Semi-Automatic Generation of Preoperative Surgical Plans for Forearm Deformity Correction

by

L.F. Vergeer

TU Delft

LUMC Leids Universitair Medisch Centrum

Erasmus MC

Semi-Automatic Generation of Preoperative Surgical Plans for Forearm Deformity Correction

by

L.F.Vergeer

in partial fulfilment of the requirements
to obtain the joint degree of Master of Science in
Technical Medicine
at Delft University of Technology, Leiden University Medical Center, Erasmus University Roterdam

Student number:	4643542	
Project duration:	July, 2024 – November, 2025	
Thesis committee:	Dr. B.L. (Bart) Kaptein,	LUMC (Chair)
	Dr. J.G. (Jasper) Gerbers,	LUMC
	E.M. (Eline) van Es, MSc,	EMC
	Dr. J.W. (Joost) Colaris,	EMC
	Dr. R.P.J. (Roy) van den Ende,	LUMC (independent assessor)

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

Cover Image: 3D Picture of the Ulna and Radius.

Acknowledgements

As I complete my studies with this master's thesis, I reflect on the path that brought me here. With a broad range of interests, many elements came together in the program of Technical Medicine, a study I've truly enjoyed. Early on, I became particularly interested in the 'Imaging and Intervention' track. Since then, my enthusiasm for imaging, image processing, and computer-assisted surgery has only grown. During my clinical internships at various hospitals, I thoroughly enjoyed being involved in the clinic and developed a strong fascination with surgery and the innovations coming from the 3D lab. I'm very grateful that I also had the opportunity to do my graduation internship in this field. I don't know exactly what the future holds, but so far, I've learned that with the right motivation and dedication, I can go a long way.

I would like to thank my parents for always encouraging me to do my best, while reminding me that you can't do more than that. Moreover, I am very grateful to my childhood friends, who have always been a steady foundation to rely on. During my studies, my student association became my second home, where my year club, guild, and house made my student years truly unforgettable. I truly appreciate my close friends, with whom I can always be myself. A special thanks goes to my boyfriend, who kept everything running when I couldn't and always does his best to make me happy.

My internist has inspired me to go the extra mile for patients, and I also want to thank the specialists who have helped me better understand myself. I am grateful to my supervisors, who not only guided me academically but also gave me the freedom to focus on what truly matters in life, such as family and health. Over the past year, I have learned a lot about myself and gained valuable insights into how I want to shape my future. I look back on this time with appreciation and look forward to the next chapter.

Lishia Vergeer
Delft, November 2025

Preface

The goal of this thesis is to develop a method that supports the planning of corrective surgery on deformed forearm bones. My motivation arose from an interest in the innovative potential of 3D imaging and 3D printing, as well as the opportunity to contribute to improving the accuracy and efficiency of surgical procedures. Before starting this project, I conducted a literature study on the design of 3D-printed patient-specific tools used to guide osteotomies, which are surgical cuts in the bone, to aid in accurately repositioning bone fragments and placing fixation plates. The purpose of this thesis concerns the planning of where the bone should be cut and how it should be repositioned. This step of surgical planning is often time-consuming, and a clear need was expressed for the optimisation of this planning process.

Previous research conducted at the Erasmus University Medical Center (EMC) confirmed that a semi-automated approach is a promising direction. The Leiden University Medical Center (LUMC) successfully performed a proof-of-concept study for semi-automating femoral osteotomies using an open source framework for multi-objective optimisation. These parties joined efforts to further develop and extend this line of research.

This collaboration gave me the opportunity to enhance my programming skills, to explore how to design software for broader clinical application, and to tailor the software to the needs of the user. I leveraged the project to meet my personal learning goals while ensuring that the forearm planning tool is suitable for further development.

To accommodate the interests of the diverse readership of this thesis, the content has been divided into five parts.

- Part I: Validation Paper
  Presents the clinical and scientific context, and provides insight into the method's feasibility and performance.

- Part II: Step-by-step Guide
  Contains installation instructions, usage guidelines, and troubleshooting tips for clinical users.

- Part III: Technical Document
  Outlines the development process, theoretical considerations of alternatives, and key technical aspects of the semi-automated forearm osteotomy planning tool.

- Part IV: Code Documentation
  Intended for developers, this part details the module structure, data flows, and the main classes and functions.

- Part V: Conclusion and Future Work
  Summarizes the main findings and provides recommendations for future research, including a reflection on the project overall.

Summary

Introduction

Corrective forearm osteotomies are complex procedures aimed at restoring anatomical alignment and function in patients with deformities caused by trauma or congenital conditions. While computer-assisted planning improves precision, existing workflows are labour-intensive and limit exploration of optimal solutions. This thesis presents the 3D-SurgiGen Add-on, a semi-automated, open-source planning tool designed to enhance efficiency, consistency, and clinical applicability.

Methods

The 3D-SurgiGen Add-on was developed as a modular plugin for Blender using open-source technologies and a multi-objective optimization framework. Development involved defining requirements, analyzing and restructuring an existing codebase, designing a new architecture, incremental integration, and iterative refinement based on testing and clinical feedback. The tool enables import of 3D forearm models, annotation of anatomical landmarks, and planning based on the mirrored contralateral bone. It supports both opening and closing osteotomies via an innovative optimisation strategy that enables multiple osteotomies while providing explicit control over distal alignment, critical for joint congruency. A key innovation is the user-in-the-loop workflow, which enables clinicians to interactively explore osteotomy alignment options and select a preferred solution from multiple optimised alternatives. The performance and usability of the tool were subsequently evaluated to assess its potential as a clinically relevant semi-automated planning solution.

Results

The tool was prospectively evaluated in five clinical cases involving deformities of both the radius and ulna. Semi-automated planning yielded a comparable number of clinically acceptable plans to manual planning, with similar failure rates and no reduction in planning time (median difference: 7 minutes, range: 4–17). Limitations include the small sample size and the exclusive evaluation of single-shaft osteotomies. Usability feedback further highlighted current limitations in visualization, parameter interaction, and overall robustness of the Add-on.

Conclusion

The 3D-SurgiGen Add-on demonstrates the feasibility of semi-automated, optimisation-based forearm osteotomy planning. Future work should expand validation, refine clinical objectives, and improve usability. With continued refinement, the 3D-SurgiGen Add-on could serve as an efficient complementary tool for clinical decision-making, providing insight into surgical options and enabling the customisation of preoperative corrective osteotomy plans efficiently.

Table of Contents

List of Definitions

Radius and Ulna:
The two bones of the forearm.

Correction osteotomy:
In this thesis also referred to as simply "osteotomy"; one or two bone cuts used to correct anatomical deformities.

Blender:
An open-source 3D modeling software used for visualization and interaction with anatomical models.

Python:
The primary programming language used for developing and integrating various components in this project.

Pymoo:
A Python-based multi-objective optimization library used to implement and customize the optimization framework.

User interface:
The interactive components in Blender, including custom input fields, panels, and buttons used to guide the planning process.

Operator:
A custom Blender function (typically bound to a button) that triggers a defined action or sequence, such as starting the optimisation.

Optimisation algorithm:
A computational method used to search for optimal solutions based on defined objectives and constraints.

Parameters:
Variables adjusted during optimization, such as cut position or rotation angles.

Objectives:
Criteria the optimisation algorithm seeks to minimize or maximize, such as alignment accuracy.

Constraints:
Conditions that restrict the solution space, ensuring feasibility in anatomical or surgical terms.

Bounds:
Predefined limits for parameters used in the optimization process.

Interpreters:
Components that translate user-defined or anatomical input into computational parameters used by the optimisation.

Property:
A value or attribute associated with a class or Blender object.

Module:
A file or logical unit of code in Python that groups related functions and classes.

Classes:
Structures used to define and organize related properties and methods in the code.

Functions:
Individual reusable blocks of code that perform a specific task.

List of Abbreviations

LUMC	Leiden University Medical Center
EMC	Erasmus University Medical Center
CORA	Center of Rotation of Angulation
CT	Computed Tomography
STL	Standard Triangle Language
PSI	Patient-specific Instruments
ID	Identification
UI	User Interface
PRUJ	Proximal RadioUlnar Joint
DRUJ	Distal RadioUlnar Joint
CAS	Computer-Assisted Surgery
3D	Three-Dimensional
SSM	Statistical Shape Model
STC	Standardization and Terminology Committee
JCS	Joint Coordinate System
SVD	Singular Value Decomposition
ICP	Iterative Closest Point
PCA	Principal Component Analysis
EAs	Evolutionary Algorithms
SQP	Sequential Quadratic Programming
UX	Uniform Crossover
HUX	Half Uniform Crossover
XML	Extensible Markup Language
JSON	JavaScript Object Notation
IDE	Integrated Development Environment
API	Application Programming Interface
ROI	Region of Interest
CAD	Computer-Aided Design

List of Figures

List of Tables

Part I:
Validation Paper

This chapter presents the validation paper that forms an intergral part of this thesis. The paper describes the background, methods, results, and discussion of the semi-automated forearm osteotomy planning approach, providing insights into its feasibility and performance.

Semi-Automatic Generation of Preoperative Surgical Plans for Forearm Deformity Correction

Lishia F. Vergeer[a,b], Bart L. Kaptein[a], Jasper G. Gerbers[a], Eline M. van Es[b], Joost W. Colaris[b]

[a] *Department of orthopaedics, Leiden University Medical Center, Leiden, The Netherlands*
[b] *Department of orthopaedics, Erasmus University Medical Center, Rotterdam, The Netherlands*

Abstract

Introduction Forearm deformities can impair function and may require corrective osteotomies to restore anatomical alignment. Computer-assisted planning enables three-dimensional (3D) assessment of the deformity, osteotomy planning, and design of patient-specific surgical guides, typically using the mirrored contralateral bone as a reference. However, current workflows remain labour-intensive, limiting the exploration of alternative osteotomy options. This study introduces and evaluates the 3D-SurgiGen Add-on, a semi-automated, interactive framework for preoperative planning of forearm osteotomies, aimed at enhancing efficiency, accuracy, and clinical flexibility in surgical planning.

Methods The 3D-SurgiGen Add-on, developed as a plugin for the open-source Blender platform, provides a semi-automated framework for preoperative planning of forearm osteotomies using 3D bone models. Through its interactive environment, it enables real-time exploration of osteotomy strategies, reference bone scaling, and adjustment of alignment constraints. A multi-objective optimisation algorithm subsequently minimises malalignment and surface discontinuities to facilitate plate positioning. By presenting a set of solutions representing trade-offs between clinical objectives, the system supports integration of patient- and surgeon-specific preferences. Qualitative validation was performed on five cases requiring corrective osteotomies of both the radius and ulna. Each case was planned in randomised order by an experienced biomedical planner using both 3D-SurgiGen and conventional manual planning in 3-Matic. Two independent orthopaedic surgeons evaluated joint alignment, osteotomy location and orientation, wedge size, bone contact, osteotomy alignment, and overall bone shape, documenting any failures or weaknesses. Usability feedback was collected from four participants following a planning task, using a structured questionnaire to assess clinical feasibility.

Results Both planning methods produced an equal number of accepted plans, with three failures each across ten bones, including two failures on the same bone. The reasons for plan rejection and perceived weaknesses, primarily related to overall bone shape and osteotomy alignment, were identical for both methods. The semi-automated workflow for planning both the ulna and radius did not reduce planning time compared to manual planning (30 min, range 23–40 vs 42 min, range 34–47), with a median time difference of 7 min (range 4–17).

Conclusion The 3D-SurgiGen Add-on produced the same number of valid osteotomy plans as manual methods, while enabling interactive exploration of surgical options. In this study, it did not reduce planning time and was evaluated on a small sample, limiting generalizability. The system allows integration of clinical expertise into the optimization process, and further development should focus on refining bone shape metrics, improving usability, and validating the approach across a broader range of deformities. With these improvements, the Add-on has the potential to support more efficient and flexible planning in forearm surgery.

Keywords: preoperative planning, forearm, osteotomy, computer-assisted surgery, optimisation, automation

1. Introduction

Deformities of the forearm bones can lead to substantial impairments in performing everyday tasks. Recent innovations in surgical planning have provided more optimised treatment options to correct forearm deformities.(16) However, further advances are desired to allow a more efficient exploration of surgical options.

Deformities of the forearm bones can arise from trauma, congenital malformations, or bone disorders. A common cause is malunion, in which a fractured bone does not regain its anatomical alignment after healing.(11) (12) Forearm fractures are particularly frequent in children and are typically managed non-operatively. Although many deformities remodel during growth or remain clinically subclinical, a subset of patients develop persistent symptomatic malunion that requires surgical intervention.(2)

Accurate preoperative planning for forearm deformity correction is of high value, as the radius and ulna, located on the side of the thumb and on the side of the little finger, respectively, interact closely. The rotation of the radius around the ulna enables supination and pronation, turning the palm of the hand upwards and downwards, respectively.(10) This rotational movement is facilitated by the complex shape of the radius', including a lateral curve known as the radial bow.(4) As illustrated in Figure 1, the radius and ulna articulate proximally at the radioulnar joint (PRUJ) near the elbow and distally at the radioulnar joint (DRUJ) near the wrist.(10) The bones are connected by the interosseous membrane, which provides stability, transfers axial loads, and maintains appropriate spacing between the radius and ulna during rotation.(18)

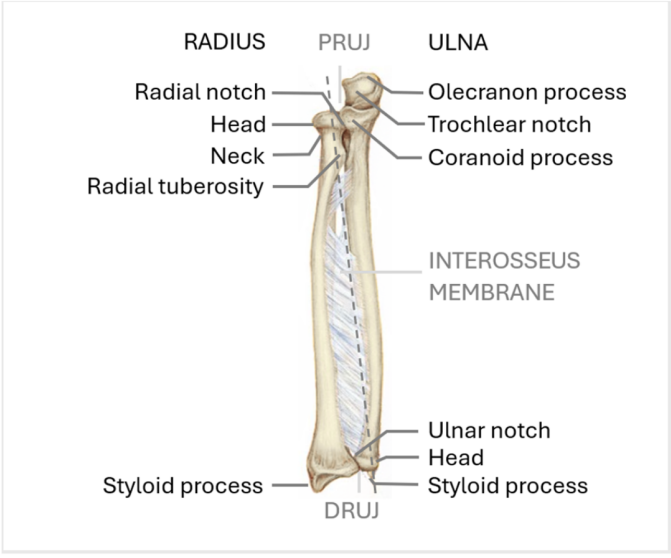


Figure 1: Bony anatomy of the forearm with the rotational axis indicated by a dotted line. Adapted from (1), with modifications

Due to this close functional relationship, deformities in one or both forearm bones can result in not only aesthetic concerns but also substantial functional limitations. These may include restricted motion, reduced grip strength, pain, joint instability, and, over time, degenerative changes such as osteoarthritis.(8)(15)(10)

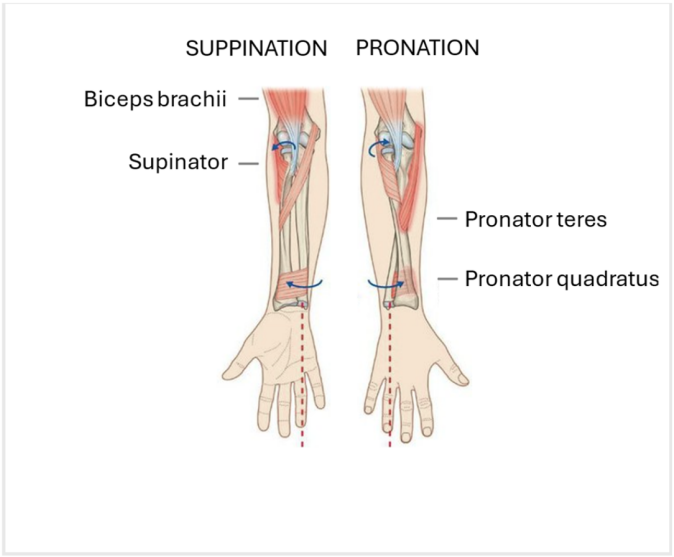


Figure 2: Muscle anatomy enabling pro- and supination. Adapted from (17), with modifications

Figure 2 illustrates how forces from different muscles act on the forearm bones. When transmitted to fracture fragments, these forces can lead to a variety of deformities. Restriction of pronation often occurs due to bony impingement in the proximal forearm, where the clearance between the radius and ulna is typically only a few millimetres. Angular deformities can also shift the attachment sites of the ligaments and the interosseous membrane relative to the rotational axis, creating excessive tension and thereby restricting supination (16). Additionally, axial rotational deformities can cause misalignment of the proximal and distal radioulnar joints (PRUJ and DRUJ)(18).

Corrective osteotomies involve surgically cutting and repositioning the bone, followed by fixation using osteosynthetic plates. As illustrated in Figure 3, there are several types of osteotomies. In addition to a single transverse or oblique cut, a wedge-shaped gap can be created, providing angular correction with bone lengthening. To promote bone growth, this gap could be filled with a bone graft or synthetic material. Closing osteotomies remove a section of bone, after which the fragments are reduced, aligning the osteotomy surfaces to achieve angular correction with shortening. (14). Hybrid osteotomies combine these approaches to maintain bone contact while minimising length changes.

(3) (6)

The execution of corrective osteotomies has advanced over time, yet challenges remain. Traditionally, osteotomy planning is based on two-dimensional (2D) radiographs, which provide limited spatial information. (3) Computer-assisted surgery (CAS) integrates digital three-dimensional (3D) technology for surgical planning and execution. Segmentation of computed tomography (CT) scans allows the creation of digital 3D bone models composed of triangular meshes. These enable comprehensive visualisation and evaluation of the deformity in three

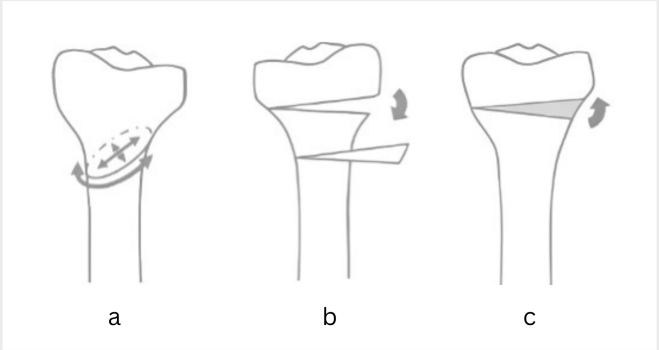


Figure 3: Schematic representation of common osteotomy types: (a) Oblique single cut osteotomy, (b) Closing osteotomy, and (c) Opening osteotomy. Adapted from (14), with modifications

dimensions, which is especially beneficial for complex rotational deformities. This facilitates optimal osteotomy planning, precise positioning of osteosynthesis plates, and the design of patient-specific surgical guides to support accurate execution of the preoperative plan. [7]

A typical computer-assisted corrective osteotomy plan is illustrated in Figure 4. If the contralateral forearm is unaffected, it can be mirrored to serve as a reference to restore the affected bone to its physiological shape. (3) Alternatively, statistical shape models provide an averaged template derived from comparable patient populations. (20) After aligning the proximal side of the reference bone to the corresponding side of the affected bone, various osteotomy approaches can be virtually simulated and assessed. After identifying a suitable osteotomy site, the model is cut, and the distal segment is aligned with the reference model. (3) Additionally, the placement of plates and screws can be predetermined for optimal fixation. (5) 3D printing enabled not only the production of anatomical models but also the introduction of patient-specific instruments (PSI). These custom 3D-printed surgical guides are designed to fit precisely onto specific areas of the bone, ensuring the accurate execution of the pre-operative plan by directing surgical tools through predefined slits and holes. (13)

Despite these technological advances, challenges remain. Planning such procedures can be time-consuming, often requiring multiple iterative design and evaluation steps. (5) In clinical practice, this limits the number of osteotomy options that are assessed. To address these inefficiencies, the 3D-SurgiGen Add-on was developed, providing a semi-automated framework for generating optimised osteotomy plans. By streamlining the planning process, a semi-automatic approach has the potential to enhance accuracy and efficiency, reduce manual workload and associated clinical costs, and support improved surgical precision and outcomes.

