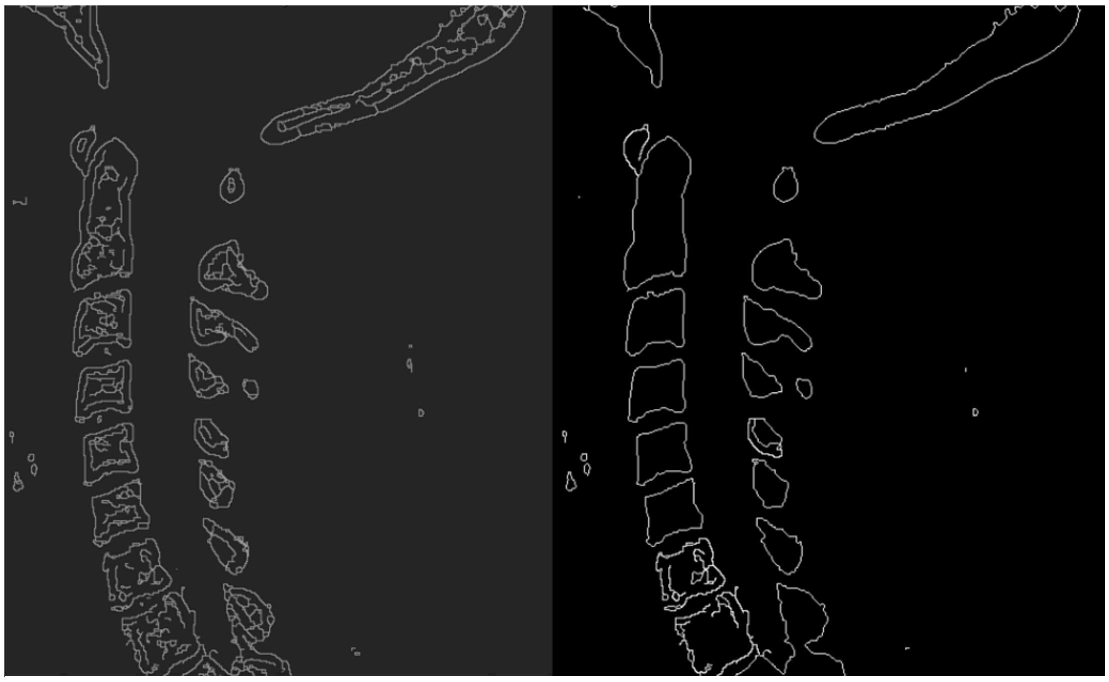


Figure 2.25 Edges are traced out of the thinned image. These edges are used for the subsequent measurement and shape detection methods.

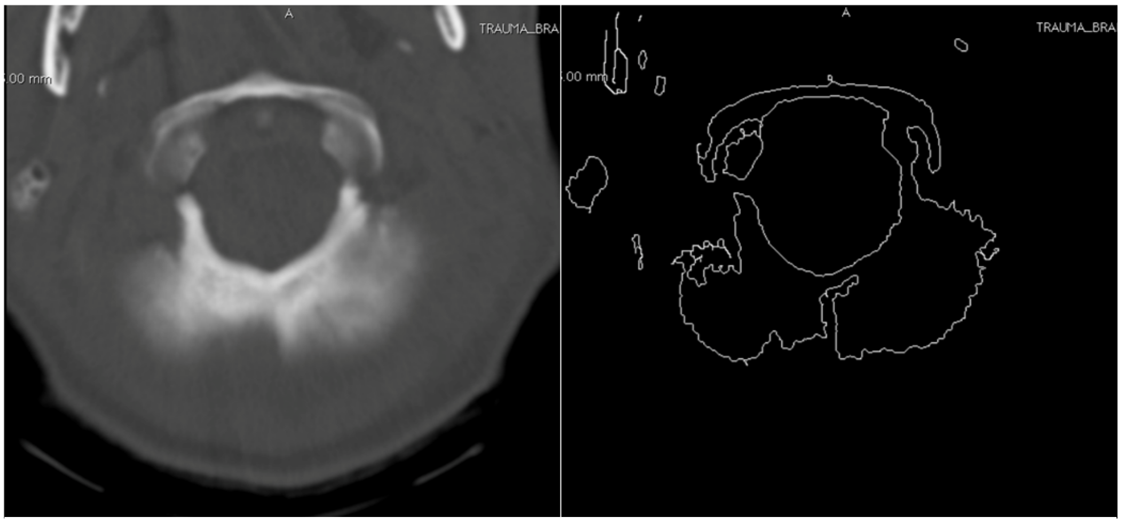


Figure 2.26 The image on the left is a view of the base of the skull and the top of C1. The image was passed through the same edge detection algorithm. The image on the right demonstrates that the algorithm is capable of finding edges even when they are much fuzzier than those found in the sagittal view of the neck.

