Curvature Detection and Visualization of Normal vs. Scoliotic Spine

Benno Steinegger[*]
TU Vienna

Jakob Peischl [†]
TU Vienna

ABSTRACT

Scoliosis, characterized by abnormal spinal curvature, is commonly assessed using the Cobb angle, a metric with limitations in capturing the full three-dimensional deformities of the spine. Our methodology integrates the MeVisLab platform and Python scripting using the VTK library to create an interactive system for spinal curvature analysis. Users annotate critical vertebrae on 3D-rendered spines to compute the Cobb angle and classify scoliosis severity. Additionally, we provide comparative 3D visualizations of normal and scoliotic spines.

1 INTRODUCTION

Scoliosis is an orthopaedic condition characterized by abnormal curvature of the spine. Although the term sometimes describes general three-dimensional (3D) spinal deformities [9], it more commonly refers to only the curvature in the frontal plane of the body. This planar curvature is commonly assessed using the Cobb angle, which has become the standard metric for this purpose [4]. Therefore, a precise measurement of the Cobb angle is essential to classify the severity of scoliosis and guide treatment decisions.

Since spinal deformities are a 3D problem, the Cobb angle can, despite its widespread use, fail to accurately capture the spinal condition of a patient. In the literature, numerous proposals have been made with the aim of improving these shortcomings with different quantitative measures. However, a more practicable metric than the Cobb angle has not been brought forward yet. For this reason, it is useful to avoid reducing complicated curvature into a single metric and instead create meaningful visualizations that help physicians assess cases of scoliosis. Furthermore, it might be of clinical and educational interest to compare different stages of scoliosis progression, prompting the question of how to create effective comparative visualizations.

In this work, our aim is to provide clinically applicable contributions to both sides of the problem by a) developing a user-guided system for quickly and easily estimating the Cobb angle and by b) creating informative visualizations for assessing the general state of spinal deformities as well as comparing differently affected spines with each other.

We integrate the MeVisLab platform with Python scripting and the VTK (Visualization Toolkit) library to develop a semi-automated system for spinal curvature analysis. The program processes medical image data to segment and visualize the spines in 3D. Users interact with the system by identifying two critical vertebrae through line annotations. These user-defined lines provide the basis for computing the Cobb angle, which is then used to classify the spine as normal or scoliotic.

2 METHODS

Our analysis involved two main tasks: semi-automatic calculation of Cobb angles to quantify spinal curvature and visualization of normal and scoliotic spines for comparative analysis. These tasks were implemented using a combination of segmentation techniques, mathematical modelling, and 3D rendering tools.

2.1 Calculation of the Cobb angle

The Cobb angle [1], a standard metric for quantifying spinal curvature, is calculated using the following approach.

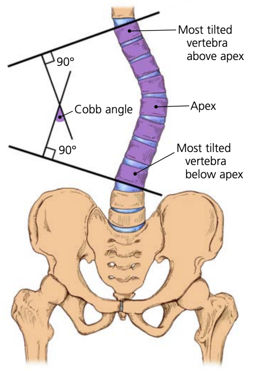

Most tilted vertebra above apex
90°
Cobb angle
Apex
90°
Most tilted vertebra below apex

Figure 1: A comprehensive illustration of the Cobb angle calculation. Taken from Greiner et. al. [2]

First, the most tilted vertebrae at the superior and inferior ends of the curve are identified in the image. For these vertebrae, lines are drawn along their endplates. These vectors are denoted as v_{top} and v_{bottom}, corresponding to the upper endplate of the superior vertebra and the lower endplate of the inferior vertebra, respectively. To calculate the Cobb angle in three dimensions, we used the start and end points of lines representing the orientation of the superior and inferior vertebrae. The procedure is as follows:

1. **Extract Line Vectors**: For each vertebral line, the start and end points, $\mathbf{p}_{\text{start}}$ and $\mathbf{p}_{\text{end}}$, were extracted from the 3D data. The directional vector for each line was calculated as:
$$\mathbf{d} = \mathbf{p}_{\text{end}} - \mathbf{p}_{\text{start}}. \tag{1}$$
2. **Calculate orthogonal vectors**: As shown in Fig. 1, we need to compute the angle between the orthogonal vectors of the directional vectors. For this calculation, we are using the right-hand rule. In addition, we also set the z coordinate to a constant λ, because it is not relevant for the Cobb angle. Let $\lambda = 1$, and let d_1 and d_2 represent the x- and y-coordinates, respectively.
$$\mathbf{v} = [d_2, d_1 * -1, \lambda] \tag{2}$$
This step is not strictly necessary, otherwise, you can just extend the directional vectors.
3. **Angle Calculation**: The angle between two orthogonal vectors $\mathbf{v}_{\text{top}}$ and $\mathbf{v}_{\text{bottom}}$, representing the superior and inferior vertebral lines, was computed using the cross and dot products of the vectors:
$$\theta = \arctan 2\left(\frac{\|\mathbf{v}_{\text{top}} \times \mathbf{v}_{\text{bottom}}\|}{\mathbf{v}_{\text{top}} \cdot \mathbf{v}_{\text{bottom}}}\right), \tag{3}$$
where:

[*]e-mail: e12117772@student.tuwien.ac.at
[†]e-mail: e12123459@student.tuwien.ac.at

- $\mathbf{v}_{\text{top}} \cdot \mathbf{v}_{\text{bottom}}$ is the dot product
- $\mathbf{v}_{\text{top}} \times \mathbf{v}_{\text{bottom}}$ is the cross product

4. **Conversion to Degrees**: The angle θ in radians was converted to degrees:

$$\text{Cobb Angle} = \theta \cdot \frac{180}{\pi}. \tag{4}$$

After the Cobb angle computation, we used the following clinically accepted classification table [6]:

Cobb Angle Range (Degrees)	Classification
≤ 10	Normal spine
$10 < \text{Angle} \leq 20$	Mild scoliosis
$20 < \text{Angle} \leq 40$	Moderate scoliosis
> 40	Severe scoliosis

Table 1: Classification of scoliosis based on Cobb angle.

A similar approach is presented by Tanure et al. [8]. Generally, a good overview of the possible approaches is provided by Safari et al. [7]. More sophisticated and more recent strategies involve machine learning or more explicit convolutional neural networks [3].

2.2 Visualization of physiological vs. scoliotic spines

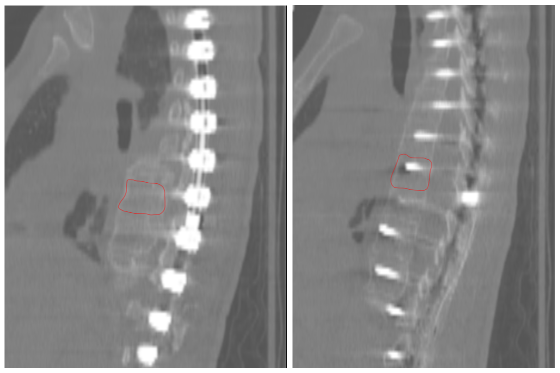
The visualization of the physiological and scoliotic spines was performed to allow a direct and interactive comparison of their anatomical structures and curvatures. This process involved segmentation of individual vertebrae, 3D rendering, and some advanced visualization techniques.

2.2.1 Segmentation of Individual Vertebrae

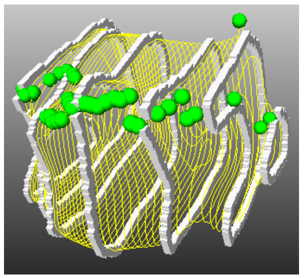
The individual vertebrae were segmented manually using a mixed approach that combined livewire segmentation and manual contouring while using linear contour interpolation for both as shown in Fig. 2. Although labor intensive, this approach was necessary because traditional and more automated segmentation methods, such as region growing, iso-surfaces, and watershed algorithms, failed to segment vertebrae accurately. This is because they do not make prior assumptions about the shape or structure of the spine and individual vertebrae. Only machine learning models such as the one proposed by Klein et al. [5] are powerful enough to detect the patterns in the data that are required to successfully separate individual vertebrae. However, we did not have the necessary hardware resources to utilize such a model. Therefore, we performed the segmentation process using a more manual approach. This was essential to achieve sufficiently accurate results because of the complexity of the spine's anatomy. In particular, for scoliotic cases as well as for images affected by artefacts, such as postoperative CT images that contain artefacts due to metallic surgical devices, this was the only viable approach based not on deep learning.