The aim of this study is to evaluate the 3D-SurgiGen Add-on for preoperative planning of reconstructive forearm osteotomies in patients with post-traumatic malunions of the radius and ulna. Conducted at Erasmus MC, Rotterdam, and LUMC, Leiden, the study investigates whether the Add-on can serve as a time-efficient tool to generate accurate surgical plans and provides insights for its further development.

2. Methods

2.1. 3D-SurgiPlan Add-on

The 3D-SurgiGen Add-on provides tools for the semi-automatic creation of preoperative surgical plans for forearm osteotomies. Developed as a plugin for Blender (Blender Foundation, Amsterdam, The Netherlands), an open-source 3D graphics platform, the Add-on facilitates preparation of a contralateral forearm model as a reference bone, adjustment of the reference bone scale, and specification of parameter bounds for the optimization algorithm. Based on these inputs, a set of osteotomy plans is automatically generated.

Key reconstruction goals include restoring the overall shape of the bone, particularly ensuring the anatomical position of the DRUJ, and maintaining continuity of the bone surfaces, especially in the region where the osteosynthesis plate will be placed, to facilitate accurate fixation and alignment.

2.2. Add-on architecture

Preoperative planning begins with importing 3D models of the patient's forearms into Blender. The user annotates the bone models with relevant landmarks to establish orientation. The reference bone is then automatically mirrored and aligned with the proximal region of the affected bone, with the option for the user to adjust the alignment percentage.

The Add-on provides an interactive environment for scaling the reference bone to optimize contact between bone fragments. Adjustments in scale, osteotomy position, and distal alignment are visualized in real time, enabling rapid evaluation of potential configurations.

Osteotomies are defined by two planes: the pre-cut plane, marking the initial bone cut, and the post-cut plane, to which the distal segment is subsequently repositioned. The planes are positioned according to a relative height along the bone's centerline. Depending on their relative height, an opening, closing, or hybrid osteotomy can be simulated. This approach allows correction through multiple osteotomies, bringing several bone segments into the desired alignment.

This is achieved by duplicating the affected bone and aligning it with the distal reference. Once cut, this model represents the distal segment in its optimal DRUJ position. Slight deviations from the ideal distal alignment are permitted

To allow the user to incorporate patient-specific requirements and surgeon-defined preferences, translational, axial rotational, and tilt deviations from the optimal distal alignment are defined along anatomical planes. The user can adjust the minimum and maximum values of these variables and observe the effects of these and other parameter bounds on the final postoperative result in real time.

The optimisation cost function quantifies similarity between the postoperative bone and the reference. It comprises two clinically relevant objectives: (1) the distance between centerline points of the affected and reference bones, and (2) alignment

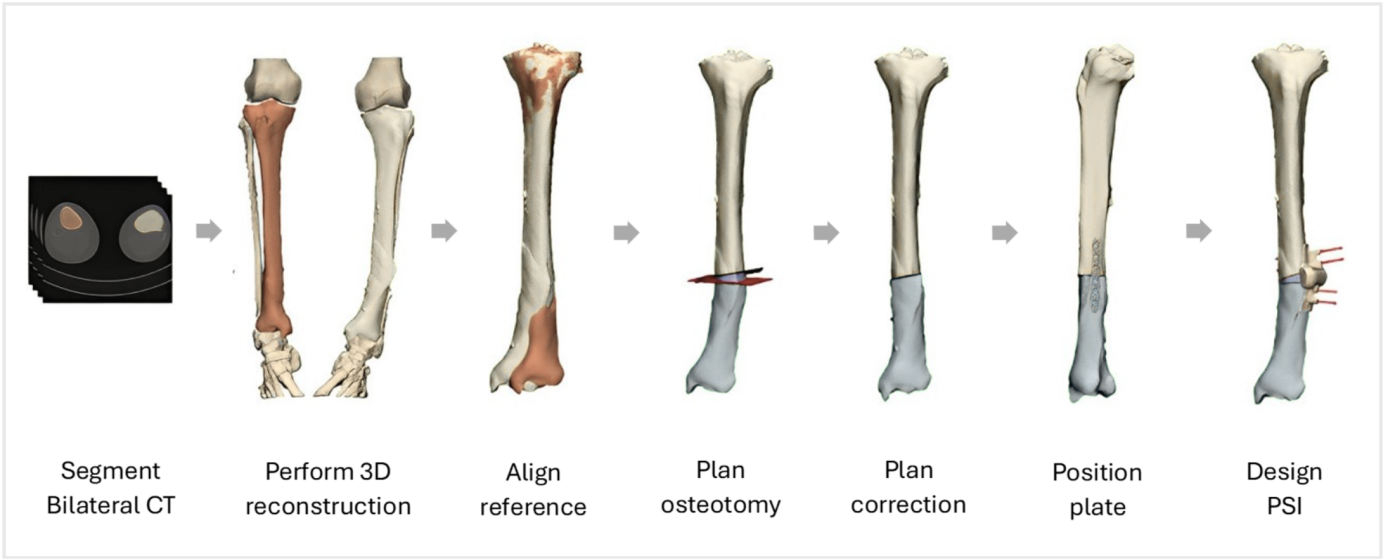


Figure 4: A typical computer-assisted corrective osteotomy, illustrated for the tibia. planning workflow. Adapted from (13), with modifications.

of osteotomy surface borders, critical for accurate plate placement. By evaluating only points defining the osteotomy surfaces instead of large bone surfaces, the process remains computatationally efficient.

The optimisation algorithm generates multiple solutions representing trade-offs between clinical objectives. These solutions are ranked according to deviation from ideal distal alignment, allowing the user to select the plan that best meets clinical and personal preferences.

2.2.1. Study Participants

The study included five patients with post-traumatic midshaft malunion of the radius and ulna, for whom corrective surgery was indicated for both bones and preoperative CT scans were available. Three-dimensional bone models were generated from these scans, with segmentation and model enhancement performed in Mimics and 3-matic (Materialise, Leuven, Belgium).

2.2.2. Study design

Preoperative planning was conducted in parallel using both the 3D-SurgiGen Add-on and manual planning with 3-matic (Materialise, Leuven, Belgium). All plans were generated by a single experienced biomedical planner, and the order of software use was randomised to reduce bias. Additional iterations were allowed even after a plan was deemed clinically acceptable.

Two orthopaedic surgeons experienced in corrective forearm osteotomies independently evaluated the osteotomy plans. Each plan was reviewed in a blinded manner, focusing on overall bone shape, joint alignment, osteotomy plane height and orientation, and osteotomy alignment, with assessment of both anatomical plausibility and surgical feasibility.

2.3. Deformity characteristics

Indication of deformity severity is provided by angular measurements, as depicted in Figure 5, derived using in-house developed software. Dorsal–Volar Angulation (DVA) represents the maximal angulation of the bone in the sagittal plane. Radio–Ulnar Angulation (RUA) represents the maximal angulation in the coronal plane. Maximum Angle of Deformity (MDA) was calculated by combining these measurements. Internal Rotation (IR) represents the maximal axial rotation around the bone's longitudinal axis.[1]

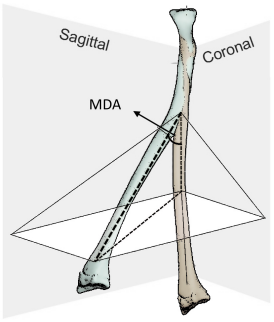


Figure 5: Depiction of the Maximum Deformity Angle (MDA) as calculated from Dosal-Volar Angulation (DVA, saggital) and Radio-Ulnar Angulation (RUA, coronal). MDA = $\sqrt{\tan^2(\text{DVA}) + \tan^2(\text{RUA})}$. Figure is adapted from (9), with modification.

2.3.1. Performance

To compare the performance of the semi-automatic and manual planning methods, a semi-structured evaluation form was

[1]DVA is equivalent to Dorsal Angulation of the Radius (DAR) and Palmar Angulation of the Ulna (PAU), with reversed convention. RUA is equivalent to Radial Angulation of the Radius (RAR) and Ulnar Angulation of the Ulna (UAU). IR is equivalent to Internal Rotation of the Radius (IRAR) and Ulna (IRAU).

used to assess each bone individually (see Appendix). Each plan was rated by both observers as very bad, bad, good, or very good. Plans receiving a rating of good or very good from both observers were considered accepted.

2.3.2. Failure criteria

Observers were asked to report failure criteria and to identify perceived strengths and weaknesses of each plan, as outlined in the evaluation form (see Appendix). Reduced deformity measurements are provided for context regarding possible failures or weaknesses, but were not used to guide the observers' assessments, in line with standard clinical practice.

2.3.3. Duration

To evaluate planning efficiency, the total planning time was recorded from initial model inspection to the final export of both surgical plans, not necessarily as a continuous session. The computational run time of the 3D-SurgiGen Add-on was separately logged.

2.3.4. Questionnaire

The usability of the 3D-SurgiGen Add-on was evaluated by using a qualitative questionnaire, as presented in the Appendix. General demographic information was collected to provide context regarding the participants' experience with preoperative planning for forearm deformity correction. The questionnaire addressed functionalities and aspects of interface design, usability, clinical relevance, and overall satisfaction, using a six-point scale: *Strongly disagree, disagree, neutral, agree, strongly agree*, or *not applicable*. Results are presented as categorised diverging bar charts illustrating the distribution of responses per question. Participants were also invited to comment on perceived strengths and limitations.

All participants performed at least a single bone correction using the Add-on. Those without prior experience completed a dedicated test case following a compact step-by-step guide. Assistance provided during software use, along with participants' comments, was translated into actionable insights and categorised into seven domains: *Functionalities, Performance, User Interface, Terminology, Object Visibility, Default Values,* and *Documentation.* Each entry was linked to the relevant section of the software. For each item, the number of participants requiring assistance or providing feedback was recorded, together with the observer's assessment of its priority, rated as *low, medium,* or *high.* The qualitative feedback was analysed thematically to identify key themes and insights into usability.

3. Results

3.1. Patient characteristics

The deformity characteristics of the test dataset are summarized in Table 1. Five patients with combined radius and ulna deformities were included. For the ulna, the median maximal dorsal–volar angulation (MDA) was 16.5° (range 10.0–24.7°) and the median internal rotation (IR) was 27.1° (range 0.1–40.7°). For the radius, the median MDA was 14.9° (range 8.9–17.6°) and the median IR was 8.9° (range 3.4–52.4°).

Table 1: Deformity characteristics (N = 10)

Ulna deformity characteristics (n = 5)	
Maximal Deformity Angle (MDA, absolute °)	16.5 [10.3 – 24.0]
Dorsal-Volar Angulation (DVA, absolute °)	16.1 [1.0 – 19.8]
Radio-Ulnar Angulation (RUA, absolute °)	9.7 [3.6 – 18.9]
Internal Rotation (IR, absolute °)	27.1 [0.1– 40.7]
Radius deformity characteristics (n = 5)	
Maximal Deformity Angle (MDA, absolute °)	15.0 [8.9 – 17.6]
Dorsal-Volar Angulation (DVA, absolute °)	5.5 [1.9 – 15.3]
Radio-Ulnar Angulation (RUA, absolute °)	12.9 [7.0 – 16.7]
Internal Rotation (IR, absolute °)	8.9 [3.4 - 52.4]

.

3.2. Clinical performance

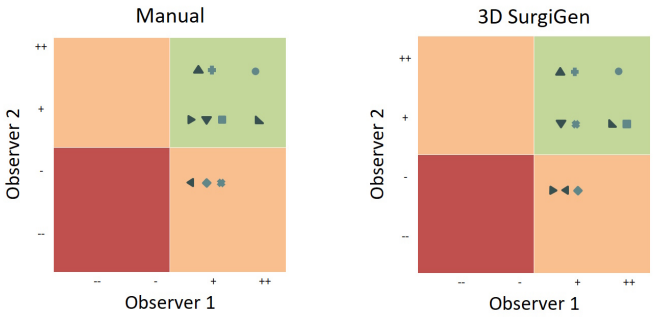


Figure 6: Individual assessment of preoperative plans by two clinical experts. Each case is represented by a marker, with shapes corresponding between methods for the same bone. Light blue markers indicate the ulna, dark blue markers indicate the radius. Points in green area were rated good by both assessors, in red were rejected by both, and in orange were accepted by one assessor and rejected by the other.

Complete patient-level data are provided in Table 1 in the Appendix. Across both methods, three of ten corrective osteotomy plans were rejected by a single observer, with two rejections referring to the same bone. One case received a "very good" rating from both observers for both methods (Figure 6).

Osteotomy plans from both methods failed according to the same criteria and showed identical weaknesses, with each criterion noted an equal number of times. Recorded failure criteria included osteotomy alignment (1) and overall bone shape (3), while perceived weaknesses limiting a "good" plan from being rated "very good" comprised joint alignment (6), osteotomy alignment (5), and overall shape (6). Osteotomy location, osteotomy orientation, wedge size and bone contact were not identified as failures or weaknesses.

The plans created with the 3D-SurgiGen Add-on produced a median MDA of 2.3° (range 1.2–2.9°), compared to 3.6° (range 2.1–5.3°) for manual planning. The median IA achieved with the 3D-SurgiGen Add-on was 3.1° (range 0.4–20.3°), while manual planning resulted in 3.2° (range 0.4–22.4°).

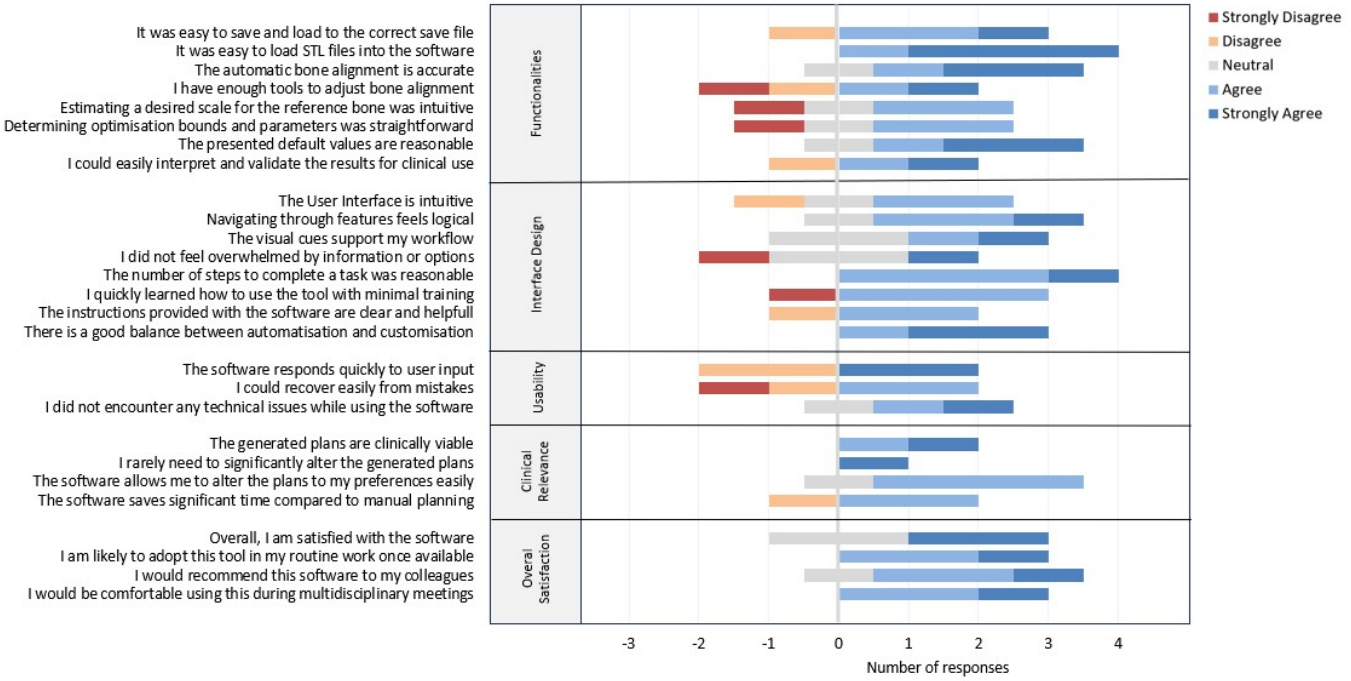


Figure 7: Diverging bar chart showing participants' responses to the questionnaire's categorical questions. Responses of (strongly) disagree contribute to the negative score, responses of (strongly) agree contribute to the positive score, and neutral responses are centered at zero, with each neutral response contributing 0.5 to both scores. Responses marked as "Not applicable" are excluded from the chart.

3.3. Duration

The median time to create osteotomy plans for both the ulna and radius was 42 minutes (range 34–47) using the semi-automated workflow, compared to 30 minutes (range 23–40) for the manual method. The median difference per case was 7 minutes (range 4–17), favouring the manual method. The median computational run time of the 3D-SurgiGen Add-on was 146s (range 110–176s).

3.4. Questionnaire

Figure 7 presents the distribution of the responses for all survey items, with clearly indicated proportions of Strongly Disagree to Strongly Agree. Comments regarding unclear aspects, limitations, or potential improvements were recorded (see Appendix). During this testing phase, three errors attributable to the software were documented.

Among the four participants, one was an orthopaedic surgeon with experience in forearm surgery, and two were 3D lab engineers, one of whom had extensive experience in forearm osteotomy planning. The remaining participant had no prior experience in either osteotomy planning or forearm surgery. Participant responses are summarised in Figure 7 using a diverging bar chart.

File saving and loading were rated positively by most respondents, and STL import and automatic bone alignment were perceived as accurate. Two participants strongly recommended adding a tool for fine-tuning the final alignment around a defined rotation point. Functionalities for estimating the desired scale and setting optimisation bounds received variable scores, Reported issues included distal malalignment during scaling. Additionally, although users should alternate between scaling, plane setup, and distal alignment to fully exploit the software's functionalities, the interface implies a sequential workflow. The participants agreed that the default values were reasonable; however, the ease of adjusting bounds was limited by the presence of irrelevant objects in the display. A major issue concerned the mismatch between the parameters selected for display and those that were actually adjusted. One participant reported difficulty interpreting the results, but attributed this to their limited experience with planning corrective surgeries.

Participants noted that the user interface was generally considered intuitive, with logical navigation, supportive visual cues, and a reasonable number of steps to complete each task. Most participants reported learning to use the software quickly. However, one participant felt overwhelmed by the information and options, struggled to learn the tool with minimal training, and found the provided instructions insufficient. Among those able to evaluate this aspect, all agreed that there was an appropriate balance between automation and customisation. Software responsiveness and error recovery showed substantial variability. Three errors attributed to the software were documented.

Despite these challenges, most participants indicated that they would recommend the software to colleagues and felt comfortable using it during clinical discussions. Two participants specifically highlighted the interactive osteotomy environment and the presentation of multiple osteotomy results as particularly valuable features that enhanced the planning and decision-making process.