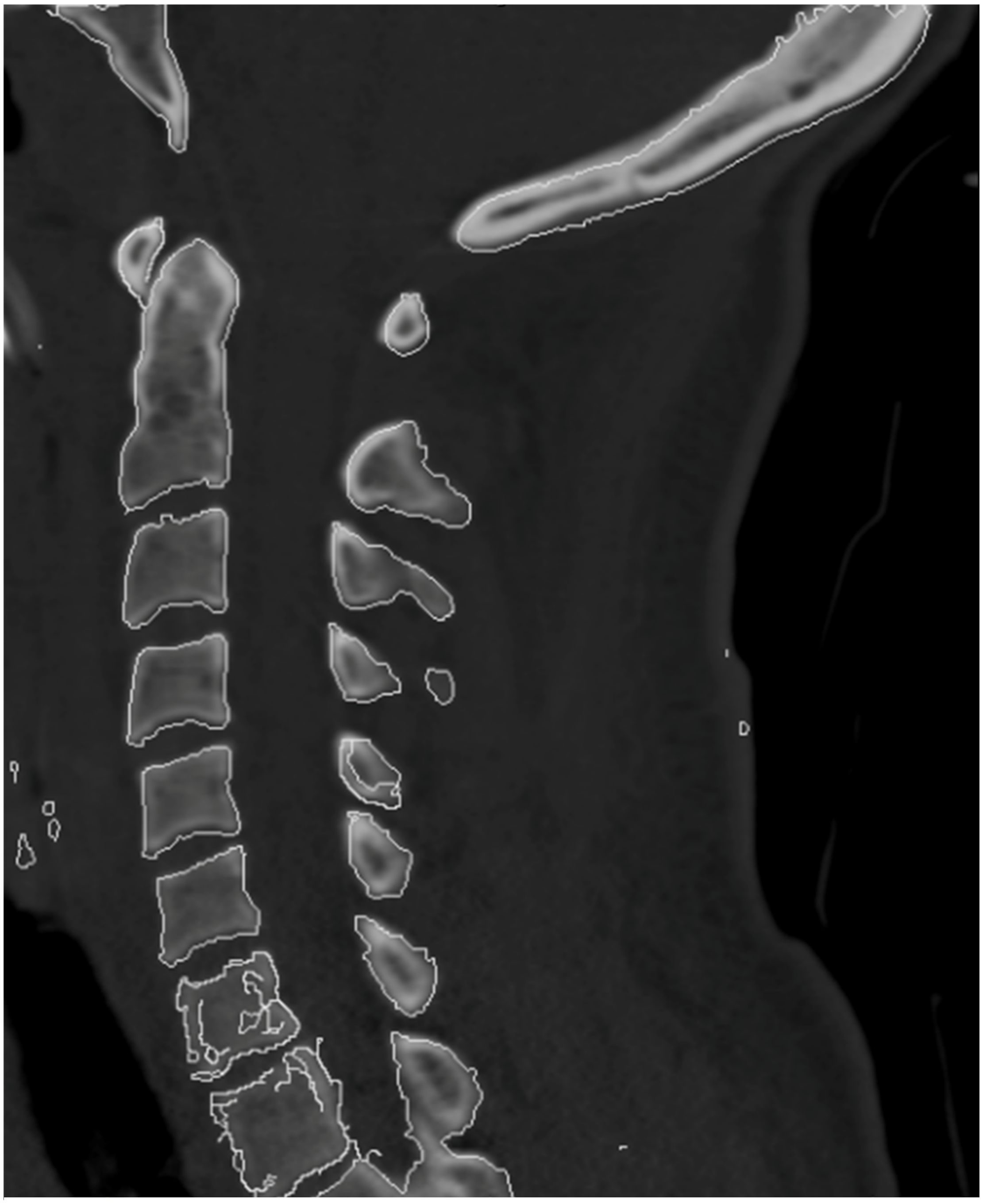
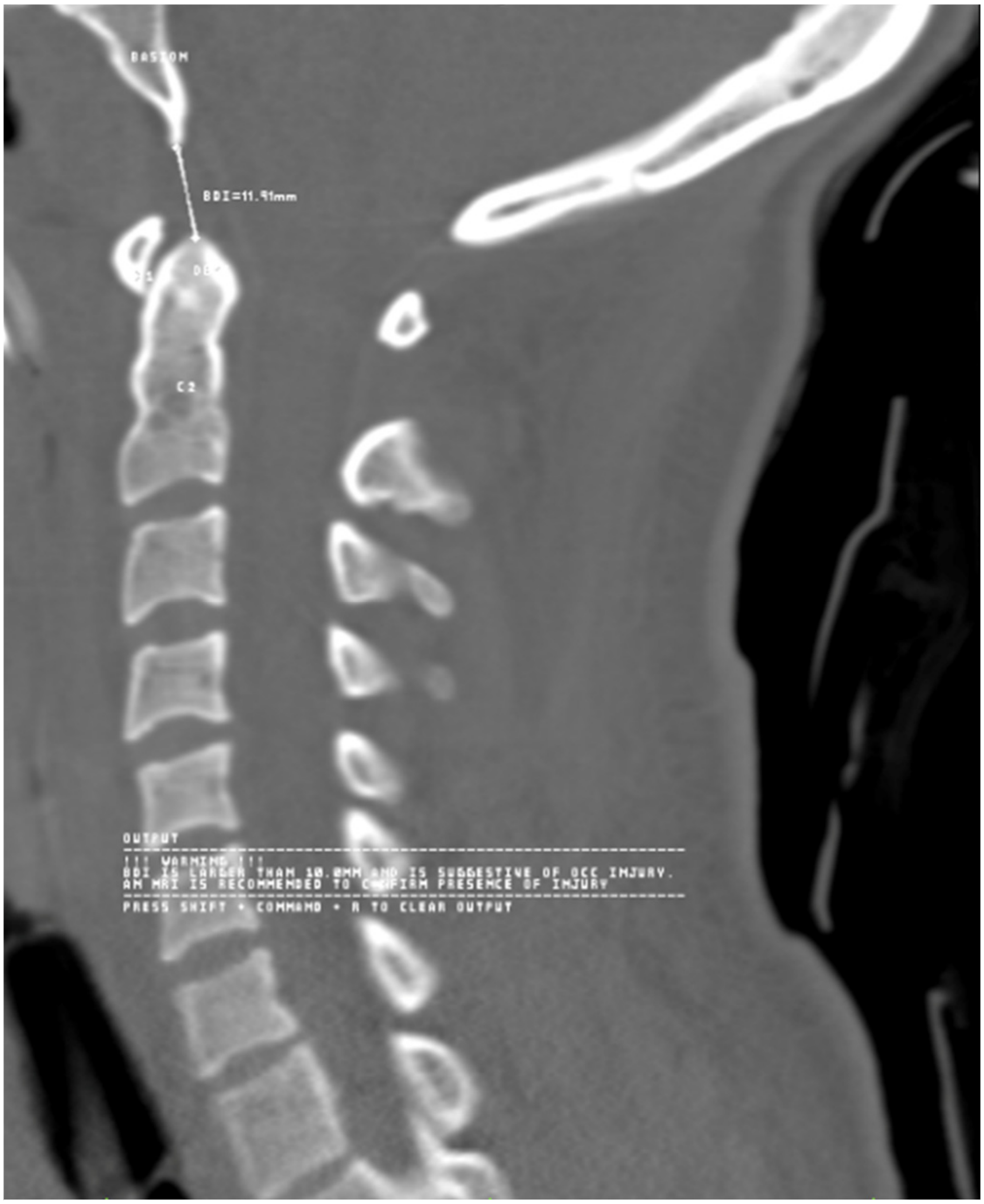