Generally, vertebral segmentation was done with a focus on accurately delineating the vertebral bodies rather than their processes. This is because we found that the vertebral bodies and their endplates are significantly more effective in communicating the curvature of the spine. To capture the orientation of the endplates as precisely as possible, the contour drawing was done after a sagittal reformat of the base image. This way, were in full control over the inclination of each vertebra and how it is reflected in the resulting interpolations and vertebra surfaces.

3D Rendering and Comparison The segmented vertebrae were rendered in 3D to visualize the curvature and alignment of the spine. The visualization juxtaposed the normal (physiological) spine, the scoliotic spine, and a postoperative corrected spine side by



(a) Medial slice (b) Slightly lateral slice



(c) Segmentation contours of a vertebra

Figure 2: (a)-(b): Interface for manual contouring and livewire-based contouring for T8 (eighth thoracic vertebra). Contours are drawn with a thin red marker. A sagittal view is used. (c): Contours of a segmented vertebra with manual contours drawn in white and interpolated contours visualized in yellow.

side, allowing a direct comparison of their anatomical deformities and assessing the effectiveness of the surgery. The deformations in the scoliotic spine, including lateral displacement, rotation, and curvature, were clearly highlighted through this comparative rendering. Additionally, we used color-coding of individual vertebrae to make a connection to the Cobb angle measurements. Specifically, we highlighted the cranial and caudal most tilted vertebrae that provide the direction vectors for the Cobb angle as well as the apex vector which has the most lateral displacement. The resulting segmentation is shown in the next section.

Interactive Visualization with Contextual Information To maintain anatomical context during visualization, a 2D maximum intensity projection (MIP) of the original CT volume was displayed behind the 3D-rendered spine. This projection provided a view of the structures surrounding the spine, such as the ribs and soft tissues, to ensure the spine was visualized in its broader anatomical setting. However, the background anatomical context was chosen to be 2D to not distract from the curvature of the 3D spine. The MIP dynamically adapted to the position of the camera, continuously showing the volume projection in the current view direction. This interactive feature allowed users to rotate, zoom, and explore the spines while retaining the spatial context of the surrounding anatomy. The idea was based on visualizations by Illés and Somoskeöy [4].

3 Implementation

For the implementation of the semi-automatic calculation of the Cobb angle as well as for the comparative visualization of spines, we used MeVisLab together with its scripting API that combines its own module definition language and Python scripts.

3.1 Determining the Cobb angle

3.1.1 Automated Segmentation of the Spine

Segmenting the spine in CT images is a challenging task, particularly when trying to identify an approach that fits well without making assumptions about the data or image characteristics. We explored several methods, as discussed in Sect. 2.2.1, but none fully satisfied our needs. To address these limitations, we opted for a combination of a ROISelect macro for 3D model generation for our semi-automated Cobb angle computation. ROISelect empowers the user to define the region of interest in the image, focusing on the specific area they want to segment. After this, the user can refine the minimum iso value used for 3D model generation. This adjustment can be performed dynamically, with a slider allowing real-time modifications while viewing the evolving model. We recognized that segmenting the spine as a whole instead of its individual vertebrae is sufficient to determine the Cobb angle when guided by a human operator. Although individual vertebrae would not be segmented correctly using isosurfacing, this approach usually generates a segmentation where junctions between the vertebra are easily detectable by the human vision system. This makes our approach especially efficient and viable for clinical use.

3.1.2 Calculating Cobb angle

The next step in our process involves detecting the spinal curvature and calculating the Cobb angle. Users are provided with an interactive live view, allowing them to draw lines directly on the 3D surface using the `SoCSODrawOnSurface` tool. These user-drawn lines are stored in a custom macro that contains a CSOList and is linked to a custom Python script. The script continuously monitors updates to the CSOList, checking whether two lines have been defined. Once two lines are detected, their start and end points are extracted as 3D vectors. Using this information, the Cobb angle is calculated as detailed in Sect. 2. We are aware of the drawbacks and limitations of inputting 2D data using a 3D interface.