4. Discussion

4.1. Main Findings

The present study provides a preliminary evaluation of the 3D-SurgiGen Add-on, a semi-automated software framework developed to support preoperative planning of reconstructive forearm osteotomies.

Both 3D-SurgiGen and conventional manual planning using 3-Matic were evaluated across ten bones from five patients, each presenting diaphyseal deformities of the radius and ulna. Plan acceptance rates were identical for both methods, with three accepted plans out of ten bones and failures attributable to overall bone shape and osteotomy alignment. The nature and frequency of plan failures were consistent across methods, indicating no inherent limitation of the 3D-SurgiGen Add-on. Median planning time for both bones was 42 minutes (range 34–47) using the semi-automated workflow, compared to 30 minutes (range 23–40) for the manual method. The median runtime of the optimisation algorithm was 146 seconds (range 110–176 s), highlighting that the algorithm contributes only a small portion of the total planning time and does not, by itself, lead to faster planning, as steps such as marker placement and initial bone positioning are time-consuming. Variation in usability scores was observed, reflecting differences in participants' experience and interaction with the software.

4.2. Interpretation of results

Several strengths of the study include expert evaluation of osteotomy plans and user-centered feedback, providing insight into clinical applicability and workflow integration. The study design allowed direct comparison within matched cases, and the inclusion of both the radius and ulna ensured a comprehensive assessment of diaphyseal deformities. Having an experienced biomedical planner conduct all tests minimised variation due to differences in expertise. Four participants with varying levels of experience in forearm osteotomy planning provided perspectives ranging from expert to novice, enabling identification of both development needs and opportunities to improve system intuitiveness.

Limitations of this study include the small sample size (n = 10) and the restriction to a limited set of bone types and deformities. Therefore, the generalisability of the findings is limited. The results regarding planning time should be interpreted cautiously. Despite efforts to minimise prior knowledge by using new osteotomy plans, the planner had previously discussed some cases, and strong morphological recognisability of the bones limited anonymisation. This particularly affects the time spent estimating the scaling of the reference bone. Combined with user feedback indicating that the interactive scaling module was not robust due to distal alignment issues, the impact of this deficiency on overall planning time was likely reduced by the presence of bias. Moreover, future evaluations should include a more extensive training phase to allow users to reach a defined proficiency level before testing.

Plan weaknesses identified in this study primarily concerned osteotomy alignment and overall bone shape. Osteotomy alignment was repeatedly noted as a weakness and served as a criterion for plan failure in one instance. User feedback suggested that enabling selective optimisation of specific sides of the osteotomy surfaces could facilitate osteosynthesis plate positioning. Overall bone geometry, such as the radial bow or other critical curvatures, was implicated in three instances of plan failure, suggesting that reliance on the centerline alone is insufficient to capture the overall shape. Since the manual method also failed based on the same criteria, an alternative explanation could be that this deformity requires a second osteotomy. Usability limitations included interruptive object visualisation, nonintuitive parameter interaction, a lack of specific functionalities, and reduced robustness. Suggested improvements comprised incorporating saw loss simulation, tools for selective bone region removal, more explicit feedback on which parameters are currently displayed and adjustable, and clearer visualisation of final parameter settings. Participant responses were heterogeneous, reflecting the deliberate inclusion of diverse experience levels. Only a minimal step-by-step guide was provided, which helped identify areas for improvement independently of the user manual.

During planning, one case was identified that would require a bone graft, which was beyond the scope of this study. Consequently, failure criteria related to excessive scaling were disregarded, and for comparison purposes, both methods were allowed to use the same scale. This adapted case was only successfully completed using the manual method. Pronounced local surface curvatures revealed limitations in handling complex geometries and demonstrating that the centerline metric alone was insufficient for accounting for bone protrusions. The algorithm prioritised optimisation of osteotomy surface alignment rather than minimising protrusions, also highlighting the need for additional tools to customise the plan.

Both the 3D-SurgiGen Add-on and the Materialise software allow variation in alignment based on a user-defined percentage that specifies the relative position of the reference bone. Differences in scaling and proximal alignment, such as axial rotation, can substantially influence the corrections required to achieve distal alignment. Even visually well-aligned proximal references may therefore result in unexpectedly large angular deformity measurements.

In this small test sample, lower maximum deformity angles (MDA) were associated with the 3D-SurgiGen Add-on compared to manual planning. Some large inclination angle (IA) deviations were observed in plans with high quality scores, suggesting that these outliers may reflect measurement uncertainty. Given the small sample size, no direct relationship between quality score and IA deviation can be established.

Measurements in this study were performed using in-house developed software that has not yet been fully validated. Because these measurements were not replicated alongside manual measurements, and different scaling and alignment percentages were applied across methods, no definitive conclusions can be drawn from the observed deviations. Aligning osteotomy plans based on articular surface positioning may offer an alternative approach for plan assessment and comparison, complementing existing evaluation methods that focus on proximal alignment.

4.3. Existing literature

Comparison with previous studies revealed both similarities and differences in methodological approach and outcomes. Carillo et al. (2017) assessed plan quality in 36 radius bones through expert judgement of six readers, reporting 100% feasibility with a fully automated method and an average runtime of 85.38 minutes. In contrast, Dobbe et al. (2020) conducted a proof-of-concept study of 15 automated oblique double-cut osteotomies, achieving 100% plan acceptance with a median runtime of 182 seconds. Optimisation algorithms developed by Carrillo et al. (2020) support opening, closing, and single-cut osteotomies, and Cui et al. (2024) demonstrated closing osteotomy approaches for congenital radioulnar synostosis.(19) Dikland et al. (2023) demonstrated a semi-automatic approach for multi-level closing osteotomies of the femur.(7) Building on these foundations, the 3D-SurgiGen framework extends previous work by providing precise control over allowed distal alignment, an interactive environment for simulating osteotomy plan configurations, the ability to adjust optimisation bounds interactively, and optimisation towards multiple outcome results.

4.4. Implication of Findings

The study's findings have multiple clinical implications. Semi-automated planning has the potential to reduce the workload associated with generating preoperative osteotomy plans while generating configurations that might otherwise remain unexplored. The framework could support effective collaboration between the surgeon and engineer by using the interactive environment to align reduction goals. By presenting multiple outcome options, it may enhance informed surgical decision-making. Semi-automated solutions may ultimately be able to implement clinical decisions such as plate bending or minor bone reshaping, which fully automated optimisation procedures cannot account for and which may otherwise produce plans that are geometrically optimal but clinically impractical.

4.5. Future Research

Future research should focus on refinement of metrics, user interaction, and additional tools for small adjustments. Addressing weaknesses such as distal alignment during scaling, is critical. A promising direction involves a two-stage optimisation approach: first generating a spectrum of osteotomy plans, then allowing the user to select a preferred plan for further optimisation. A time-intensive manual step is the positioning of landmarks. In-house developed codebase for the positioning of these landmarks could therefore contribute to a faster workflow. Although not optimised nor evaluated, the 3D-SurgiGen uses an innovative method combining both the optimisation of multiple osteotomies, with strict control on distal alignment. The methodology could also be extended to other osteotomy types, and bone regions.

Distal alignment is currently controlled by restricting maximal deviations at the midpoint of the articular surface, but positioning the rotational center at clinically significant landmarks, such as the ulnar notch of the radius and the corresponding sigmoid fossa of the ulna, could better reflect clinically relevant alignment.

Overall, this study demonstrates that semi-automated interactive planning can achieve comparable plan quality to manual approaches while providing opportunities for iterative adjustment, improved workflow efficiency, and enhanced collaboration. Future work should focus on overcoming current limitations, refining interactive features, and validating the framework across larger, more diverse patient populations.

5. Conclusion

This in silico study demonstrates the potential of the 3D-SurgiGen Add-on as a clinically relevant tool for semi-automated preoperative planning of forearm osteotomies. The Add-on can generate feasible planning solutions and enables interactive exploration of surgical options, supporting surgeon-specific preferences by offering a range of solutions that balance competing clinical objectives. In this study of five cases with deformities of both the ulna and radius, semi-automated planning did not reduce planning time, and the small sample size limits generalizability. With continued refinement, The 3D-SurgiGen Add-on may serve as an efficient complementary tool for clinical decision-making, providing insight into surgical options and enabling the generation of customised preoperative corrective osteotomy plans.

References

[1] Forensic science student resource arm & hand. Accessed: April 2, 2025.

[2] B. Saravi et al. Corrective osteotomy of upper extremity malunions using three-dimensional planning and patient-specific surgical guides: Recent advances and perspectives. *Front Surg*, 8:615026, 2021.

[3] D. Bauer et al. Conventional versus computer-assisted corrective osteotomy of the forearm: a retrospective analysis of 56 consecutive cases. *The Journal of hand surgery*, 42(6):447–455, 2017.

[4] E. Schemitsch et al. The effect of malunion on functional outcome after plate fixation of fractures of both bones of the forearm in adults. *The Journal of Bone Joint Surgery*, 74:1068–1078, 1992.

[5] F. Carrillo et al. A time saver: Optimization approach for the fully automatic 3d planning of forearm osteotomies. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 488–496. Springer, 2017.

[6] F. Carrillo et al. An automatic genetic algorithm framework for the optimization of three-dimensional surgical plans of forearm corrective osteotomies. *Med Image Anal*, 60:101598, 2020.

[7] F.A. Dikland et al. *Semi-Automatic Generation of Preoperative Surgical Plans for Complex Femoral Deformity Correction*. Msc thesis, 2023.

[8] J. Cognet et al. Distal radius malunion in adults. *Orthop Traumatol Surg Res*, 107(1S):102755, 2021.

[9] K. Roth et al. Accuracy of 3d corrective osteotomy for pediatric malunited both-bone forearm fractures. *Children*, 10:21, 12 2022.

[10] M.J. Richard et al. Malunions and nonunions of the forearm. *Hand Clin*, 23(2):235–43, vii, 2007.

[11] M.Massobrio et al. Forearm post-traumatic deformities: classification and treatment. *Injury*, 45(2):424–427, 2014.

[12] M.N. Rasool et al. Radioulnar fusion for forearm defects in children-a salvage procedure. *SA Orthopaedic Journal*, 7(1):60–67, 2008.

[13] N. Assink et al. A two-step approach for 3d-guided patient-specific corrective limb osteotomies. *J Pers Med*, 12(9), 2022.

[14] P. Belei et al. Computer-assisted single- or double-cut oblique osteotomies for the correction of lower limb deformities. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, 221:787 – 800, 2007.

[15] P. Jayakumar et al. Reconstruction of malunited diaphyseal fractures of the forearm. *Hand (N Y)*, 9(3):265–73, 2014.

[16] P. Reyniers et al. The role of 3d technology in corrective osteotomy for forearm malunion. *Journal of Hand Surgery (European Volume)*, 2025.

[17] R.L. Drake et al. *Gray's Anatomy for Students*. Churchill Livingstone, an imprint of Elsevier, Inc., 2nd edition, 2009. All rights reserved.

[18] T.J. Graham et al. Disorders of the forearm axis. *Hand Clinics*, 14(2):305–316, 1998.

[19] Y. Cui et al. A dual dimensional optimization strategy for automatic osteotomy preoperative planning in congenital radioulnar synostosis. *Scientific Reports*, 14(1):30759, 2024.

[20] E.M. van Es et al. Corrective osteotomy in a patient with congenital absence of pronation based on three-dimensional statistical shape modeling. *Hand*, 19(6):NP1–NP5, 2024.

QUESTIONNAIRE ADDON

3D-SURGIGEN ADDON

Demographics

1. What is your speciality

☐ Orthopedic Surgeon ☐ 3D lab engineer ☐ Other, namely... ☐

1. How many years have you been practicing your specialisation?

☐ 0-1 y ☐ 1-5y ☐ 5+y ☐ N/A

1. How much experience do you have in using osteotomy planning software?

☐ first time ☐ once or twice ☐ occassionally ☐ regularly

1. Do you have experience with forearm deformity corrections?

☐ No ☐ once or twice ☐ occassionally ☐ regularly

Interventions

Provided instructions

Error log

Additional space for comments

QUESTIONNAIRE ADDON

3D-SURGIGEN ADDON

Functionalities	Strongly disagree		Neutral		Strongly agree	N/A
1. It was easy to save and load to the correct save file	☐	☐	☐	☐	☐	☐
2. It was easy to load STL files into the software.	☐	☐	☐	☐	☐	☐
3. The automatic bone alignment is accurate	☐	☐	☐	☐	☐	☐
4. I have enough tools to adjust bone alignment	☐	☐	☐	☐	☐	☐
5. Estimating a desired scale for the reference bone was intuitive	☐	☐	☐	☐	☐	☐
6. Determining optimisation bounds and parameters was straightforward	☐	☐	☐	☐	☐	☐
7. The presented default values are reasonable	☐	☐	☐	☐	☐	☐
8. I could easily interpret and validate the results for clinical use	☐	☐	☐	☐	☐	☐
Interface Design						
1. The User Interface is intuitive	☐	☐	☐	☐	☐	☐
2. Navigating through features feels logical	☐	☐	☐	☐	☐	☐
3. The visual cues support my workflow	☐	☐	☐	☐	☐	☐
4. I did not feel overwhelmed by information or options	☐	☐	☐	☐	☐	☐
5. The number of steps to complete a task was reasonable	☐	☐	☐	☐	☐	☐
6. I quickly learned how to use the tool with minimal training	☐	☐	☐	☐	☐	☐
7. The instructions provided with the software are clear and helpfull	☐	☐	☐	☐	☐	☐
8. There is a good balance between automatisation and customisation	☐	☐	☐	☐	☐	☐
Usability						
1. The software responds quickly to user input	☐	☐	☐	☐	☐	☐
2. I could recover easily from mistakes	☐	☐	☐	☐	☐	☐
3. I did not encounter any technical issues while using the software.	☐	☐	☐	☐	☐	☐
Clinical Relevance						
1. The generated plans are clinically viable	☐	☐	☐	☐	☐	☐
2. I rarely need to significantly alter the generated plans	☐	☐	☐	☐	☐	☐
3. The software allows me to alter the plans to my preferences easily	☐	☐	☐	☐	☐	☐
4. The software saves significant time compared to manual planning	☐	☐	☐	☐	☐	☐
Overal satisfaction						
1. Overall, I am satisfied with the software	☐	☐	☐	☐	☐	☐
2. I am likely to adopt this tool in my routine work once available	☐	☐	☐	☐	☐	☐
3. I would recommend this software to my colleagues	☐	☐	☐	☐	☐	☐
4. I would be comfortable using this during multidisciplinary meetings	☐	☐	☐	☐	☐	☐

In case technical issues did occur, please describe:

What features or improvements would you most like to see added?

What was the most helpful feature of the software?

Are there any specific limitations or concerns you encountered while using the software.

Any additional comments or feedback?

Failed		Failure criteria	Strength		
A	B		A	B	=
		Osteotomy position			
		Osteotomy orientation			
		Bone length			
		Segment length			
		Bone contact			
		Bone protrusion			
		Joint alignment			
		Overall shape			
		Other, namely...			

Figure 2: Form for first run performance assessment with failure criteria: A and B refer to the osteotomy plans, either generated by the software, or manually planned. If the planning has failed, tick the relevant failure criteria underneath 'Failed'. For each criterion, tick the box indicating which plan performs better or whether they perform comparably. Accompany each with a brief justification explaining the rationale behind the assessment.

				Scale factor		Quality score (obs1/obs2)		MDA (°)		IA (°)	
Patient	Bone	MDA (°)	IA (°)	Manual	3D-SurgiGen	Manual	3D-SurgiGen	Manual	3D-SurgiGen	Manual	3D-SurgiGen
1	Radius	10.37	-0.17	0.985	1.000	+ / +	- / +	3.90	1.54	14.31	-2.84
2	Radius	24.03	28.67	0.960	0.960	- / +	- / +	5.34	2.90	22.44	17.33
3	Radius	21.73	-27.07	0.900	0.900	++ / +	++ / +	2.07	2.64	-3.50	-20.26
4	Radius	16.45	40.73	1.000	1.000	+ / +	+ / +	3.61	1.22	-2.89	6.44
5	Radius	16.34	-5.33	0.965	0.985	+ / ++	+ / ++	2.57	2.38	0.54	-8.32
1	Ulna	17.35	-52.43	0.985	1.000	++ / ++	++ / ++	2.59	2.19	-4.96	3.30
2	Ulna	14.97	-29.02	0.960	0.960	++ / ++	++ / +	3.84	2.85	-6.14	-2.93
3	Ulna	8.83	-7.34	0.900	0.900	- / +	+ / +	3.53	2.20	-0.94	-0.51
4	Ulna	13.32	8.89	1.000	1.000	+ / +	+ / ++	2.50	2.05	0.44	-0.36
5	Ulna	17.53	-3.41	0.965	0.965	- / +	- / +	4.04	2.40	1.36	1.30

Table 1: Characteristics of each bone (bonetype, Maximal Deformity Angle [MDA] and Internal Rotation [IA]) with applied scale factor of the reference bone, observer quality assessments, and angular measurements (MDA, IA), shown separately for the results of both Manual and 3D-SurgiGen methods.