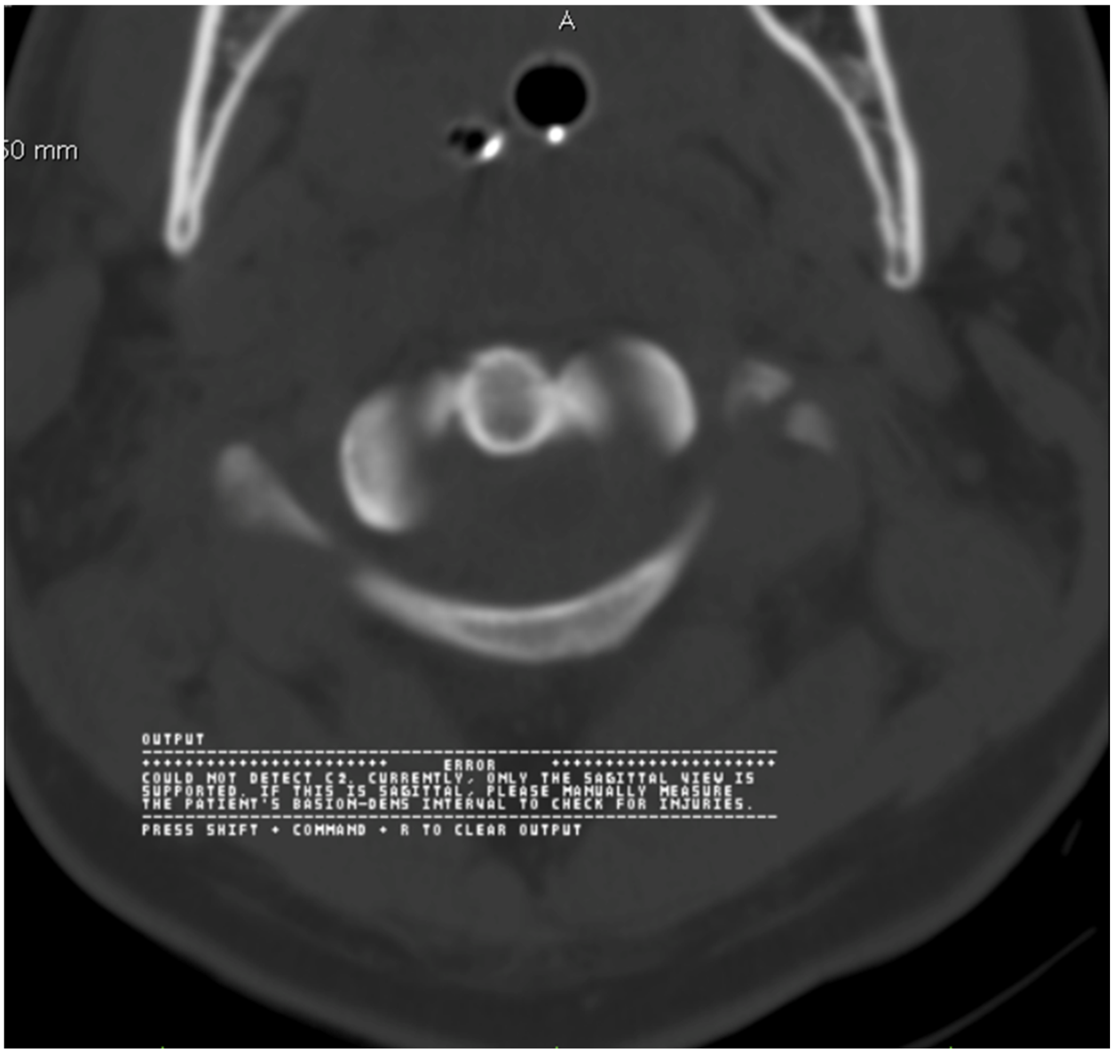