To enhance usability, the system visually extends the two drawn lines and displays their intersection, providing immediate feedback on the chosen geometry. In addition, the calculated Cobb angle, along with its corresponding classification, is shown in the upper left corner of the interface. This approach ensures that the user can easily evaluate the accuracy of their input and understand the clinical significance of the detected curvature in real-time.

To provide better context for the spine, we aimed to embed it within its surroundings and display the adjacent bones as well. To avoid duplicating the spine (once in red from the segmented image and again as part of the whole 3D model), we employed a Boolean operator to visualize only the difference between the two 3D objects. However, this approach proved to be computationally demanding, resulting in significant lag during live visualization on a MacBook M1 Pro, rendering it impractical for real-time use. It is important to note that even reducing the resolution by a factor of 3 did not sufficiently improve performance.

3.2 Comparative Visualization

Unlike for the Cobb angle determination, the interactive comparative visualization of physiological and scoliotic spines required a segmentation of individual vertebrae. This not only allowed us to create a more visually pleasing rendering of the investigated spines but also enabled us to selectively highlight individual aspects of the spine.

As mentioned previously, segmentation was done using a combination of manually drawn spline-based contours and livewire-based contours with a sagittal view on the CT image. With MeVisLab, this was realized with the help of `CSO` related macros like `CSOManger`, `CSOSliceInterpolator`, `SoCSOLiveWireEditor` and `SoCSOVisualizationSettings`. Generally, we encountered a lot of technical problems during the segmentation which was especially true for live-wire-based contours and somewhat mitigated for spline-based contours. Please, see the corresponding `.mlab` file for more details about this.

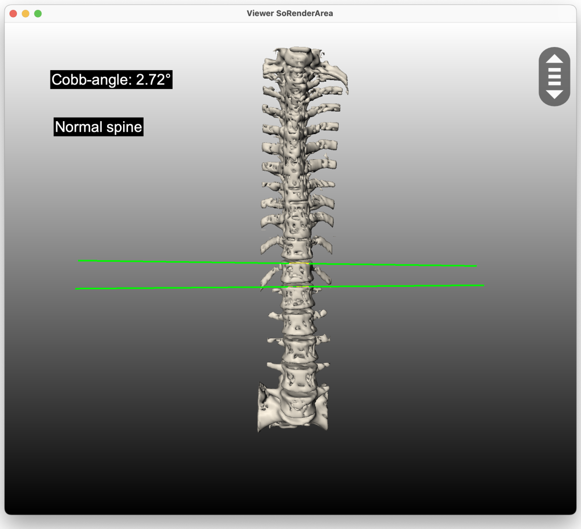


Figure 3: In this image you can see an example with a healthy spine. Note the text in the upper left corner and the iso slider in the upper right corner. Lastly, the two green lines are the extended lines, which are used to calculate the Cobb Angle.

The 3D rendering and the interactivity was implemented with the help of various Open Inventor modules such as `SoWEMRenderer`, `SoRenderArea` and `SoCameraInteraction`. The visualizations were juxtaposed using `SoViewportRegion` modules. In order to generate WEM data for individual vertebrae, we employed a custom reusable macro called `VertebraeToSurfaces` which is capable of providing separate vertebrae in its output and which was used to apply different rendering properties to individual vertebrae.

The 2D MIP was performed using a `itkMaximumProjectionImageFilter` and a number of `SoTransform` modules as well as `SoCalculator` modules that ensure that the 2D image remains in the correct position during camera movement.

4 Conclusion

This project presented an interactive Cobb angle calculator and a static visualization of pre- and postoperative spinal conditions, offering a medical visualization tool for understanding scoliosis diagnosis and treatment outcomes. While our tools try to provide valuable insights, our approach has significant limitations. A key challenge was the difficulty in segmenting the spine and individual vertebrae from medical images, which required manual input and reduced automation potential.

A Work split

We split the project into two parts: curvature detection and comparative visualization. Benno Steinegger performed the curvature detection, while Jakob Peischl was responsible for the visualization part. The same split also applies to this report.