	Provided instructions	Names	Functions	UI	Visibility	Manual	Performence	Defaults	Panel	Participant 1	Participant 2	Participant 3	Participant 4	Priority
1	Difficult to select affected side before the STL file has been loaded.			x					0	x	x	x	x	H
2	Clarify that different optimization runs require separate run files and notes.					x			0	x	x	x	x	H
3	Clarify meaning of “affref”.	x				x			0	x	x	x	x	H
4	Clarify meaning of "template alignment"	x				x			4	x	x	x	x	H
5	Clarify goal of panel 6: for scaling and as reference during bounds adjustments; does not set bounds.	x				x			6	x	x	x	x	H
6	Clarify how to use the distal and plane bounds to visualise the osteotomy.			x		x			6		x	x	x	H
7	Clarify use of advanced buttons for alignment when there is a scale difference.					x			6		x	x	x	H
8	Clarify purpose of panel 9: Adjusting optimalisation bounds.			x		x			9	x	x	x	x	H
9	Clarify in the manual that first page provides general info and other pages correspond to panels.					x			x	x	x	x		H
10	Clarify difference between “branch” and “copy”.					x			0	x	x	x	x	M
11	Clarify the viewport shading should be in material preview.			x					0	x	x			M
12	Clarify that workflow starts directly with run 1 after patient selection.			x					0	x			x	M
13	Clarify that not all bones can be shown through the visualisation tools in the first steps.			x					0	x			x	M
14	Clarify how to save via Blender UI or with Ctrl+S.			x					0	x			x	M
15	Clarify how precisely markers should be placed.					x			2	x	x	x	x	M
16	Clarify that radial marker can be on either side of the styloid–ulnar dome line, but must be consistent.					x			2	x	x	x	x	M
17	Clarify how to interpret distal radius orientation, e.g. indicate volar side.					x			2	x	x	x		M
18	Clarify that all four bones must be annotated.			x		x			2		x		x	M
19	Clarify that markers won’t be placed if the wrong bone is selected.				x				2		x			M
20	Clarify that bones will become visible after using “Align Bones”			x		x			4		x		x	M
21	Clarify that the reference bones must be aligned either distally or proximally, not across the entire bone.					x			4		x	x		M
22	Clarify meaning of "demo bones"	x				x			6	x			x	M
23	Clarify color meanings.			x		x			x		x	x		M
24	Clarify the bones become visible after using the initialization buttons.		x						x		x		x	M
25	Clarify meaning of "affected" and "reference" bones. At Erasmus MC the term "affected (A)" and "not affected (NA)" is more commonly used.	x				x			x		x		x	M
26	Clarify general mouse manipulation in Blender, e.g., Select + Period.					x				x	x		x	M
27	Clarify where the STL load window shows which bone should be loaded.					x			0	x				L
28	Clarify that for each "Import STL" button, the corresponding STL file must be loaded.		x						1		x			L
29	Clarify the marker position on the proximal radial head.					x			2	x			x	L
30	Clarify in the manual that marker colors correspond to image points.					x			2	x				L
31	Clarify that pressing Escape exits selection freeze mode.			x		x			2		x		x	L
32	Clarify difference between pressing the checkbox and the marker button.			x					2			x		L
33	Clarify that all centerlines should be verified, even if not displayed.				x				3		x			L
34	Clarify difference between "waypoints" and "markers".	x				x			x	x				L

	Comments	Names	Functions	UI	Visibility	Manual	Performence	Defaults	Panel	Participant 1	Participant 2	Participant 3	Participant 4	Priority
35	The visualization panel does not necessarily reflect the bone whose parameters are currently being adjusted, leading to potential confusion between radius and ulna.		x						x	x	x	x	x	H
36	Lack of visual connection to show which setting is being changed.			x					0	x	x		x	H
37	Bone visualization becomes cluttered after using branching functionality.				x				0	x	x		x	H
38	Visualisation is not intuitive. When the user wants to hide specific bones, which cannot be hidden through the visualisation panel, the user tries to navigate trough the Blender object trees. However, these hidden bones reappear after using SurgiGen functionalities.					x			0				x	H
39	When parameters are modified, it should be clearly indicated that realignment is required.			x					4	x				H
40	Distal alignment in the scaling module is not robust.		x						6		x		x	H
41	Scaling does not update when values are typed manually.		x						6		x		x	H
42	The “Show both demo bones” functionality is confusing; users would expect to be able to select the show-buttons for both the minimum and maximum bounds to achieve this behavior.		x						9	x	x	x	x	H
43	Changes in the visualization panel have no effect on the demo bones.				x				9	x	x	x	x	H
44	It is not evident which bone represents the minimum or maximum distal bounds.			x					9	x	x		x	H
45	It is unclear that the two “Show” buttons correspond to displaying the minimum or maximum bounds.			x					9	x	x	x	x	H
46	There is currently no tool to define an area with limited step-off for plate positioning.		x						9				x	H
47	The rotation center for adjusting distal alignment is presently at the articular surface center; it should instead correspond to the distal ulnar notch (radius) or ulnar dome (ulna) to preserve clinically important alignment.		x						9				x	H
48	Implement a functionality to force the removal of a specific bone region.		x						9				x	H
49	Results should also display the final parameter settings.			x					10	x			x	H
50	Give a specific color (pink) to the wedge that should be removed.		x						10				x	H
51	It is confusing that all steps are performed for both bones in a single run, while optimization requires separate run files for each bone.		x						10				x	H
52	Implementation of saw loss		x						x	x		x	x	H
53	Adjusting object visibility is slow.						x		0	x	x		x	M
54	It appears that the user must select the bone in the visibility panel instead of initiating it from the main workflow.			x					0	x			x	M
55	After selecting a landmark, all unrelevant bones are displayed again.				x				2	x	x	x	x	M
56	Implementation of automated marker positioning develloped at Erasmus MC.		x						2				x	M
57	It is unintuitive that the user must manually click “get bone data.”			x					3	x			x	M
58	Planes associated with a hidden bone remained visible.				x				4	x			x	M
59	It appears that “align bones” must be applied to each visualized bone separately.			x					4		x			M
60	The name “bounds” on panel 6 is confusing.	x							6	x			x	M
61	Panel 6 implies a sequential workflow, but in practice, users should alternate between scaling, plane setup, and distal alignment.			x		x			6	x			x	M

	Comments (continued)	Names	Functions	UI	Visibility	Manual	Performenc	Defaults	Panel	Participant	Participant	Participant	Participant	Priority
62	The purpose of the “estimate ROI” feature is unclear.		x			x			7	x			x	M
63	The default evaluation value of 2000 is considered high.							x	9	x		x	x	M
64	Planes remain visible even after the corresponding subpanels are hidden.				x				9	x			x	M
65	The purpose of the different bounds is unclear.	x							9		x		x	M
66	Optimization parameters are not clearly explained.			x		x			9		x		x	M
67	Currently, the adjustable distal bounds are limited to x-tilt and Y-tilt for simplicity. However, to allow user modifications, such as permitting the ulna to be slightly shorter than the radius, these controls should be made visible.		x						9				x	M
68	Currently, the adjustable distal bounds are limited to x-tilt and Y-tilt for simplicity. However, to allow user modifications, such as permitting the ulna to diverge in the ulnar direction, these controls should be made visible.		x						9				x	M
69	Previous results are lost if “Initiate” is pressed accidentally, which can disrupt workflow.		x						10				x	M
70	Clarify slots and override functionality .					x			x	x		x	x	M
71	The “initialise” button should be included in navigation controls.			x					x	x		x	x	M
72	It is unclear which button is pressed. This could be visually emphasized with bold text or a frame.			x					x	x	x			M
73	Viewport shading does not automatically switch to material preview to show bone colors.		x						x	x	x			M
74	File selection, visibility, and transparency controls appear disorganized.			x					0	x			x	L
75	Users may mistakenly believe they need to click ‘import’ again to select a bone.			x		x			0	x				L
76	There is no save button in the user interface.			x					0	x				L
77	Instead of “get runs,” the “save and load” button is shown.			x					0			x		L
78	In the manual, proximal and distal bones are listed next to each other, whereas in Blender, the ulna and radius are listed together.					x			2	x				L
79	The mouse cursor does not indicate when rotation is not possible during marker selection.			x					2	x				L
80	Marker points are not sufficiently noticeable, color coding should be considered.			x					2	x				L
81	Consider color blindness in color coding					x			2	x				L
82	Consider checkmarks next to bone names to indicate which bones are marked			x					2	x				L
83	"Ridge center" is a confusing shorter name for "Center of fossae ridge".	x							2			x		L
84	Loading each bone individually is perceived as inefficient, but requiring specific file names is not considered an improvement.		x						2				x	L
85	Centerline generation did not reliably handle a deformity in the distal bone; repositioning the markers resolved the issue.		x						3				x	L
86	The advanced button is not easily noticeable.			x					4		x			L
87	It would be more useful if the “Apply settings from other bone to this bone” option were replaced with “Apply these settings to other bone.”		x						4			x		L
88	Steps 5 and 8 are redundant, as the same bones are aligned in the preceding steps.			x					4				x	L
89	Ulna and radius alignment tools are positioned together, which is inconsistent with other panels.			x					4				x	L
90	The labels x, Y, and Z used for scaling are not intuitive.	x							6			x	x	L
91	On panel 6, the reference bone aligned to the distal side is no longer available, which may limit assessment of the region of interest (ROI).		x						6			x		L
92	The “Show” label disappears on small screens.			x					9		x			L
93	Colors used to distinguish functionalities within the user interface are not sufficiently contrasting.			x					9		x			L
94	It is confusing that only the hyperparameters are visible in step 10; it is suggested to include the optimalisation button at step 9.			x					10		x			L
95	The algorithm could automatically favor osteotomy surface alignment at the bone sides suitable for plate positioning.		x						10				x	L
96	The workflow assumes that the bone remains perfectly aligned proximally and that adjustments to distal alignment only affect the DRUJ. In practice, both proximal and distal alignments may be suboptimal. It would be useful to visualize the final alignment of both bones.		x						10				x	L
97	Providing a heatmap of the corrected bone compared to the affected bone would improve visualization.		x						10				x	L
98	Implementation of the rotation model develloped at Erasmus MC.		x						10				x	L
99	Implementation of the automated measurements software develloped at Erasmus MC.		x						10				x	L
100	Implementation of a tool to manually adjust the distal alignment of the final result, with automatic adjustment of the cutting planes.		x						10				x	L
101	TODOs and development notes should not be visible.			x					x	x			x	L
102	Clarify manipulating sliders					x			x	x			x	L
103	Processing could be indicated by an hourglass cursor			x					x	x				L
104	Variables cannot be adjusted by scrolling.		x						x	x				L
105	Tooltips are incorrect.			x					x	x				L
106	Files should be automatically saved when switching panels.		x						x	x				L
107	Two versions of the SurgiGen panel are visible, causing confusion.			x					x			x		L
108	Blender “fly” mode can be activated using the right mouse button, but requires prior adjustment of Blender settings.		x						x			x		L
109	Implementation of multiple osteotomies		x						x		x			L
110	Implementation of tools allowing the use of a segment from one bone to be inserted into another.		x						x				x	L

	Error Log	Names	Functions	UI	Visibility	Manual	Performence	Defaults	Panel	Participant 1	Participant 2	Participant 3	Participant 4	Priority
114	Branch function also removes objects from current run file.		x						0	x	x	x	x	H
112	Faulty generation of centerline at case with deformed distal ulna led to inaccurate alignment		x						3				x	M
113	Unknown error > Bone not selected		x						2				x	L

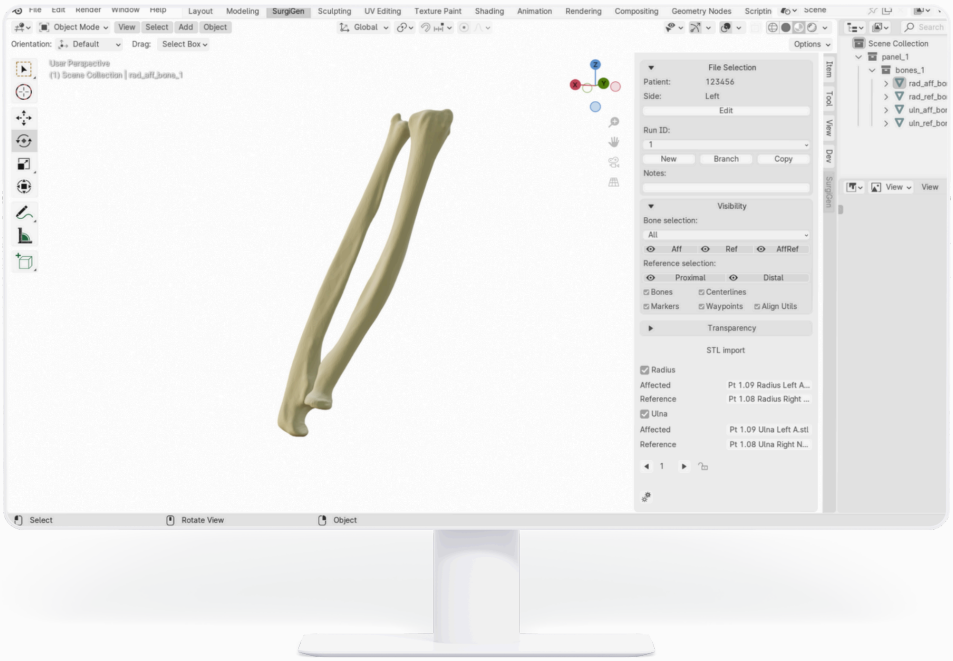
Part II:
Step-by-step Guide

This chapter presents a practical step-by-step guide for the osteotomy planning add-on. It includes installation instructions, a guide to the use of the main functionalities, and troubleshooting tips that support end users in applying the tool effectively.

3D SURGIGEN

Osteotomy planning

Step-by-step Manual



3D SurgiGen - Open-source add-on for osteotomy planning under development
Developed for use with Blender 4.1.1 © Blender Foundation, 2024
This document is issued in October 2025

3D SURGIGEN

Overview

This concise step-by-step describes how to use the 3D-SurgiGen Add-on, a semi-automated framework designed for preoperative planning of forearm osteotomies. It guides users through each panel of the add-on in a step-by-step workflow, from data import to optimalisation result selection.

The add-on operates within Blender 4.1.1 and provides a structured interface that allows you to navigate between steps using arrow buttons.

Each step corresponds to a dedicated panel within the add-on, and this manual mirrors that same structure. Before proceeding, it is important to understand the four core models used throughout the workflow:

- aff (yellow) – the affected bone that requires correction.
- ref (blue) – the mirrored contralateral bone used as the anatomical reference.
- affref (green) – the affected bone aligned to the reference bone at the distal side, ensuring both bones correspond correctly before further analysis and optimization.
- result (purple) – the optimized outcome of the surgical plan, representing the final corrected configuration.

This step-by-step guide gives a compact overview of the 3D-SurgiGen Add-on, organized into three sections: Installation, General Controls, and Panel Workflow.

Section 1 – Installation

Instructions for installing the add-on in Blender 4.1.1.

Section 2 – General Controls

Describes interface elements that apply to all panels:

- File Selection – managing new run files
- Visibility Settings – toggling bone and plane visibility.
- Transparency – adjusting display transparency for better model inspection.
- Add-on Settings – visual preferences.

Section 3 - Panel Workflow

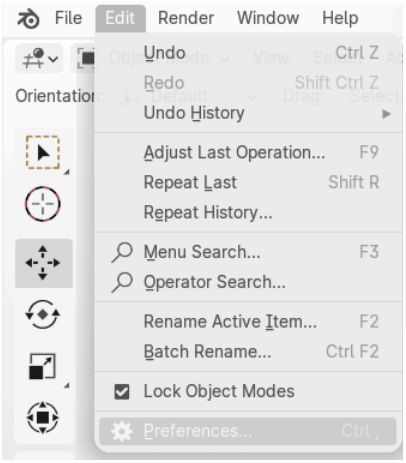
The following sections correspond directly to the panels in the add-on interface.

Panel	Name	Purpose
1	STL Import	Import the aff and ref bone models (.stl).
2	Landmark Positioning	Define anatomical landmarks on both aff and ref.
3	Bone Analysis	Retrieve bone centerlines and geometric axes.
4	Bone Alignment	Mirror ref, align ref to aff
5	Template Alignment	Align a copy of aff to the ideal distal alignment of ref.
6	Interactive Scaling	Fine-tune the scaling of ref interactively.
7–8	Alignment Verification	Reassess aff–ref alignment consistency after scaling adjustments.
9	Optimization Setup	Define optimization settings, including maximum plane positions and distal alignment constraints.
10	Optimization Execution & Result Selection	Run the optimization algorithm and select among multiple generated results.

Section 1 – Installation

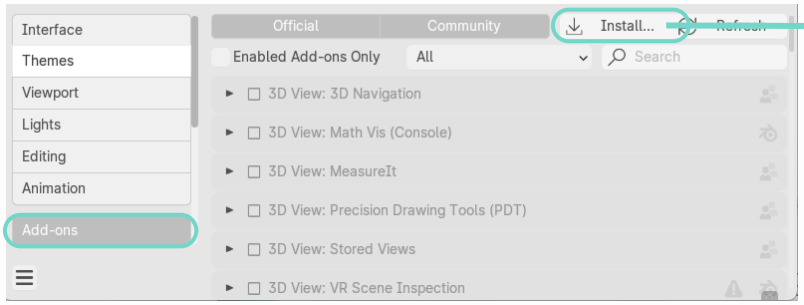
File Selection

- Download the Add-on – obtain the latest version of the 3D-SurgiGen Add-on as a .zip file.
- Open Blender Preferences – in Blender 4.1.1, go to the Edit menu → Preferences → Add-ons.
- Install the Add-on – click Install..., navigate to the downloaded .zip file, and select it.
- Enable the Add-on – check the box next to 3D-SurgiGen in the add-on list to activate it.



Edit menu

Preferences...



Install....

Standard panel layout will be automatically applied upon Blender startup. The add-on panel can be accessed in two locations within Blender:

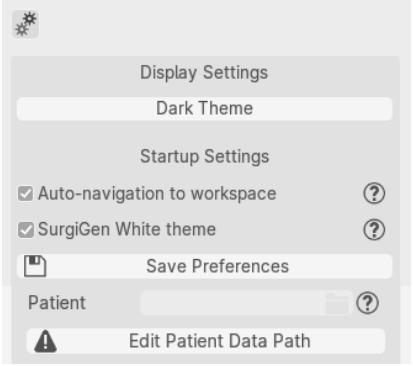
3D Viewport

Image Editor

You can navigate to either location by changing the editor type using the drop-down menu in the top-left corner of a field. Press N to display the side panel, which contains the SurgiGen tabs.

Add-on settings

The user can toggle between a light and dark theme and choose which theme is applied at startup. Automatic navigation to the Add-on workspace can be disabled. For first-time use, a folder must be selected to store patient data.



Open add-on settings

Toggle Theme

Set startup preferences

Folder for patient data should be defined once

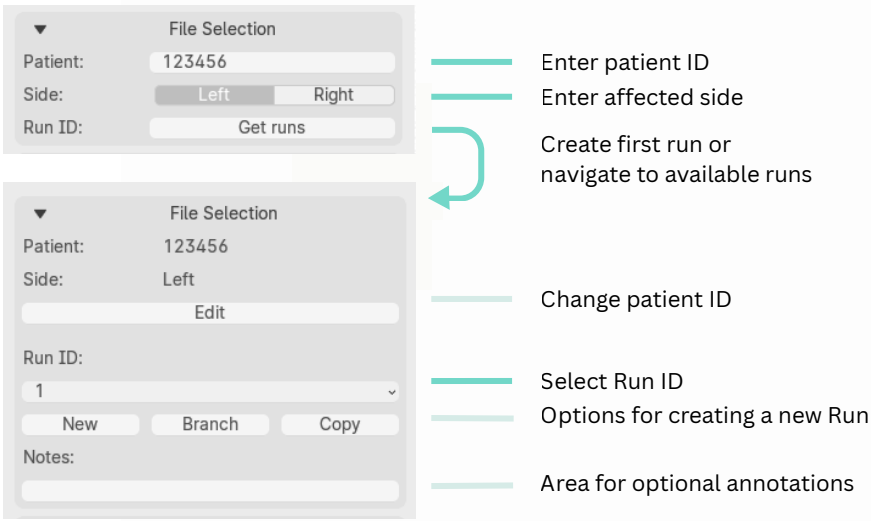
Section 2 – General Controls

File Selection

After entering the patient ID, press Enter. Now the affected side can be selected, and the runs can be retrieved. If no run is available, a new run ID is created.

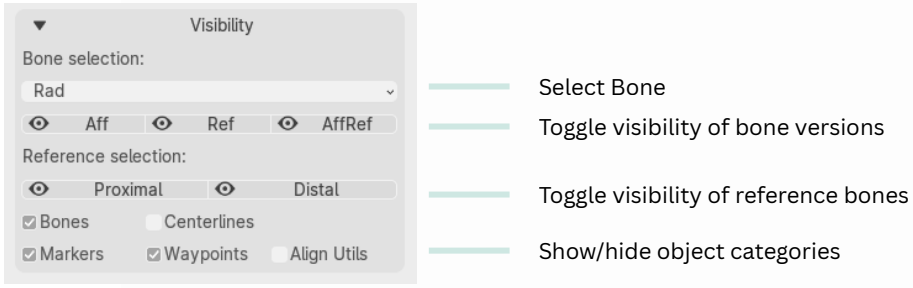
After completing all steps up to the optimization, the optimization can be started for one bone. This will present nine options. If you wish to adjust settings or optimize the other bone, a new run ID can be created using the functions described above. This keeps the settings for each run separate and allows for easy navigation between runs to select the best solution.