Figure 2.27 The original sagittal view of the cervical spine with the detected edges overlaid.



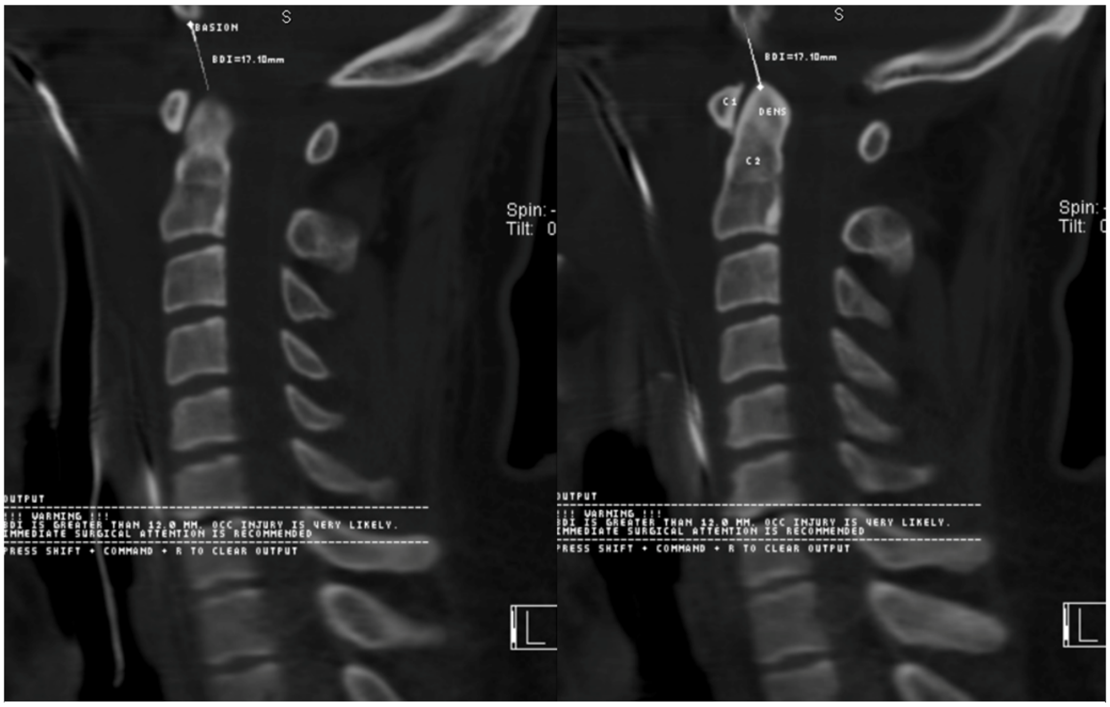
```
OUTPUT
------------------------------------------------------------
!!! WARNING !!!
BDI IS LARGER THAN 10.0MM AND IS SUGGESTIVE OF OCC INJURY.
AN MRI IS RECOMMENDED TO CONFIRM PRESENCE OF INJURY
------------------------------------------------------------
PRESS SHIFT + COMMAND + R TO CLEAR OUTPUT
```

Figure 2.28 An example of the output generated by the plugin. The dens and the basion were detected automatically in this instance. The line drawn between the two features is the BDI, measured at a deadly 11.91mm.



```
OUTPUT
-------------------------------------------------------------
++++++++++++++++++++++      ERROR      ++++++++++++++++++++++
COULD NOT DETECT C2. CURRENTLY, ONLY THE SAGITTAL VIEW IS
SUPPORTED. IF THIS IS SAGITTAL, PLEASE MANUALLY MEASURE
THE PATIENT'S BASION-DENS INTERVAL TO CHECK FOR INJURIES.
-------------------------------------------------------------
PRESS SHIFT + COMMAND + R TO CLEAR OUTPUT
```

Figure 2.29 Result of running the plugin on a series that is not the sagittal view of the neck. If the plugin cannot find C2, it assumes that it is an improper view.



```
OUTPUT
-----------------------------------------------------------
!!! WARNING !!!
BDI IS GREATER THAN 12.0 MM. OCC INJURY IS VERY LIKELY.
IMMEDIATE SURGICAL ATTENTION IS RECOMMENDED
-----------------------------------------------------------
PRESS SHIFT + COMMAND + R TO CLEAR OUTPUT
```

Figure 2.30 The plugin is capable of finding the basion and dens even when they are in separate images. Note that the length of the BDI does not include depth in its calculation, since in standard practice doctors do not take this into account in their measurements.