B Code & Disclaimer

In order to run and test our networks, a MeVisLab Evaluation License is required because our custom Python code exceeds 2 KB in

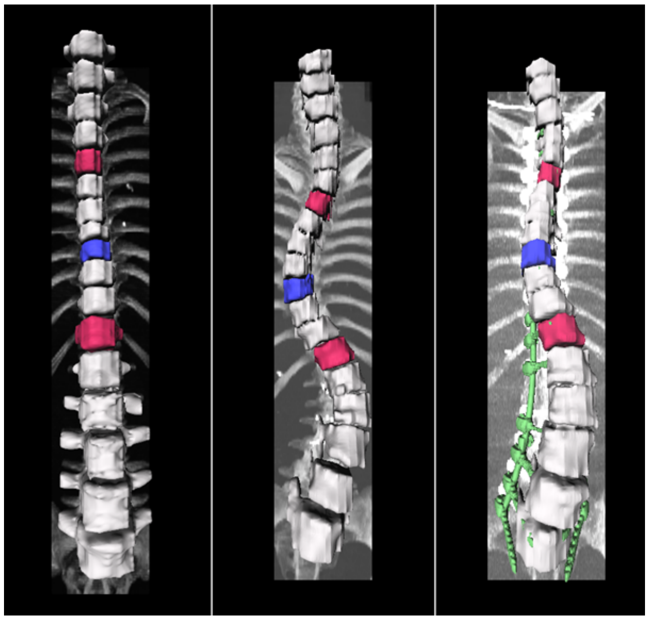


Figure 4: Comparative visualization of the physiological spine, presurgical scoliotic spine, and postsurgical spine. The figure shows an anterior view. The apical vertebra is highlighted in blue while the two most tilted vertebrae relevant for the Cobb angle computation are highlighted in red. For the postsurgical spine, the pedical screws are shwon in green. The background shows the corresponding MIPs in the anterior direction for each investigated volume.

size. This license can be obtained for academic purposes easily by contacting MeVisLab.

Our work can be found in this GitHub repository in addition to the uploaded *zip* file on TUWEL.

References

[1] J. Cobb. Outline for study of scoliosis. *Instructional Course Lectures. Ann Arbor*, 1948.

[2] K. A. Greiner. Adolescent idiopathic scoliosis: radiologic decision-making. *American family physician*, 65(9):1817–1823, 2002.

[3] M.-H. Horng, C.-P. Kuok, M.-J. Fu, C.-J. Lin, and Y.-N. Sun. Cobb angle measurement of spine from x-ray images using convolutional neural network. *Computational and Mathematical Methods in Medicine*, 2019:1–18, Feb. 2019. doi: 10.1155/2019/6357171

[4] T. Illés and S. Somoskeöy. Comparison of scoliosis measurements based on three-dimensional vertebra vectors and conventional two-dimensional measurements: advantages in evaluation of prognosis and surgical results. *European Spine Journal*, 22(6):1255–1263, Jan. 2013. doi: 10.1007/s00586-012-2651-y

[5] G. Klein, M. Hardisty, C. Whyne, and A. L. Martel. Vertdetect: Fully end-to-end 3d vertebral instance segmentation model, 2023.

[6] R. R. Maaliw, J. A. B. Susa, A. S. Alon, A. C. Lagman, S. C. Ambat, M. B. Garcia, K. C. Piad, and M. C. Fernando Raguro. A deep learning approach for automatic scoliosis cobb angle identification. In *2022 IEEE World AI IoT Congress (AIIoT)*, pp. 111–117, 2022. doi: 10.1109/AIIoT54504.2022.9817290

[7] A. Safari, H. Parsaei, A. Zamani, and B. Pourabbas. A semi-automatic algorithm for estimating cobb angle. *Journal of biomedical physics & engineering*, 9(3):317, 2019.

[8] M. C. Tanure, A. P. Pinheiro, and A. S. Oliveira. Reliability assessment of cobb angle measurements using manual and digital methods. *The Spine Journal*, 10(9):769–774, Sept. 2010. doi: 10.1016/j.spinee.2010.02.020

[9] A. Thalengala, S. Bhat, and A. Hoblidar. Computerized image understanding system for reliable estimation of spinal curvature in idiopathic scoliosis. *Scientific Reports*, 11, 03 2021. doi: 10.1038/s41598-021-86436-3