- New – starts a completely blank file with no previous data.
- Branch – creates a new file that keeps all objects and progress up to the current visible panel, allowing you to continue from there without losing earlier work.
- Copy – creates an exact duplicate of the current file, including all objects, settings, and changes, allowing you to experiment without affecting the original.



Visibility

The user can control object visibility via the Visibility panel. Bone type can be selected, and the affected (aff), reference (ref), or aligned affected-reference (affref) bones can be shown or hidden. Two copies of the reference bone, aligned to the proximal or distal side, can be hidden individually. All objects can also be hidden by category.



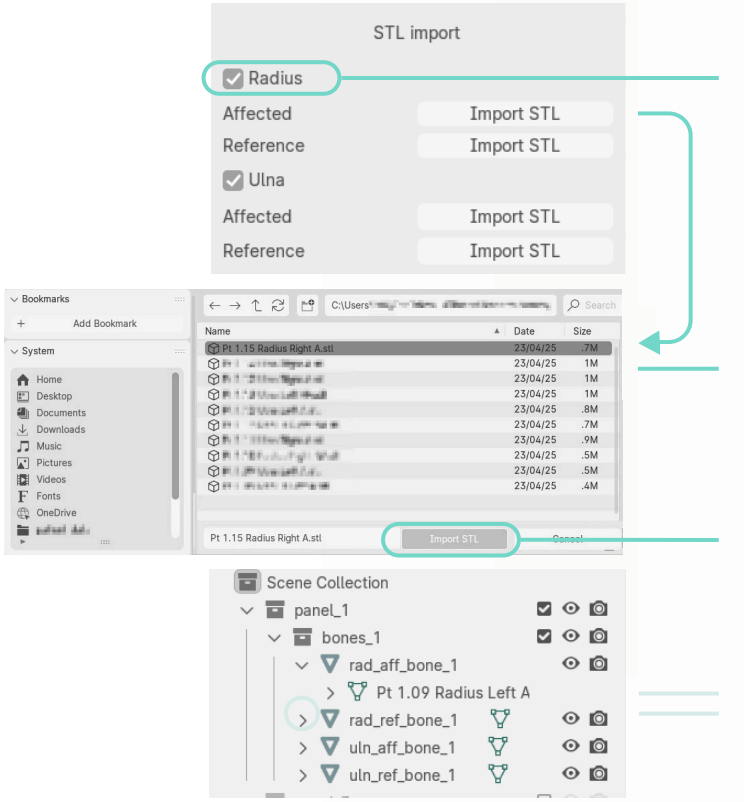
Transparency

The bones can be made semi-transparent by using the toggles at the Transparency panel

Section 3 – Panel Workflow

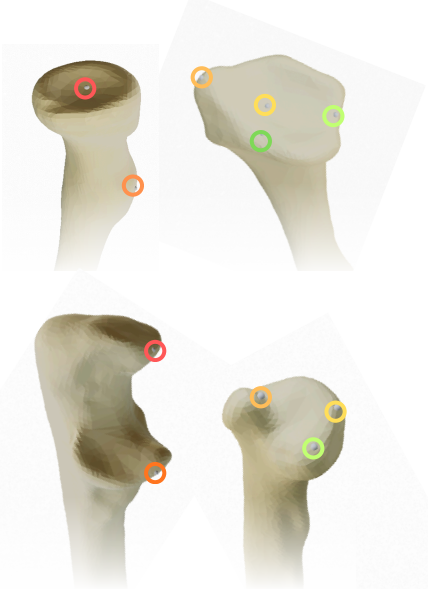
1. STL import

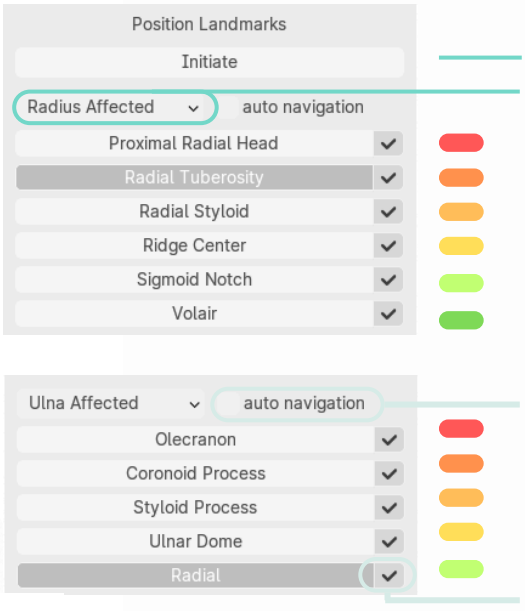
Bone models can be imported in STL format. By using the tick-fields, the user can choose wheter to optimise the radius, ulna, or both.



2. Position Landmarks

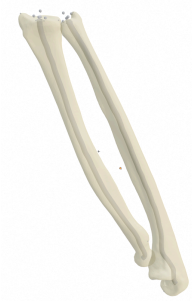
After initialising, a bone can be selected through the UI. Navigate to the correct position on the bone before pressing the corresponding marker button. Press the location on the bone. If desired, this placement can be cancelled by using ESC. When a marker has been positioned the indicator will be ticked. Press this to remove the marker.

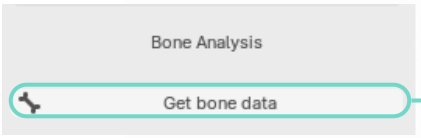




3. Bone Analysis

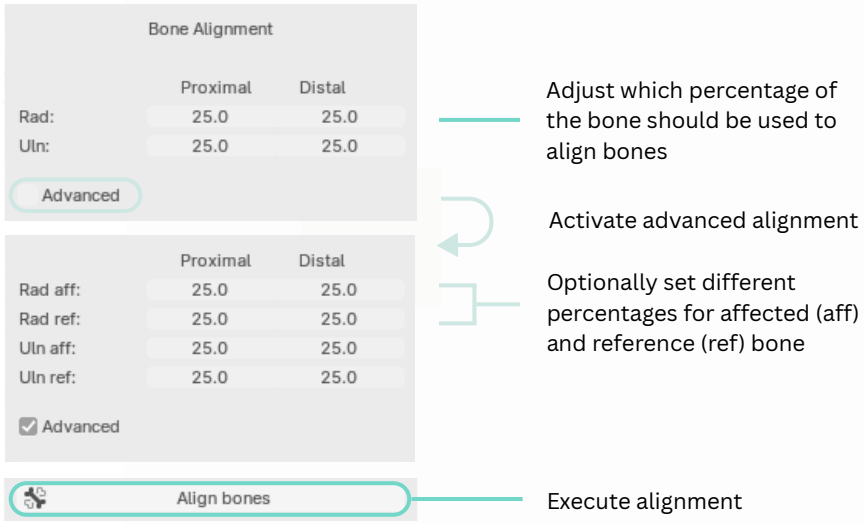
After retrieving the bone data, the user should verify that the centerline has been created successfully.
Error-handling: If the centerline has not been created successfully, reposition the landmarks so that the Ridge Center and the midpoint between the ulnar dome and the styloid process accurately describe the bone center.





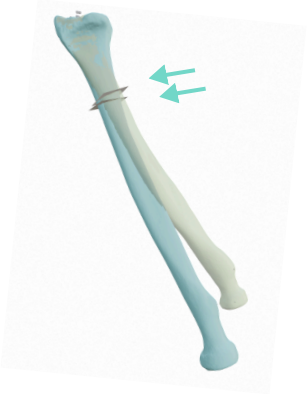
4. Bone Alignment

Mirroring and distal–proximal alignment of the reference bone (blue) to the affected bone (yellow) are performed automatically using the Align Bones function. Afterwards, the user can refine the alignment region by adjusting the percentage range, which is visualised through interactive planes, and then execute the alignment again.
Advanced: For bones with great difference in scale, the advanced option can be used to select different percentages for both the affected as the reference bone.



5. Template Alignment

A copy of the affected bone (green) is aligned to the distal reference bone (blue), representing the desired final alignment of the distal bone part.
See Panel 4: Bone Alignment

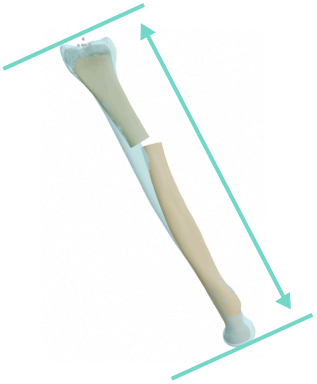


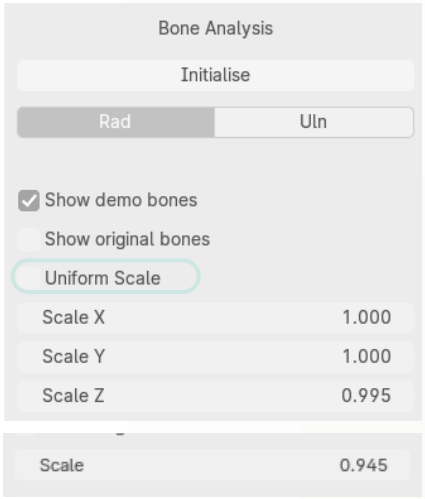
6. Scaling

After initialization, an interactive visualization of the final osteotomy is presented to help estimate a suitable scaling factor for the reference bone. To visualize how the final planning would appear with the selected scaling factor, the user can modify the osteotomy height through the *Plane Bounds* subpanel and adjust the distal fragment alignment via the *Distal Bounds* subpanel. Using these components in an alternating manner allows for stepwise fine-tuning of the configuration.

6.1 Reference scale

The user can adjust the reference bone's dimensions along the X, Y, and Z axes, or apply uniform scaling.





Initialise interactive enironment

Select bone of interest

Show or hide the bones providing an estimate of the planning

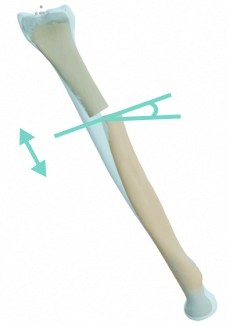
Show or hide the original bones

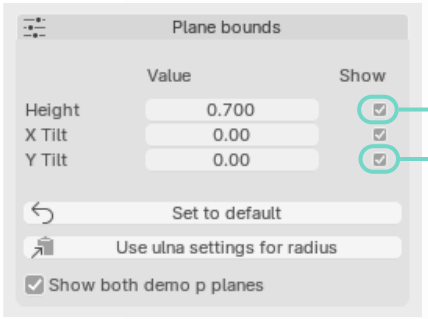
If enabled, the same scale will be used for the X, Y and Z axis.

Adjust scale of the reference bone

6.2 Osteotomy orientation

Enable options to adjust preview osteotomy orientation





Enable osteotomy preview

Apply osteotomy tilt

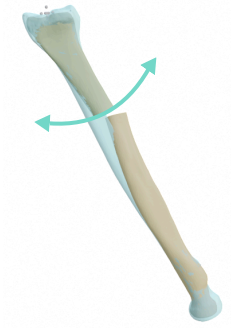
Set settings to default

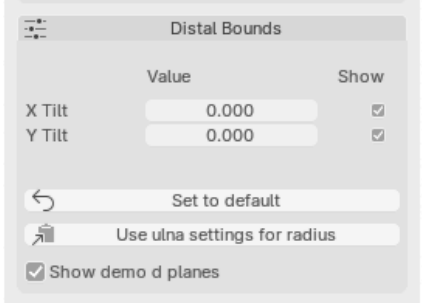
Use the settings as the other bone

Hide plane objects used for alignment

6.2 Distal Alignment

Enable options to adjust preview orientation of distal bone fragment





7. Bone Alignment

Realign the (scaled) reference bones. See 4. Bone alignment

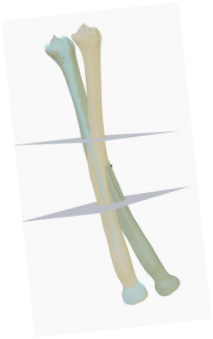
8. Template Alignment

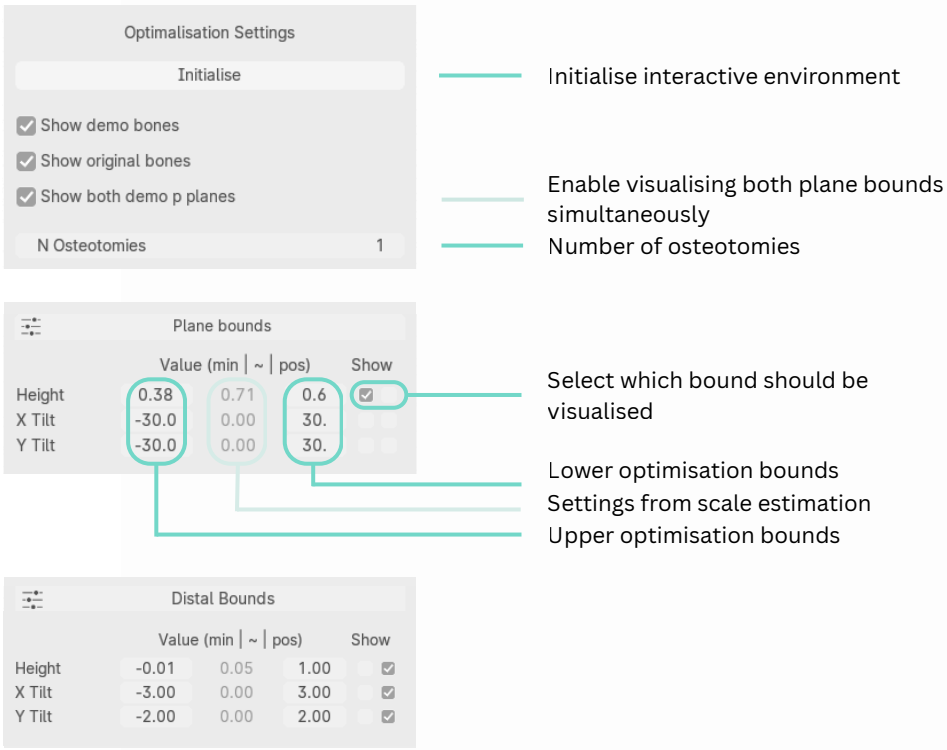
Realign the affected bone to the (scaled) distal reference bone. See 5. Template alignment

9. Optimisation Settings

9.1 General bounds

After initialization, an interactive visualization is presented to adjust the lower and upper bounds in which the optimisation algorithm may search for solutions. The parameter settings as set during the determination of the scaling factor are presented as well as guidance.
Tip: The user could initially restrict the distal bounds, after which they may be set to bigger interval in case the result does not lead to desired result.



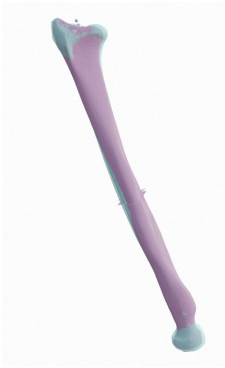


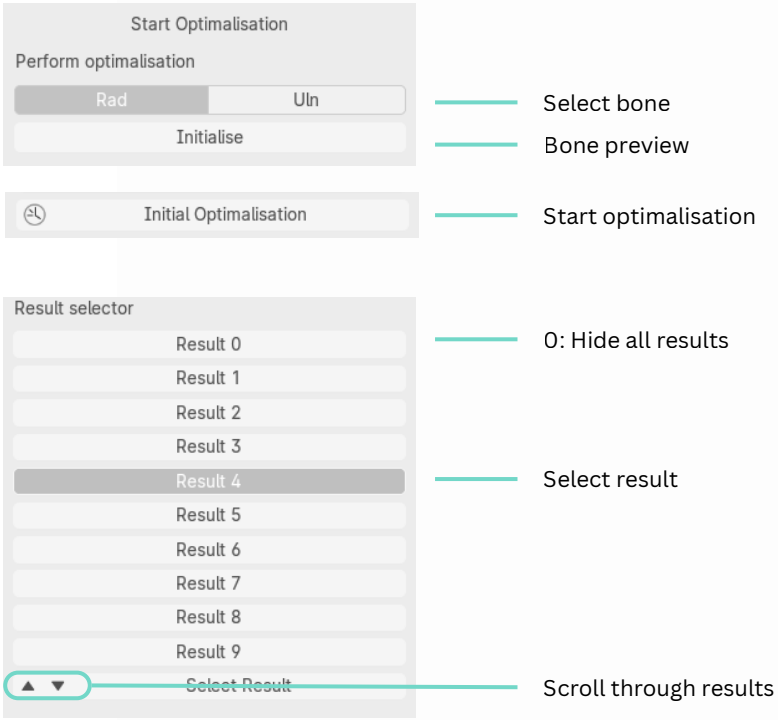
9.2 Hyperparameters

Hyperparameters can be adjusted to suit the users intention. Smaller population size and number of generations leads to shorter run time, but may not yield in optimal results.
Error handling: In case the memory usage limits CU performance, the number of CPUs to process the optimisation iterations can be reduced by the user

10. Optimalisation

After selecting the bone to be optimised at this run file, the user can initialise to view the bones, and start optimisation by using initial optimalisation. A spectrum of results is presented from optimised centerline correspondence to optimised cut osteotomy surface alignment.





Part III:
Technical document

This technical document presents the development process and technical aspects of the semi-automated forearm osteotomy planning tool. It builds upon the clinical and scientific context presented in the validation paper and examines the challenges encountered and design choices made during its creation.

6
Introduction

6.1. General Introduction

Forearm deformities can significantly affect both the function and appearance of the limb. [8] [12] [6] Corrective osteotomies are surgical procedures aimed at realigning the bones by making precise cuts and repositioning segments, followed by fixation with osteosynthesis plates. [7] Advances in medical imaging and Three-Dimensional (3D) modelling have enabled more optimised surgical planning, yet manual planning remains time-consuming and subject to variability. The time investment is especially high, since a process of repetitive manual iterations is required to find the optimal configuration. [1].

Currently, the process of creating preoperative configurations for forearm osteotomies involves multiple manual iterations of simulating osteotomy cuts and repositioning bone fragments. This labour-intensive process forces clinicians to make trade-offs between the time spent on planning and further optimising the surgical plan. There is a clear need for methods that improve planning efficiency without compromising accuracy. Developing a robust semi-automated planning tool for correction of forearm malalignment involves addressing challenges related to variability in imaging data, computational efficiency, user-interaction design, and integration with clinical workflows.

6.2. Project goal and subgoals:

6.2.1. Statement:

The research goal can be stated as follows:

"To develop an open-source semi-automated surgical planning tool for forearm osteotomies, improving surgical planning efficiency."

6.2.2. Subgoals

The following sub-goals break down the project into key focus areas, each supporting a specific part of the tool's development. They are accompanied by a selection of guiding principles that reflect the good practices to be upheld throughout the process.

1. Translate clinical needs into algorithm design to generate feasible osteotomy plans.

   Develop a tool for creating viable surgical plans addressing key surgical considerations, while ensuring patient safety, regulatory compliance, and adherence to ethical principles throughout algorithm development.

2. Designing an intuitive, user-centered interaction workflow.

   Design guided by the workflow and preferences of preoperative planners and clinical staff, utilising User Interface (UI) and Uniform Crossover (UX) principles for efficiency, simplicity, and focus to support effective use.