CHAPTER THREE

Results

The following table contains the measurements provided from Scott and White Hospital and those generated by the plugin. There are two columns for the algorithm, one for the automated detection of the dens and basion, and another for when the dens and basion were indicated by the user. Twenty-one images were successfully anonymized and evaluated by Dr. Chaput and the algorithm.

Performance of Automatic Dens and Basion Recognition

While the original plan for the plugin did not require automated detection of the dens and basion, throughout development it began to show promise. However, Table 3 reveals that the plugin could identify the bones successfully only seven times out of twenty-one. This includes instances where neither features were recognized, where the dens was recognized and the basion was not, and vice versa.

Performance of Assisted Dens and Basion Recognition

By using point-ROIs, the basion and dens could be identified by the user for the plugin. This supersedes the plugin's method of automatic detection, and explains its drastically improved performance. The plugin successfully made a measurement of the BDI all but three of the twenty-one test series. The numbers generated by this version of the algorithm were used to compute the average difference between the measurements from Scott and White and those generated by the plugin.

Table 3.1 Summary of measurements made at Scott and White Hospital and those made by the plugin. Red cells correspond to CT-series that were unsuccessfully measured by the respective algorithm. The plugin generated measurements that were on average 1.33mm longer than those made at Scott and White Hospital.

Patient ID	***Scott & White Measurement***	***Automatic Detection***	***Assisted Detection***	***Difference In Measurement (Assist - Scott)***
1010	4.60	🟥	4.34	-0.26
1011	5.98	🟥	5.78	-0.2
1016	3.79	🟥	5.10	1.31
1018	4.97	6.64	6.64	1.67
1022	6.52	8.44	8.44	1.92
1024	6.67	🟥	7.82	1.15
1026	5.35	🟥	6.26	0.91
1027	3.77	5.78	5.78	2.01
1028	5.37	🟥	5.86	0.49
1029	6.07	🟥	5.40	-0.67
1030	5.78	🟥	6.49	0.71
1031	5.75	🟥	🟥	
1032	5.75	7.42	7.42	1.67
1033	5.91	🟥	2.93	-2.98
1034	8.63	9.30	9.30	0.67
1039	7.07	🟥	11.00	3.93
1040	6.41	🟥	7.39	0.98
1041	5.87	5.03	5.03	-0.84
1042	7.57	9.18	9.18	2.01
1045	5.16	🟥	🟥	
1047	7.02	🟥	🟥	
			Average:	1.33
🟥 Failed to measure properly				
	All measurements are in MM			

The plugin managed to successfully produce a measurement of the BDI fully automatically seven times out of twenty-one. When the user indicated where C2 and the basion were located in the CT-series, the plugin produced a measurement successfully nineteen times out of the same sample of patients. On average, the plugin's measurements were 1.33mm different than Dr. Chaput's measurements. Despite this, the plugin only incorrectly predicted if a patient was injured once. The measurement for patient 1039 was 11mm, over the 10mm interval used to predict a ligamentous injury. Dr. Chaput's measurement of patient 1039's BDI is 7.07mm, signifying that the patient was not injured even though the plugin would say that they are. Out of the twenty-one measurements provided by Dr. Chaput, the plugin produced larger measurements all but in five instances.

CHAPTER FOUR

Discussion and Conclusion

Analysis of Results

Automatic detection of the basion and dens was tried, but was not critical to the project. The results clearly indicate that the automatic method was unsuccessful. The assisted method performed much better than the automated method. By allowing the user to select the basion and C2, much of the guesswork was taken out of the hands of the computer. However, the assisted method still failed to measure the BDI of three of the same series that the automated method struggled with. *Figure 4.1* depicts patient 1031, whose BDI measures in the normal range with a value of 5.75mm. The automated method failed to recognize C2 and the basion because it saw them as connected. The light area in between the two bones was mistaken for bone. The same problem prevented the assisted method from producing a valid measurement. Thus the difficulties encountered by both algorithms on these series were a result of the algorithms.

Possible Explanations for Mistaken Bone

One possible explanation for the algorithm mistaking flesh for bone is the fact that the local thresholding applied to the gradient filter does not have a minimum requirement for selecting a pixel. Even if the gradient at a particular pixel is 1, as long as it is greater than the surrounding gradient pixels, it will count. This was done so that even the faintest bone would not be excluded. However this reveals that doing so has caused more problems than it has prevented. The global thresholding applied initially was also designed to include faint bones, and so this problem could be rectified if either of them is adjusted to be stricter in their thresholding.

Another patient that both algorithms had difficulty measuring was patient 1045. This time the plugin mistook the faint object highlighted by the box in *Figure 4.2* as being part of the basion. Since the bottom of this object is lower than the bottom of the basion, the algorithm selected it as the basion and thus it measured the BDI to be extremely large. Again, the approach for fixing this would be to adjust the thresholding methods in order to prevent faint objects from registering.

The Measurements Were Longer Than the Official Ones

The plugin for the most part generated measurements that were longer than the ones made by Dr. Chaput. The reason is that what was found to be the tip of the dens and bottom of the basion must have been off from the points used at Scott and White. There are many possible factors explaining this. It is certain that it is not the thresholding methods that contribute to this, since they tend to add to the size of bone. This would make measurements smaller, as seen in *Figure 4.3*, where the bottom of the basion is estimated to be lower than where the actual bottom is.

Instead, the blame might be placed on the skeletonization, or edge-thinning method. It has a tendency to make the top edge of bone lower than where the human eye would perceive it. Observe in *Figure 4.4*, how the bone's top edge is beneath where one would perceive it to be.

After inspecting the code, it was found that the reason this happens is that the skeletonization process only scans the image from top to bottom. When the algorithm was first created, this was not seen as an issue. However in retrospect it is evident that this method of scanning, by removing pixels from the top of a region first and from the bottom of a region last, pushes the top edge of an object to the bottom of its edge region.

To fix this, the process would need to scan from not only top to bottom, but also bottom to top, left to right, and right to left. This would help keep the thinned-out edges towards the middle of their edge regions.

Future Developments and Relevant Research

This project is planned to continue development into a finished product that has the potential to be sold commercially. This section describes ways in which the plugin may be improved to make it into a stand-alone product.

Make the Plugin Independent of OsiriX

Though OsiriX possesses an impressive plugin API, it would be in the plugin's best interest to be completely independent of any particular DICOM viewer. This would make it available to a broader customer base, since OsiriX is only available to Macintosh users. In fact, the plugin was first written in Objective-C before being ported to C++ to support this need. It was also the motivation for creating a series of algorithms that are designed to detect edges without any set threshold, since '200' in OsiriX could be an entirely different intensity in another DICOM viewer.

Another possibility would be to make the plugin entirely independent of any DICOM viewer by giving it its own ability to open CT-series. A working prototype of this has already been made by Dr. Garner, taking advantage of the GLUT libraries of OpenGL.

Improving the Existing Algorithms

An obvious improvement that the plugin must go through is to address the issues experienced in its first trial run. The adaptive thresholding needs to be more exclusive of bone, and the skeletonization algorithm must be changed to make passes in all directions, rather than just scan in one direction. A larger set of trial images would help with tuning the algorithm's performance, since it was built on an initial test-bed of just 5 CT-series.

Biasi et al. at the University of Milan devised a method for characterizing plaque in the carotid artery by means of CT scan. The study investigated the use of CT scanning as a screening tool for patients suffering from lesions in the carotid artery (Biasi et al. 1998). This bears striking similarity to this thesis, in that this thesis applies image processing techniques to a CT series to screen patients for a specific injury. The study done by Biasi et al. focuses on deriving a specific metric for determining if a patient needed surgical attention while this thesis applies an already existing metric. In future research, it may be necessary to conduct a study that focuses on manipulating the parameters used in the algorithms and set up an image processing standard for this type of injury.

Improving the Automated Method

Adding versatility to the plugin would bring more value to it. Allowing it to make measurements in other parts of the body would make it a handy tool in other diagnostic situations that involve injuries that easily go unnoticed. It may be sufficient to have the user tell the plugin which region of the body that they are currently using, or possibly obtain this information from the DICOM viewer. However, if the plugin could store a 'dictionary' of shapes that correspond to bones in different parts of the body, it could still just rely on the user selecting a bone and handling it from there. Further research in applying a neural network to the algorithm would be interesting.

Soler et al. developed a fully automated segmentation algorithm of the liver and surrounding tissue for automated surgery (Soler et al. 2001). Instead of using a neural network, a vast series of assumptions are made about how the liver appears in CT series. Their algorithm works for approximately 85% of all livers, the other 15% being comprised of those that are oddly-shaped. It used more constraints and assumptions than this study. This thesis experienced a similar success rate and with more specific assumptions about CT scans (such as always setting the global threshold above a certain value), it could decrease its rate of failure.

Armato et al. developed a fully automated method for detecting lung nodules with a success rate of 70% (Armato et al. 2001). They implemented a comparative method for separating nodules from non-nodules, which was not done in this thesis. Perhaps most of their success owed to their database of 43 CT scans that provided their algorithms with ample reference material to make decisions upon. This study used only five series to build its definition of C2 and the basion. Having access to more images and implementing machine learning techniques could allow this algorithm to become much smarter in its approach to image segmentation and object recognition.

Conclusion

In conclusion, this project has proved that the assisted measurement of the BDI via a DICOM viewer plugin is an entirely possible venture. Though it is clear that the plugin is still in need of future development, such an investment would not be in vain. The successful trials of the plugin demonstrate that there is a future in applying computer vision to CT-scans, helping to prevent the occurrence of avoidable deaths associated with whiplash injury.