3. Develop a backend infrastructure designed to support future implementation.

   Adhere to best practices in system architecture, database management, API design, and security to ensure robustness and scalability.

4. Establish a code architecture that ensures maintainability and extensibility.

   Emphasise modularity, documentation, error handling, and quality assurance to ensure maintainable and high-quality software.

5. Evaluate the potential to improve osteotomy surgical plans and its usability.

   Apply evidence-based methods and ensure reproducibility, transparency, and ethical data use throughout the evaluation.

6.2.3. Related Work

Previous research conducted at Erasmus MC, University Medical Center Rotterdam, by F. Koopman et al.(2022), Focused on the semi-automation of forearm osteotomy planning using in-house developed optimisation algorithms implemented within 3-Matic (Materialise, Leuven).[3] This commercially available medical Computer-Aided Design (CAD) software is widely used for creating osteotomy plans and offers scripting capabilities via a Python Application Programming Interface (API), enabling partial automation of planning tasks [19]. However, reliance on commercial software limits both accessibility and flexibility, emphasising the need for open-source alternatives. Although the in-house optimisation algorithms did not meet performance requirements, the project provided valuable insight into surgical needs and identified key bottlenecks in automating forearm osteotomy plans.

A foundational research project for this work was conducted at the LUMC by Dikland et al. (2023). A proof-of-concept was developed using the pymoo multi-objective optimisation framework and the open-source Blender software (Blender Foundation, Amsterdam, The Netherlands). Although the codebase was not fully suitable for reuse, this work delivers valuable insights into the implementation of several key aspects: how the osteotomy procedure can be mathematically defined, which measures are appropriate for evaluating shape dissimilarity, and how parameters can be optimised to minimise the difference between the post-operative bone and the target shape [18].

6.2.4. Technological Context

In line with similar initiatives at the LUMC, this project focuses on developing an open-source solution using the Python API of Blender. This project employs Blender and Pymoo as core tools. Blender, an open-source 3D modelling platform, enables flexible visualisation and user interface customisation through its Python API. [25] Pymoo is a multi-objective optimisation library in Python, suitable for exploring complex parameter spaces, such as osteotomy plane height and tilt under various constraints. [20] Its ability to balance multiple competing objectives makes it well-suited for osteotomy planning.

6.2.5. Document Outline

This document is structured as follows. It begins with a user needs assessment, followed by a description of the implementation of each planning phase. Key decisions regarding the problem definition and algorithm are then clarified, and the document concludes with an overview of the data management strategies.

Manually creating preoperative osteotomy plans for the forearm is a labour-intensive process, often requiring multiple iterations of adjusting osteotomy planes and repositioning bone fragments. This thesis proposes a semi-automated, open-source planning tool designed to optimise the workflow. The tool is developed in Blender using its Python API, and uses the Pymoo library for multi-objective optimisation. The project prioritises alignment with the clinical requirements, usability, and a modular code architecture. Building on prior research at Erasmus MC and LUMC, the project aims to provide a reusable and flexible alternative to commercial tools, incorporating new functionalities to better support osteotomy planning.

7 Methods

The project was divided into six phases. This structured approach allowed the functionalities to be tailored to forearm planning, to leverage insights from existing research, and to support the design of a modular and extensible foundation.

Phase I: Conceptual foundation

This phase focused on understanding the specific requirements of forearm planning and exploring opportunities within the optimisation framework, data handling, and system architecture. This facilitated preparation for future enhancements and supported the identification of limitations in the codebase..

Phase II: Codebase dissection

The second phase focused on a detailed analysis of the original codebase by Dikland et al. (2023). The code was dissected to clarify data flow and computational logic, then restructured into modular, self-contained functions wherever possible. The functions were classified according to reusability: (1) those directly reusable with minor adjustments, (2) those serving as conceptual inspiration, and (3) those deemed irrelevant or unsuitable for further development. When functional boundaries were unclear due to intertwined logic, code segments were reorganised and renamed to improve clarity, traceability and alignment with the new implementation. Although functionality was temporarily sacrificed for structural clarity, this systematic dissection enabled efficient navigation through the codebase and produced reference code better aligned with the project's needs.

Phase III: Architecture Design

Guided by earlier insights, a new architecture was designed with clear separation of concerns, consistent naming conventions, and structured data handling. A pseudocode-based blueprint outlined the purpose and dependencies of each module, specifying whether the logic was derived from the original codebase or developed anew. Where appropriate, functions with similar purposes were consolidated using conditional statements, or shared logic was encapsulated within separate functions.

Phase IV: Incremental integration

Newly developed and codebase-inspired components were integrated gradually, ensuring compatibility with the redesigned inputs, outputs, and workflows.

Phase V: Iterative development

This phase focused on refining both functional and non-functional aspects of the software. Features were progressively improved based on internal testing and feedback from clinical users.

Phase VI: Evaluation

The final phase assessed the performance and usability of the newly developed planning tool.

8

User needs assessment

8.1. Stakeholders

Table 8.1 provides an overview of key stakeholders involved in the osteotomy planning add-on, including their roles, interests, and potential impact on the project.

Stakeholder	Role	Interest	Influence	Impact on project
Surgical planners	Key users, including technical physicians and biomedical engineers	H	H	Influence in defining functional requirements and ensuring workflow integration for usability.
Orthopedic surgeons	End-users who rely on the tool for accuracy and reliability	H	H	Insights shape clinical feasibility and patient safety requirements, crucial for practical use in surgeries.
Hospital	Contributes to funding and will need to invest in final development.	M	M	Interested in time efficiency, cost-effectiveness, and potential for scientific publications. Supports wider implementation.
Patients	Beneficiaries of improved surgical outcome	M	L	Satisfaction impacts tool adoption indirectly
Medical researches	Individuals in orthopedic research potentially using the method as a foundation for further studies	M	L	Interest in well-documented code; contributions aid validation and improvement; findings can influence community adoption.
University	Contributing to funding.	L	L	Expects research to meet scientific standards, ensuring code quality and best practices in development.

Table 8.1: Stakeholder analysis for the osteotomy planning addon. *Abbreviations:* H, High; M, Medium; L, Low.

8.2. Workflow Analysis

The current workflow for creating forearm osteotomy plans can be described as follows: First, a suitable correction template is established, often by mirroring and scaling a model of the contralateral bone. Next, this reference bone is aligned with the model of the deformed bone based on a selected proximal region, which forms the correction template. A second model is aligned distally to assist in estimating the Center of Rotation of Angulation (CORA). The CORA represents the site in a deformed bone where a corrective osteotomy is generally performed to realign the deformity. The bone model is cut, and the distal bone segment is aligned with the distal side of the correction template. The distal segment can then be reoriented, allowing slight deviations from this distal alignment to meet all clinical objectives.

This process may be repeated for multiple osteotomy positions, after which intermediate positions are evaluated based on their optimised orientations.

8.3. Risk Management

Table 8.2 summarises the potential risks, their estimated impact and likelihood, and the corresponding mitigation strategies.

Risk	Imp	Freq	Mitigation
Mismatch between STL files and patient ID	H	M	Require patient ID before STL upload; alert on mismatches.
Poor handling of large STL files	H	M	Reduce file size without loss of accuracy.
Data privacy concerns	H	L	Use ID-based filenames; enable manual file deletion.
User errors	M	M	Apply input validation and concise, clear guidance.
Data loss	M	M	Enable backups and recovery options.
Over-reliance on automation	M	M	Provide tools to assess and verify results.
Case diversity issues	L	M	Validate using broad case testing and manual review.
Bone misidentification	L	M	Visual aids and clear labeling during file selection.
Excessive bone length changes	L	M	Compare output to both scaled and original STL.
Accidental disabling of constraints	M	L	Emphasize constraint status and list inputs in report.

Table 8.2: Stakeholder analysis for the osteotomy planning addon. *Abbreviations:* Imp, Impact; Freq, Frequency; H, High; M, Medium; L, Low; STL, Standard Triangle Language; ID, Identification; UI, User Interface

8.4. Software Requirements

Table 1 and Table 2 provide an overview of the identified functional and non-functional requirements, respectively. The requirements are prioritised by using to the MoSCoW classification[15], which distinguishes the following categories:

M	S	C	W
Requirements that are necessary for the completion of the project.	High-priority requirements that should be included, if possible.	Requirements that are nice-to-have, but could be left out.	Requirements that will not be implemented at this stage of the project.

Figure 8.1: MoSCoW requirement categories with icons and definitions.

Functional requirements are outlined in the Appendix, categorised according to the MoSCoW method. These requirements were identified at the project's inception, based on surgical needs and potential risks, which serve as key drivers for development. The add-on should facilitate the workflow by enabling the use of the contralateral bone as a correction template and by incorporating correction objectives and surgical constraints. Non-functional requirements address aspects related to the usability of the workflow and the maintainability of the system for future development.

The osteotomy planning tool engages multiple stakeholders and prioritises clinical requirements and user-friendliness. Key risks such as data errors and user mistakes are mitigated through validation and warning mechanisms. Core features enable the generation of viable pre- and postoperative forearm osteotomy plans, with options for manual adjustments. The software must have an intuitive interface, support efficient data handling and robust recovery, and be maintainable through clear code structure and sufficient documentation.

9

Optimisation Strategies

9.1. Introduction to Optimisation

To enable automatic optimisation of surgical plans, osteotomies must be defined using decision variables. These variables are interpreted to determine the height and tilt of planes that cut a model bone, while others specify how the bone is subsequently repositioned. As described by Dikland et al. (2023), the osteotomy cut and displacement can be represented using two geometric planes along the bone's centerline, providing a stable anatomical reference for specifying osteotomy locations via clinically relevant height and tilt parameters.

Figure 9.1 illustrates the concept of a search space in osteotomy planning using two variables, such as the heights of both planes. The search space comprises all possible combinations of these variables, with bounds setting the minimum and maximum values and additional constraints excluding surgically infeasible combinations. Only the allowed regions of the search space are explored for optimal solutions.

Optimisation requires evaluating the quality of each variable combination, achieved via an objective function, a mathematical scoring function quantifying how well a solution meets the desired criteria, such as the distance between a point on the affected bone and its reference. Visualising scores across all combinations produces a conceptual landscape, where regions of lowest scores, the global minima, correspond to optimal surgical plans.

Exhaustive evaluation of every combination is computationally infeasible, therefore, an optimisation algorithm efficiently explores the search space. It typically begins by sampling a set of combinations, evaluating their scores, and iteratively selecting new combinations based on the most successful results. Depending on the chosen algorithm, priority may be given to speed, rapidly identifying low-scoring regions, or reliability, thoroughly exploring the search space to ensure the best solution is found. [17]

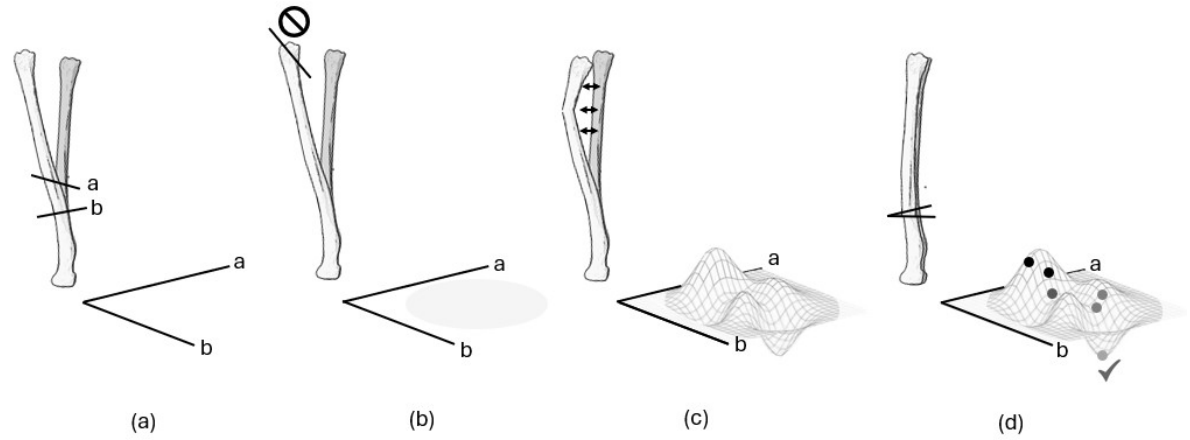


Figure 9.1: Schematic overview of optimisation concepts, illustrating their effect on surgical plan evaluation and selection. (a) Parameters defining the search space, (b) Constraints restricting the feasible region of the search space, (c) an Objective function that assigns quality scores to each variable combination, and (d) an Algorithm that samples and evaluates points in the search space to identify the global minimum.

9.2. Applied Optimisation Strategies

At the start of the project, a clear set of optimisation strategies was identified to guide the design and implementation of the tool.

- Strategy A: Rapid Optimalisation
  Because evaluating all combinations of variables is computationally expensive, optimisation algorithms iteratively sample the search space. They focus on promising regions but may overlook better solutions elsewhere, potentially converging on a local rather than a global minimum. This project prioritises providing rapid results to avoid disrupting the user's workflow. Since not all clinical considerations can be fully captured by objective functions, even the global minimum may be suboptimal in practice. By generating quick solutions, the optimisation settings can be manually refined and repeated as needed.
- Strategy B: Informed Initialisation
  This project aimed to make optimal use of prior knowledge of the parameters, such as plane height estimated from the CORA, defined by a user-selected region, or deduced from the desired final alignment of bone fragments. Incorporating this information reduces time spent exploring non-viable regions of the search space.
- Strategy C: Human-in-the-Loop
  To optimally align outcomes with clinical judgment, this project aims to involve the user in the optimisation process, for instance by allowing the adjustment of constraints.
- Strategy D: Multi-Solution
  A Pareto front represents the set of all optimal trade-off solutions in a multi-objective optimization problem. This project aimed to provide multiple solutions along the Pareto front, each representing a distinct trade-off between competing objectives.This enables the user to select a solution based on specific preferences or clinical requirements. Since evaluating subtle differences between solutions may be time-consuming, the number of results should be tuned to the user's needs.

> An optimisation algorithm seeks to minimise a mathematical scoring function, called the objective function, by exploring different combinations of decision variables within the search space. These decision variables are translated into osteotomy planes and bone segment positions by interpreters. Constraints exclude solutions that are not clinically viable. Various optimisation strategies have been considered, and this project focuses on a fast algorithm that benefits from prior knowledge about viable osteotomy heigth, involving the user extensively in the planning process and offering multiple solutions.

10

Object Preparation

10.1. Correction template

In 3D osteotomy preoperative planning, a corrective template guides bone realignment to restore forearm function. Given the close relationship between the two forearm bones, a mirrored model of the contralateral bone is generally preferred, as it provides patient-specific, anatomically realistic alignment.[5] In cases of bilateral deformity, a Statistical Shape Model (SSM) offers an alternative. SSMs are geometric models representing the average shape of a population along with its natural variation.[2] Models simulating forearm rotation can further validate functional restoration [13], though using them as the primary reconstruction target is computationally intensive.

10.2. Orientation

Bone orientation can be defined using user-selected anatomical landmarks. In this project, the workflow prompts the user to manually identify the landmarks. This process can later be complemented by initial automated positioning methods developed within this research group [22].

For the radius, the x-axis runs between the radial and ulnar landmarks, the z-axis is perpendicular to this line and represents the long bone axis, and the y-axis corresponds to the volar-dorsal direction. For the ulna, the distal coordinate system is defined similarly, with the x-axis running between the styloid process and the ulnar dome.

To create a suitable reference bone, the unaffected contralateral side is always mirrored along its medial–lateral axis, independent of its orientation within the software's coordinate system. This orientation may vary due to differences in scanning positions or preprocessing steps. Anatomical landmarks are used to define local coordinate systems representing the bone's anatomical axes. These axes are used to align the bone relative to the global coordinate system, enabling accurate mirroring of the bone.

10.3. Superimposing

The addon first aligns the bones along their local coordinate system to establish a robust prealignment, then applies landmark-based alignment using Singular Value Decomposition (SVD) which is then refined using Iterative Closest Point (ICP). Users can manually adjust the bone regions via the UI to ensure precise registration. Various methods have been considered to achieve this alignment, including:

- Local coordinate system Aligning bones along clinically relevant axes but does not ensure detailed surface matching.
- Stepwise alignment stages Bones are first aligned using one line between landmarks and then the alignment is refined stepwise using additional lines. Each step uses Rodrigues' rotation: the cross product of corresponding lines defines the rotation axis, and the dot product determines the amount of rotation. This method provides control over aligning specific, clinically relevant axes, even when the objects differ in shape, such as when aligning a bone to an implant. However, it does not ensure detailed surface matching. [14]
- Open3D toolbox Open-source 3D data processing library that includes multiple 3D image registration options. However, this library is not supported in the latest Python versions. [24]

- SVD Determines the optimal rotation and translation for alignment. After centering the data at the origin, a covariance matrix is decomposed using SVD. U and V define the main axes of the data, while Σ indicates their correspondence. Since we do not want to scale the affected bone, Σ is not used. Multiplying by $V^\top$ rotates the bone into a temporary coordinate system, and multiplying by U aligns it with the reference. The final rotation is $R = VU^\top$. [9]
- ICP Iteratively minimizes distances between corresponding points to improve registration, but can be computationally intensive and it may fail to find the best alignment if the starting position is far from correct.[16].

10.4. Centerline

The shaft's centerline is approximated using an iterative method inspired by Dikland et al. (2023). By replacing the generation of sphere objects for repeated intersection with the bone shaft with a mathematical representation of a plane, the method has become considerably more efficient.

The software uses a mirrored contralateral bone as a correction template. This reference bone is mirrored according to a landmark-based local coordinate system, after which it is registered onto the affected bone using SVD and ICP. To guide the generation of the osteotomy locations, the centerline of the shaft is estimated by a technique based on iteratively intersecting the bone.

11

Interpreter

11.1. Introduction

The optimisation process systematically explores combinations of variables, referred to as optimisation parameters. The Interpreter Module translates these parameters into planes that define how the bone should be cut and repositioned. Each resulting configuration is evaluated, and its score guides the algorithm in selecting the next parameter set. Through this iterative process, the solution converges toward an optimal outcome [17].

11.2. From planes to corrections

The plane interpreters convert optimisation parameters into a representation of the proposed osteotomies. Each osteotomy set is defined by two planes that together describe both the locations to cut the bone and the intended repositioning of the distal bone segment. Plane A defines where the bone is cut. Plane B defines the target position of the distal bone segment after correction. Building on the implementation by Dikland et al. (2023), this method is extended to not only allow closing osteotomies, but also opening and hybrid osteotomies. When Plane B is distal to Plane A along the centerline, the distal segment moves outward, producing an opening osteotomy. When Plane B is proximal, the segment is removed, producing a closing osteotomy. AIn a hybrid osteotomy, part of the bone closes while another part opens. This occurs where Plane A is distal to B in some regions and where Plane B is distal to A in others.

11.3. From parameters to planes

A new method has been developed that leverages prior knowledge of the desired distal alignment. This approach is applicable only to the most distal osteotomy. To accommodate cases with multiple osteotomies, a strategy has been designed in which parameters are interpreted differently depending on whether they define the most distal correction. All parameters, except for the final set, are processed by the general plane interpreter, while the final set is handled by the final plane interpreter.