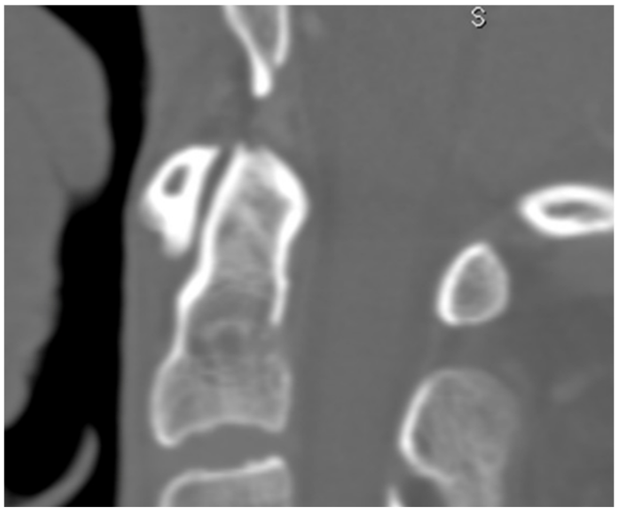


Figure 4.1 A close-up of the BDI of patient of 1031. Both the automated method and the assisted method failed to measure the BDI of this patient. The problem arises from the light gray area in-between the basion and the dens. The algorithms mistake this area for bone, and assume that the two features are connected. The solution to this would be to adjust either the global thresholding filter or the local thresholding filter applied to the gradient to be more exclusive of lighter bones.

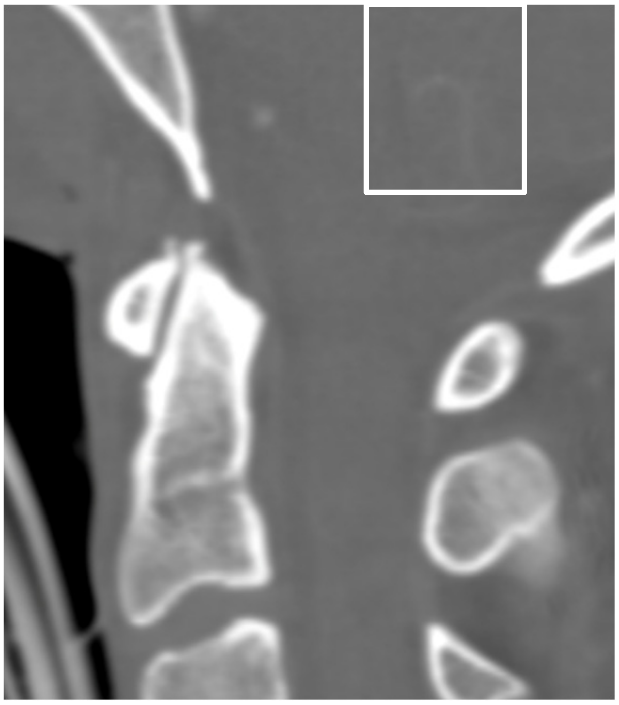


Figure 4.2 A close up of the BDI of patient 1045. The boxed area shows a faint object that the plugin mistook for part of the basion. The most likely reason the two objects are connected is due to the fact that both the local or global adaptive thresholding filters are designed to be as inclusive of bone as possible.

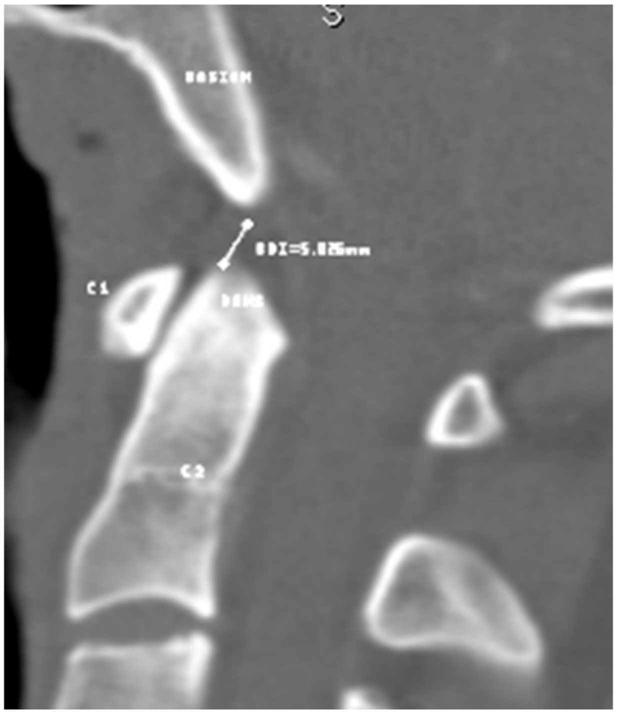


Figure 4.3 A close-up of the BDI of patient 1041. Note how the bottom of the basion was found to be much lower than where a human would detect it. This is due to the conservative nature of the adaptive thresholding algorithms.