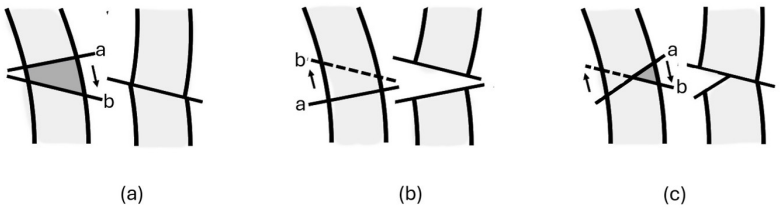


Figure 11.1: Illustration of how the order of planes can define correction osteotomies, with (a) a closing osteotomy, (b) an opening osteotomy and (c) a hybrid osteotomy.

11.3.1. General plane interpreter

Using three parameters, Plane A is defined by its relative height along the bone's centerline and its x- and y-axis tilts. This interpreter defines Plane B similarly, but additionally incorporates two parameters describing translation perpendicular to the bone's local axis and one parameter specifying rotation around the plane's normal axis. Together, these nine parameters fully define how the bone is cut and how the distal segment can be translated and rotated in all directions.

11.3.2. Final osteotomy interpreter

The final osteotomy interpreter determines the positioning of the distal fragment differently, as illustrated in Figure 11.2. Before the optimisation begins, a copy of the affected bone is aligned to the distal side of the reference bone. Even though the osteotomy cut itself is not yet generated, this allows the user to control how the distal bone segment will ultimately be positioned. This aligned model is referred to as the affected-reference (aff-ref) model. Using the parameters of the first plane (height, x-tilt, y-tilt), a corresponding plane is generated on both the affected bone and the aff-ref bone. In this setup, the affected bone retains the proximal segment, while the aff-ref bone retains the distal segment. The displacement between these two planes defines the translation and rotation of the distal bone segment, effectively specifying both Plane A and Plane B.

This approach is limited to a single specific distal alignment. However, unlike the general interpreter, where plane B is specified using an additional six parameters (height, x-tilt, y-tilt, x-translation, y-translation, rotation), plane B is now already defined based on a known position of the centerline and the first three parameters. By interpreting the remaining six parameters differently, multiple distal alignment options can be generated. The key advantage of this technique over the general osteotomy interpreter is that it provides explicit control over the extent of allowed deviations.

With the coordinate systems aligned to clinically relevant directions, the bounds of variable deviations can be independently adjusted for each axis, enabling more controlled and precise distal alignment during optimisation. This is where we introduce new plane concepts: the initial distal plane (D_0) and the distal plane (D). The initial plane D_0 is aligned with the distal axis. Plane D deviates from this initial plane D_0 based on optimisation parameters, similar to how Plane B is defined in the regular osteotomy interpreter. Rather than being defined along the centerline, Plane D is specified using the distal coordinate system, with a positive height indicating distal translation along the z-axis. Additionally, the plane can be adjusted through translations or tilts along the x and y axes in the defined directions, or rotation around the z-axis. The transformation from D_0 to D is applied to the aff-ref bone, introducing controlled deviations from the ideal distal position.

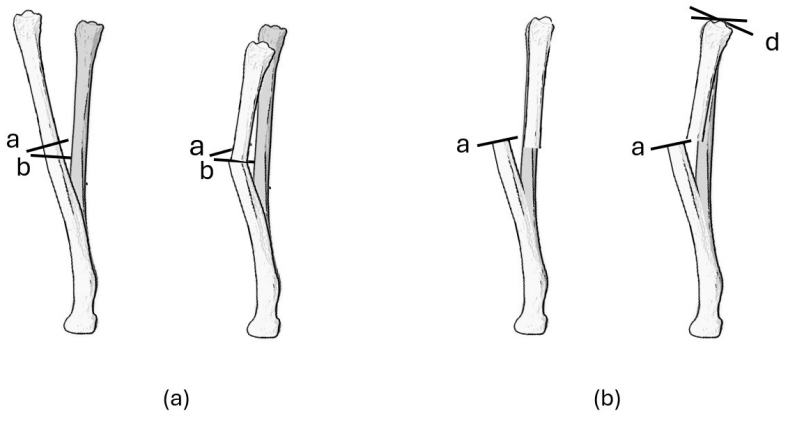


Figure 11.2: Illustration of the interpreters: (a) General interpreter – creates plane A to cut the affected bone and transforms the distal fragment to plane B. (b) Final interpreter – creates plane A to cut the affected bone and cuts the distally aligned bone at the same relative height; plane D is used to adjust the alignment of the distal fragment.

11.3.3. Implementation

Since only the most distal bone segment can be aligned before optimisation begins, the final osteotomy interpreter is applied exclusively to the last parameter set of nine variables. When two or more osteotomies are required, the first parameter sets are interpreted using the general osteotomy interpreter, which directly defines Planes A and B. Before the final osteotomy interpreter can be applied, the bone segments must be repositioned according to the more proximal osteotomies. Figure **??** provides a schematic overview of the entire workflow of dubbel-level osteotomies.

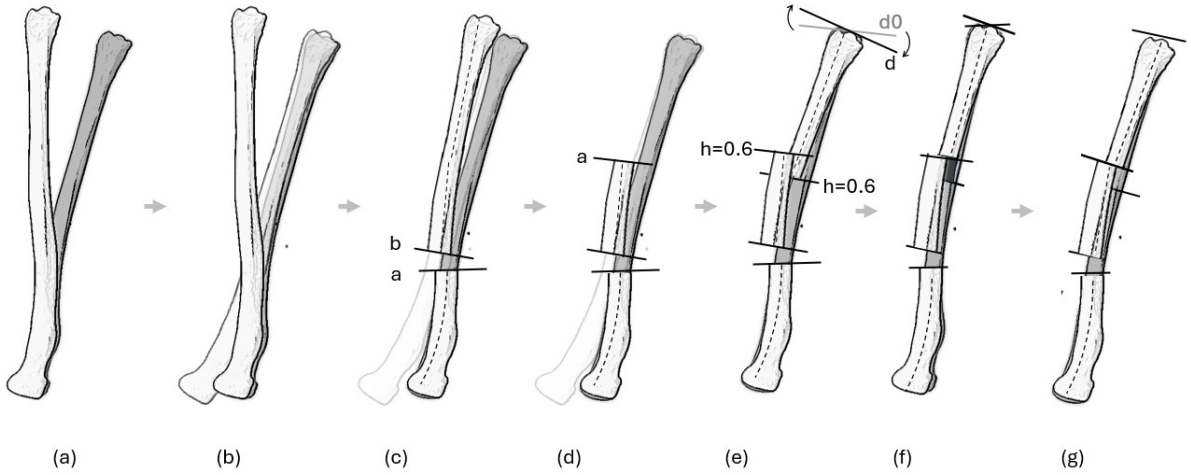


Figure 11.3: Implementation of both interpreters: (a) Align the reference bone to the proximal side of the affected bone. (b) Align a duplicate of the affected bone to the distal side of the reference bone. (c) Use the general osteotomy interpreter to cut and transform a bone part based on planes A and B. (d) Use the final osteotomy interpreter to cut the bone based on plane A and the distally aligned bone at the same relative height; plane D is used to adjust distal alignment. (e) Remove overlapping bone parts if present. (f) A double-level osteotomy is created with secured distal alignment.

11.4. Reversed Osteotomy Interpreter

During corrective surgery using PSI, all osteotomies may be cut before any realignment takes place. Therefore, the preoperative model must show the cuts on the original bone. To do this, planes are reversed along with bone segments, starting from the most distal osteotomy, by applying the inverse transformation matrices. Once all planes are reversed, the original osteotomy positions are obtained.

The Interpreter module converts optimisation parameters into bone cutting planes and transformations, describing each osteotomy with nine parameters. The Regular Interpreter defines cuts and distal bone repositioning, while the Final Interpreter uses a distally aligned reference model to ensure joint alignment and allow precise clinical adjustments. Transformations can be reversed to determine the original bone's cut locations for surgical planning.

12

Objective function

12.1. Objective function

Objectives define the goals that an optimisation algorithm aims to achieve, such as minimising bone misalignment or maximizing joint congruency. Metrics are the specific measures used to quantify performance relative to these objectives. When a metric is applied to a particular plan, it produces a dissimilarity score, indicating the degree to which the plan meets, or deviates from, the objective.

A mathematical scoring function quantifies how closely the osteotomised bone matches the reference. This can include minimizing surface-to-surface distances, maximizing volume overlap, penalizing protrusions, or prioritizing alignment of clinically critical regions.

In line with the Rapid Optimisation Strategy, 3D-SurgiGen focuses on efficient metrics, using the bone centerline rather than the entire mesh to describe overall correspondence with the reference bone. This study has achieved an effective method for describing osteotomy contact surfaces, as visualised in Figure 12.1 Instead of transforming the entire mesh to the post-operative position, the positioning of the cutting planes in pre-operative configuration is used to intersect with the mesh, and then transform only these intersections to the post-operative configuration. After resampling these points describing the outer border of osteotomy surfaces are used to optimise the acquisition of a continuous bone surface, suitable for osteotomy plate positioning.

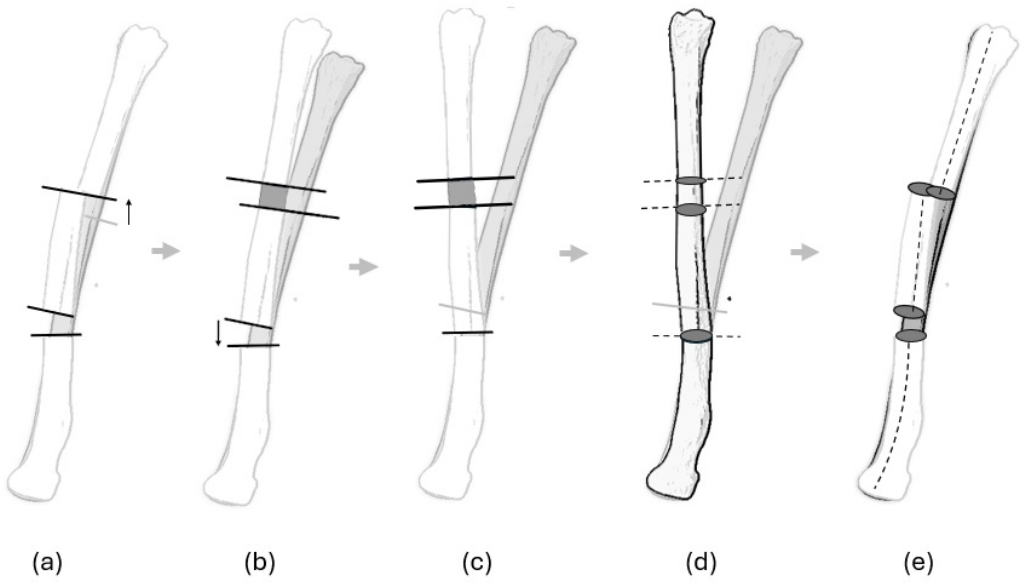


Figure 12.1: Extracting osteotomy surface borders to serve as objectives. (a) Post-operative configuration. (b) Undoing the most distal osteotomy (closing osteotomy). (c) Undoing the proximal osteotomy (opening osteotomy). (d) Representing osteotomy surface borders via intersection. (e) Retransforming planes and osteotomy border representations back to the post-operative configuration.

To manage trade-offs between multiple goals, the objective function can use scalarization by assigning weights to the individual scores to reflect their clinical importance. Alternatively, a multi-objective formulation can be applied, defining two or more objective functions to identify the Pareto front. The Pareto front illustrates how improving one objective affects the others, and the solutions on this front provide a spectrum of options, allowing the user to select a plan that best fits clinical priorities.[20][10] Although multi-objective optimisation is generally less computationally efficient than applying a single weighted objective function, it offers greater flexibility in creating a plan tailored to clinical preferences.[10]

12.2. Dissimilarity Score

A dissimilarity score quantifies the overall misalignment. Metrics can emphasise or ignore outliers, and may be weighted to capture different aspects of bone alignment.

When selecting suitable metrics, it is important to consider their influence on optimisation behaviour. Metrics that are highly sensitive to small misalignments can cause premature convergence into local minima. Using robust shape-based metrics, capping extreme deviations, or starting with coarse alignments before refinement can help prevent this.

In this work, a combination of the Hausdorff and mean distances is used as a starting point, capturing both maximum local deviations and average surface discrepancies [11]. Further research could identify the most effective metrics or combinations for capturing both the overall bone shape using the centerline and the osteotomy cut surfaces using the osteotomy borders. An optional strategy is to give users control over the weighting of these metrics, as minor protrusions can often be corrected surgically.

12.3. Constraints

Constraints are generally handled either as strict requirements that solutions must satisfy or by incorporating them into the objective function through penalty terms. In this work, however, feasibility is primarily ensured through variable bounds repair operators. Variable bounds prevent unrealistic positions or orientations, used for enforcing user-defined limits on the divergence of distal part alignment. After candidate solutions are generated, repair operators adjust variables to maintain a valid osteotomy order. Together, these mechanisms provide an efficient alternative for maintaining feasible solutions. [20]

This project implements efficient objectives to describe both the overall bone shape and osteotomy surface alignment. Metrics quantifying these objectives are combined to accommodate multiple surgical goals. While small bone protrusions can affect function and should be considered, metrics that are overly sensitive to such outliers may compromise overall alignment. Constraints are primarily enforced using variable bounds and repair operators.

13 Algorithm

This chapter provides the rationale and implementation details of the optimisation algorithm used for osteotomy planning. Understanding the characteristics of different optimisation methods, evolutionary strategies, and hyperparameters is essential for enhancing performance.

13.1. Deterministic and Stochastic Algorithms

To automatically optimise osteotomy plans, an algorithm is needed that can efficiently identify suitable trade-offs between conflicting objectives. Stochastic algorithms are particularly well suited for this purpose, as they evaluate multiple solutions simultaneously and use randomised strategies to avoid becoming trapped in local minima [21].

Deterministic algorithms, such as SQP or Nelder–Mead, follow a fixed search path and therefore produce consistent results. However, they are more likely to converge to a local minimum[23]. Deterministic algorithms may still be considered for refining a single selected solution once the search space has been explored.

13.2. Evolutionary Algorithms

13.2.1. Fundamentals

Evolutionary algorithms (EAs) are inspired by natural selection. Candidate solutions ("individuals") are encoded as parameter sets ("genes") and evolved through selection, crossover, and mutation.

The NSGA-II algorithm was chosen due to its suitability for multi-objective optimisation with continues variables and its ability to maintain solution diversity. This algorithm uses operators such as crossover, mutation, and elitism, which can be tuned to address the specific challenges of osteotomy optimisation:

- **Crossover** combines parent solutions to explore new configurations while preserving relationships between plane parameters.
- **Mutation** introduces controlled variation to avoid local minima and encourage exploration of clinically relevant solutions.
- **Elitism** preserves the best solutions across generations, ensuring that high-quality configurations are not lost due to stochastic effects.

13.2.2. Sampling

Efficient initial sampling of the variable space is critical. Latin Hypercube Sampling (LHS) was used to distribute initial solutions evenly across the parameter space, preventing early convergence to suboptimal regions [26]. Fixing non-critical parameters allows faster convergence on key variables, but reducing diversity can risk missing optimal osteotomy locations. This project also developed Gaussian-based sampling methods that use previous results as input, allowing subsequent optimisation rounds to focus on more detailed or promising regions of the search space.

13.2.3. Crossover and Mutation

Offspring are generated using crossover and mutation. Point-crossover variants preserve relationships between plane variables, while mutation introduces orientation variations. This combination was implemented to balance exploration and exploitation and prevent premature convergence [4].

13.2.4. Quality and Diversity Metrics

Maintaining a well-distributed Pareto front ensures surgeons have multiple viable options. NSGA-II algorithm uses the Crowding distance, which suits our Rapid Optimalisation strategy. It favours solutions in sparsely populated regions of the Pareto front, preventing overrepresentation of specific configurations [20]. Elitism guarantees that the fittest solutions are retained across generations. This prevents the loss of high-quality osteotomy configurations, which could occur due to random genetic drift.

13.3. Repair Mechanisms

For multi-level osteotomies, it is critical to ensure that a proximal osteotomy plane for a distal wedge does not lie above the distal plane of a more proximal wedge, as this would result in an invalid osteotomy configuration. To address this, repair operators have been implemented. Repair mechanisms such as sorting can be used to ensure that each generated solution remains within the defined constraints. Dikland et al.(2023) applied sorting after the generation of the variables. However, for effective convergence, it is essential to adapt the initial parameter directly through the use of custom repair operators. Otherwise, although the final result may lie within the required constraints, the algorithm will still evaluate solutions based on the original, unrepaired parameter values.

13.4. Hyperparameters

Hyperparameters define the behaviour of the evolutionary algorithm and strongly influence performance. The convergence plot in **??** illustrates how the evolutionary algorithm progresses over iterations, demonstrating the improvement and stabilisation of the objectives. It provides a visual example of algorithm performance and helps inform the choice of hyperparameter settings. Tuning these hyperparameters is necessary to ensure the optimisation process reliably produces diverse, clinically meaningful solutions:

- **Sampling frequency**: Affects convergence and accuracy.
- **Population size**: Balances diversity and computational cost.
- **Crossover and mutation rates**: Influence exploration and exploitation.
- **Termination criterion**: Determines when optimisation ends, balancing computation time and solution quality.

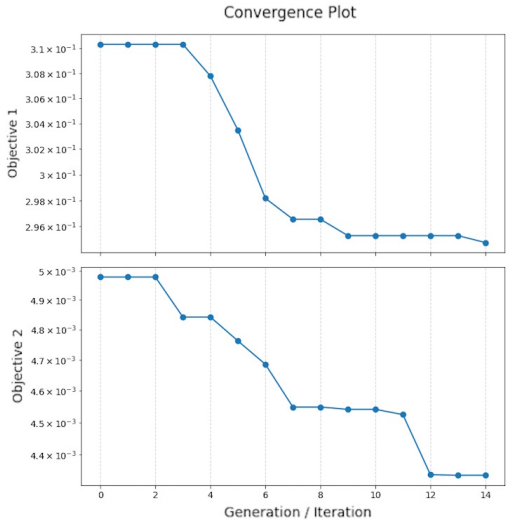


Figure 13.1: Convergence of the two objectives over generations

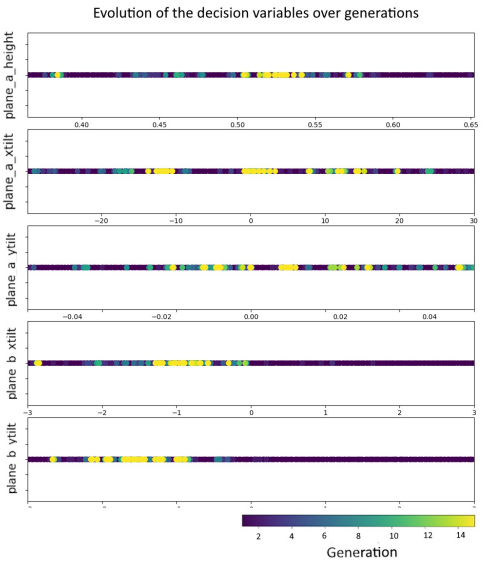


Figure 13.2: Example of the parameter evolution for several variable over generations, showing how the algorithm converges toward optimal values

13.5. Selection of final results

To present the user with multiple results reflecting different trade-offs, a final selection is made from the Pareto front. 13.3 shows an example of the Pareto front evolution over time. The final results are clustered using k-means clustering, based on the two objectives and the height of plane A, ensuring a broad spectrum of configurations is represented. The best solution for each objective is always included.