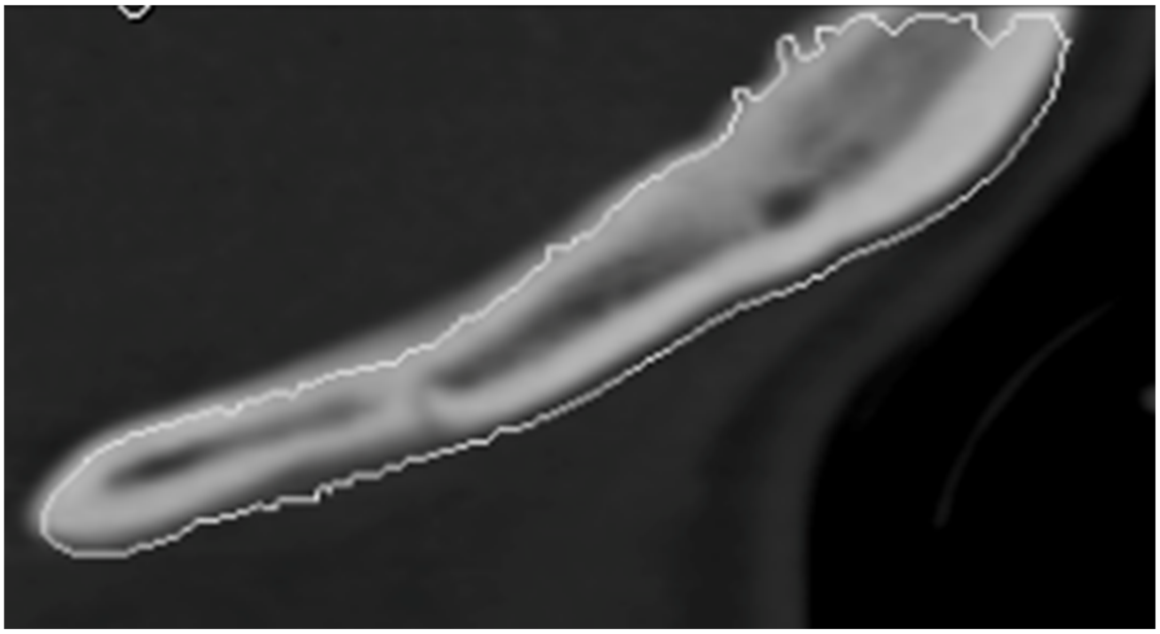


Figure 4.4 A close-up of a bone that has been passed through the edge detection methods. Notice how the detected top edge of the bone is far from where the human eye would perceive it to be. The bottom edge is also a bit off, but not to the extent the top is. The likely cause for this is that the skeletonization method only scan downwards. A proposed method would have the algorithm scan in the three other compass directions.

REFERENCES

Armato, S. G., Giger, M. L., & MacMahon, H. (2001). Automated detection of lung nodules in CT scans: Preliminary results. *Medical Physics*. 28: 1552.

Bertozzi, J., Rojas, C., & Matinez, C. (2008). Evaluation of the Pediatric Craniocervical Junction on MDCT. *American Journal of Roentgenology.* 192, 26-31.

Biasi, G. M. et al. (1998). Carotid Plaque Characterization Using Digital Image Processing and Its Potential in Future Studies of Carotid Endarterectomy and Angioplasty. *Journal of Endovascular Surgery*: August 1998, Vol. 5, No. 3, 240-246.

Changxian, S. (1998). Morphological thinning based on image's edges. *Communication Technology Proceedings 1998*, 5.

Chaput, C., Walgama, J., Torres, E., Dominguez, D., Hanson, J., Song, J., & Rahm, M. (2011). Defining and Detecting Missed Ligamentous Injuries of the Occipito-Cervical Complex. *Spine*, 9, 709-14.

Chaput, C., Walgama, J., Song, J., Hanson, J., & Rahm, M. (2011). Radiologic Criteria for Defining Atlantoaxial Dissociation (AAD) and Atlanto-Occipital Dissociation (AOD): The Reliability of CT Measurements and an Evaluation of the Effect of Age and Gender. *Spine*, 9, 158S-159S.

Clowes, M. B. & O'Gorman, F. (1973). Finding Picture Edges through Collinearity of Feature Points. *International Joint Conferences on Artificial Intelligence 1979*. 543-555.

Dominguez, D., Rahm, M., & Chaput, C. (2009) The Initial "Miss" Rate in Diagnosing Instability of the Occipitocervical Complex (OCC) at a Level 1 Trauma Center. *Spine*. 9, 11s-12S.

Gray, H. (1859). *Anatomy, Descriptive and Surgical Or, Gray's Anatomy*. Philadelphia: Blanchard and Lea. Image retrieved 11 October, 2011.

Law, W. K. & Chung, A. C. S. (2006). Segmentation of vessels using weighted local variances and an active contour model. *IEEE CVPRW 2006*, 189.

Soler, L. et al. (2001). Fully Automatic Anatomical, Pathological, and Functional Segmentation from CT Scans for Hepatic Surgery. *Computer Aided Surgery*. 6:131-132.