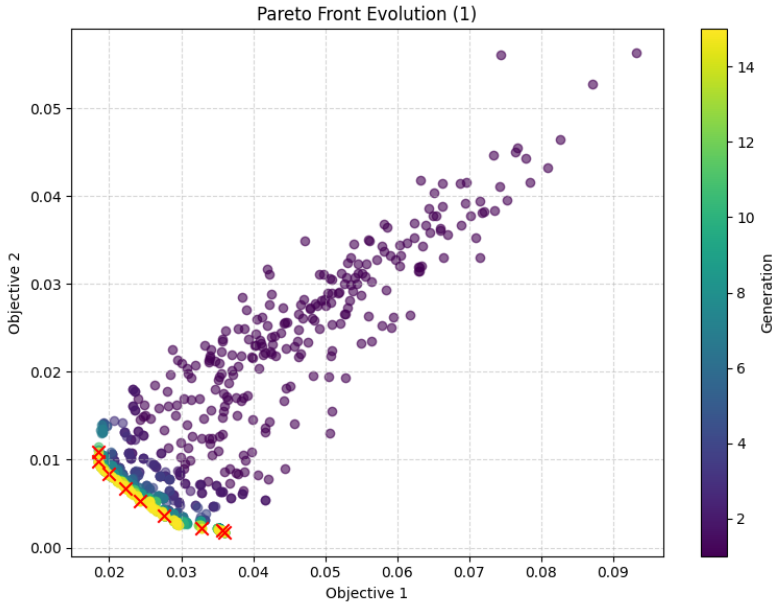


Figure 13.3: Convergence of the two objectives over generations, illustrating the progression of the Pareto front with the selected results to be presented to the user (red crosses).

In summary, the evolutionary algorithm was implemented to efficiently explore the osteotomy parameter space, generate clinically valid solutions, preserve high-quality candidates, and provide a diverse Pareto front for decision-making. Understanding sampling strategies, selection mechanisms, repair methods, and hyperparameters prior to implementation was essential to ensure that the algorithm met both computational and clinical requirements.

14

Data Management

14.1. Data storage

14.1.1. Dataclasses

To manage data flows, consistent variable naming is essential. Structured data is organised using dataclasses, but standard Python dataclasses are unsuitable for long-term storage. Pydantic dataclasses, inheriting from BaseModel, allow serialisation, though custom logic is required for types such as numpy arrays or Path objects.

14.1.2. Data Exchange Between Operators

Data must persist across operators to allow user modifications. Initial approaches using a data group manager, storing dataclasses with run_id and iteration_id, were incompatible with Blender's memory management, as operator data is cleared after execution. Instead, data closely tied to Blender objects, such as bone landmarks, is stored in local coordinates as RNA properties, accessible via the Object Property Editor. Current states and user settings are stored as ID properties, reflecting UI selections.

14.1.3. Long-term Data Storage

Long-term storage ensures data persistence across runs, enables exporting of settings and results, and supports potential recovery after software crashes. While Excel can be used, hierarchical markup languages like XML and JSON are better suited. JSON is preferred for its simplicity, flexibility, and efficient data transmission. Relevant dataclasses are clustered within a parent dataclass and exported to JSON, preserving the hierarchical structure, while functions iterate over property groups to store each property by name.

> Data is stored using Pydantic dataclasses with custom serialization to export hierarchical data. Related data is stored as RNA properties linked to Blender objects, while settings are kept as ID properties accessible in the UI. For long-term storage, JSON is used to preserve the hierarchy in Pydantic dataclasses and enable easy export and import.

Part IV:
Code Documentation

This chapter describes the code architecture and structure of the osteotomy planning add-on. It is intended for developers who wish to develop, extend, or maintain the system, and is designed to be used as a reference while navigating the code, outlining module dependencies, data flows, and the main classes and functions.

15

Add-on Architecture

15.1. Introduction

Forearm osteotomy planning currently requires repeated manual simulations of cuts and repositioning. A robust semi-automated tool is needed to improve efficiency without sacrificing accuracy.

This project uses Blender and pymoo as core tools. Blender, an open source three-dimensional modeling platform, offers flexible visualisation and customisable user interfaces through its Python application programming interface.[25] Pymoo is a Python library for multiobjective optimisation, well suited for exploring complex parameter spaces such as osteotomy plane height and tilt under constraints.[20] Its ability to balance competing objectives makes it suitable for osteotomy planning.

The code adheres to best practices for readability and maintainability, with a modular structure that facilitates updates and feature extensions.

15.2. General Design

The final product is designed to support diverse extensions for other bones and functions. To ensure maintainability and scalability, a well-defined modular structure has been implemented..

Blender's scripting system enables the creation of an addon that can be installed and activated independently. This addon can modify Blender's User Interface (UI), for example by adding panels containing buttons that activate operators and associated functions.

The codebase is organised into the following main modules:

- **Panels module:** Contains all user interface components. The `Initialise_panel` class handles switching between different panels, guiding the user through a logical workflow.
- **Operators module:** Defines actions triggered by user interaction with UI elements. Operators serve as the functional bridge between the interface and corresponding functionalities.
- **Optimisation module:** Contains the core optimisation logic, which is distributed across multiple files to improve modularity, readability, and maintainability.
- **Data module:** All code related to data storage and management is centralised here. Properties interacting with Blender's environment are grouped within `data.properties`. Additionally, `central_data` defines various `dataclasses`, providing a clear and structured representation of stored variables.
- **Utils module:** Contains general functions not tied to a specific operator. The `initialise_addon` function initialises constants, configures logging, and installs required packages. The `data_management` submodule handles data retrieval and storage, while `helper_functions` provides reusable utility functions employed throughout the addon.

15.3. Registration

Blender requires all panels, operators, and properties to be registered. Each module's `__init__.py` defines `register()` and `unregister()` functions. The main `__init__.py` calls these functions on addon activation and deactivation.

15.4. Class Structure

The code is organized into classes that group related functions, each performing a single, well-defined task. Class instances can be passed between components to support modular interaction.

A child class derives from a parent class by specifying the parent in parentheses, inheriting and extending its attributes and methods to enable code reuse and modular design.

Using type annotations makes the structure of stored data self-documenting, enhancing IDE support, code navigation, and error detection. By annotating variables with the data group's name, the IDE can determine the presence or absence of certain information without executing the code, promoting consistent coding conventions.

15.5. Object Naming and Data Consistency

To ensure data consistency, users are alerted when modifying earlier steps. Blender objects are systematically named based on the workflow step in which they are created. This naming convention clarifies which objects will be removed when reverting to a previous step, preventing conflicts or residual data from earlier actions.

15.6. Panel Navigation

`initialise_panel` prepares the interface for the selected panel, ensuring consistent visibility and locking of collections.

The panel interface drawing is delegated to the corresponding module's `draw` function, allowing panel-specific layout and content.

`panel_switch` determines the currently active panel and controls switching between panel views.

Buttons for switching panels call `switch_to_panel` with the current `panel_switch` value as target.

The module's `set_panel_objects` function configures visibility and editability by excluding objects in the active panel from ViewLayer and locking non-active collections to prevent accidental edits.

15.7. Input Fields and Property Management

User input fields are defined in the `data > properties` module, using Blender's native property types such as `StringProperty`, `BoolProperty`, and `EnumProperty`. These properties are grouped using Blender's `PropertyGroup` system, which provides a clear and modular structure for managing input and internal state.

Clustering related properties into `PropertyGroups` offers several advantages. It reduces boilerplate code during registration and unregistration, simplifies access and reuse across modules, and promotes consistency throughout the addon. These groups cover a wide range of functionalities, including UI state, patient navigation, bone and landmark definitions, optimization settings, constraint handling, rotation parameters, and result storage.

Each Blender property supports standard keyword arguments that control its behavior in the UI, such as available items, default values, tooltips, update functions, and visibility conditions. This enables fine-grained control over how input elements are displayed and interacted with during the workflow.

15.8. Initialization

The class is initialized with identifiers used to name and organize the output:

- `prefix`: identifier for the specific bone instance
- `bonetype`: bone type (e.g., radius, ulna)
- `suffix`: panel suffix
- `coll_name`: name of the collection to which the output is assigned

16

Panel A

Panel A allows users to select a patient ID, manage run data, and import STL models. It automates updates to dropdowns, handles file structure creation, and ensures consistent and traceable model imports.

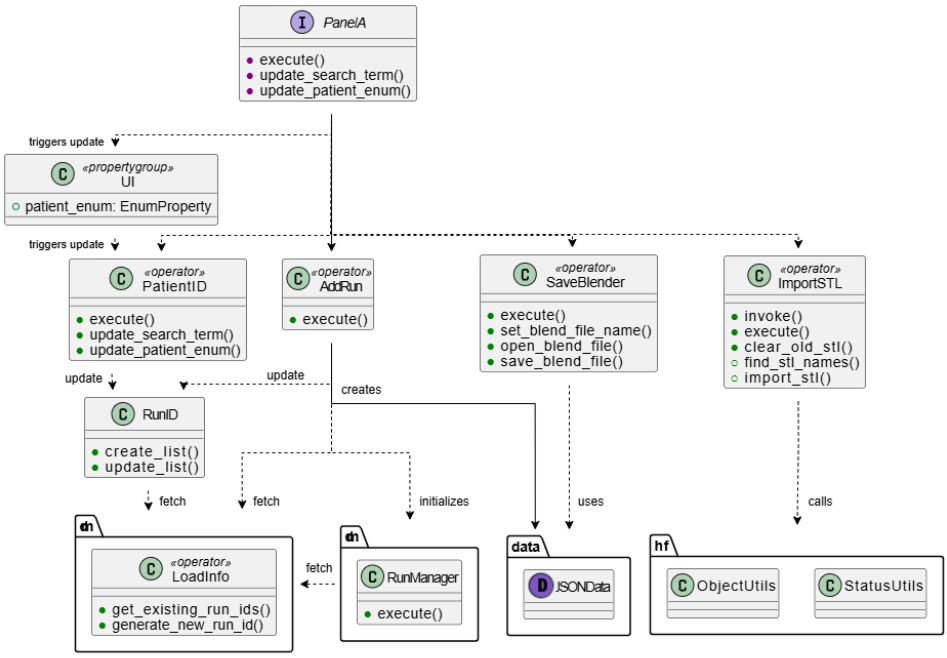


Figure 16.1: Schematic overview of Panel A

16.1. PatientID Selection and Data Retrieval

When the user begins typing into the `PatientID` input field, a series of updates are triggered to guide the user and populate related dropdowns dynamically.

- Patient ID Suggestions:
  As the user types, the `search_term` property is updated, which calls `PatientID.update_search_term`. This triggers `RunID.update_list` to refresh the list of available run identifiers. Simultaneously, the `patient_enum` dropdown updates its items by calling `PatientID.update_patient_enum`, which filters known patient IDs to those matching the typed digits.

16.2. Run ID List Management

- Run ID List:
  The available run IDs depend on both the patient ID (`search_term`) and the selected side (`deformity`). Whenever either of these properties is changed, `RunID.update_list` is triggered. This function uses `LoadInfo.get_existing_run_ids` to retrieve run data from the `index.json` file. The resulting run IDs are stored in a list, which is accessed by `RunID.create_list` to populate the `run_id` dropdown.

16.3. Run Selection and STL Import

This process involves managing and tracking available data efficiently by avoiding full directory scans.

- index.json:
  Upon pressing the `AddRun` button, `get_existing_run_ids` is called from `dm.LoadInfo` to retrieve existing run IDs documented in the `index.json` file. Next, `dm.LoadInfo.generate_new_run_id` generates a new unique run ID.
- sidedata.json:
  A `JSONData` dataclass instance is initialized with this information, after which the `RunManager.execute` method is called. This operator ensures that the folder structure for the selected patient and side exists, and creates or updates the `side_data.json` file to store metadata specific to the run.

16.4. Save Blender State

When the `run_id` property changes, the `SaveBlender` operator is executed. If the corresponding `.blend` file already exists, it is opened. Otherwise, a new file is created and saved at the following location:

- The folder structure is composed as:

```
patient_dir = addon_directory / patient_id / side
```

- The `.blend` file is named:

```
blend_dir = patient_dir / f"{patient_id}_{run_id}.blend"
```

16.5. STL Import

At this stage, the user specifies whether both the radius and ulna should be included in the optimisation process. Once this configuration is set, the user is prompted to import an STL file. During import:

- Any previously imported bone object is automatically removed to prevent duplication.
- The new object is assigned a standardized internal name in Blender to ensure consistency across sessions.
- The mesh retains its original STL filename to maintain traceability and allow validation of the imported data.

17

Panel B

Panel B enables users to place anatomical landmarks on selected bones.

17.1. Select landmark button

After a bone is selected from the `selected_bone` dropdown, the corresponding landmarks are retrieved and buttons are dynamically generated. These landmarks are defined in the `ia.Constants` class.

When the user clicks a landmark button, the `select_landmark` operator is invoked. This operator prepares the system by combining the selected bone and landmark name, and then calls the `select_landmark` function. This function checks whether the marker already exists.

- If the marker is present, it becomes selected.
- If the marker is not present, the user is prompted to place it using mouse input in the modal function.

17.2. Modal function

When the user left-clicks, the system casts a ray from the mouse position into 3D space. It transforms this ray into the local coordinate system of the selected bone and checks if it intersects with the bone's mesh. If a hit is detected, The `create_marker` function places a spherical marker at the hit location in world space and updates the corresponding landmark property boolean flag. If the user presses Escape or right-clicks, the operation is cancelled.

17.3. Landmark properties

Landmark properties show whether the landmark has been positioned. These properties are automatically generated based on the landmark and bone name constants. This approach reduces code redundancy and enhances flexibility when constants are updated.

17.4. Snap Landmark

After a landmark is placed or selected, the corresponding object is snapped to the bone surface using the `enable_snap` operator.

18

Panel C

Panel C extracts local axes, landmarks, centerlines, and matchpoints, storing them as custom properties.

18.1. Bone Info Operator

The `GetBoneInfo` operator orchestrates the extraction of bone-specific spatial data and stores it as structured properties associated with the bone object.

18.2. Definition of axis

Axis are defined based on clinically relevant directions, isntance y positve always pointing to the radial styloid. This means that contralateral bones hold opposite defined axis, lefthanded or righthanded. `landmark_axis_map` maps landmarks to positive or negative x, y, or z label. For radius The z axis corresponds to the longtudinal axis, while the radioulnair y, dorsal volar z, similar definitions for ulna. Based on this label the axis are defined as follows:

- An additional reference point (z-positive) is created as the midpoint between the two y landmarks. The Z-axis is defined from z-negative to z-positive, with the center located at this midpoint.
- The X-axis is computed as the cross product of the Z-axis and a temporary Y-axis vector from y-negative to the center. The final Y-axis is then obtained as the cross product of Z and X, ensuring the axes are mutually orthogonal.
- To ensure correct orientation, the X-axis is checked against the vector from the x-negative landmark to the center. If the dot product is negative, the X-axis is flipped.

18.3. Mirroring

Blender's global coordinate system is right-handed. When aligning a bone defined in a left-handed local coordinate system to the global axes, the object would be mirrored along the Z-axis, which is undesired. To prevent this, the object is parented to another reference object. The reference object is aligned to the global axes, while compensating for all transformations at the child object so that the mesh remains visually unchanged. From this compensated state, only the desired differences in translation and rotation are applied to the object, after which the transform is baked. Baking the transform ensures that the object is fully aligned with the global axes and that no unintended mirroring remains. At this point, the object can be mirrored along the medial–lateral axis, converting the right-handed coordinate system into a left-handed one corresponding to the contralateral bone. These mirrored and aligned objects can then be compared or aligned with the affected bone reliably.

18.4. Matchpoints

The `MatchpointInfo` class retrieves the coordinates of key anatomical points based on a selection of the user-defined landmarks. These points are later used for refined bone alignment. Only a subset of landmarks are suitable for accurate bone matching.

18.5. Centerline

`CenterlineInfo` returns a NumPy array containing the evenly spaced, smoothed centerline coordinates. Moreover, a Blender curve object is optionally created for visualisation. Following the approach presented by Dikland et al.(2023), direct coordinate handling without creating, moving and removing Blender objects, significantly improves workflow speed.

1. The process begins at the distal axis center (`center_coord_start`). A bmesh plane is defined perpendicular to the `z`-axis of the distal axis and intersected with the bone mesh to find the initial centerline point.
2. Using the distal coordinate system, a new plane is placed slightly proximal to the previous one to locate the next centerline point.
3. Subsequent planes are positioned iteratively based on the direction defined by previous centerline points.
4. This iterative process continues, adding estimated centerline coordinates until the direction becomes unstable or a maximum iteration count is reached.
5. The resulting raw centerline is then resampled, smoothed, and resampled again to produce the final, refined centerline.

19

Panel D and E

Panel D automate the alignment of affected and reference bones. It duplicates and mirrors models, applies SVD and ICP methods for coarse-to-fine registration, and records where the bones start to diverge to determine a Region of Interest (ROI) for the osteotomies. Panel E aligns the distal portion of a copy of the affected bone to the reference bone that has already been aligned proximally.

19.1. Position Operator

Panel D calls the `PositionFunctions` operator, which automates the preparation and alignment of affected and reference bones. It handles alignment of the reference bones to either the proximal or distal side of the affected anatomy. Similarly, Panel E invokes the `AffRefOperator`, which aligns the distal portion of a copy of the affected bone to the reference bone that has already been aligned proximally.

The sequential alignment steps within the `PositionFunctions` operator are as follows:

- `PositionFunctions.parent_objects` parents all associated objects of the reference bone (e.g., centerlines, markers) to an empty object, enabling them to transform consistently with the reference bone.
- `MirrorFunctions.execute` aligns the main axes of the reference bone to the global axes and flips the bone across the sagittal plane to match the affected side.
- `AlignFunctions.extract_align_part` is used by SVD and ICP to divide centerline or mesh point sets into proximal and distal segments based on a cutting plane.
- `SVDFunctions.align_svd_matchpoints()` performs coarse alignment using SVD based on matchpoints.
- `ICPFunctions.align_icp_mesh()` refines alignment iteratively by matching mesh surfaces.

19.2. Align Functions

`AlignFunctions.extract_align_part` separates proximal and distal segments of point sets based on a cutting plane at a relative height along the centerline. The function `align_plane` converts this height into a sample index on the centerline. Depending on whether the index matches a sample point exactly, one of two `MyPlaneInterpreter` functions is called: either an exact orientation function or an interpolated approximation, as presented by Dikland et al.(2023)

The local coordinate and direction are stored in `PlaneData`. `transform_planedata` applies the bone's `matrix_world` to transform the plane data to the bone's global position.
Finally, `MyStlVisualiser.create_plane_from_normal` generates a Blender object from the transformed `PlaneData`, while `MyOtomyInterpreter.split_coords` separates the proximal and distal coordinates relative to this plane.

19.3. SVD Functions

This class computes the optimal rigid transformation between two point sets. Both sets are first centered around their centroids. A cross-covariance matrix (`svd_h`) is calculated to describe the spatial relationship between the sets. SVD decomposes this matrix into directional components, from which the optimal rotation is constructed. After applying the rotation, the translation aligning the centroids is computed.

19.4. ICP Functions

This class performs alignment using the Iterative Closest Point (ICP) algorithm. Points are converted to homogeneous coordinates, and an initial alignment is applied if provided. The algorithm iteratively finds corresponding points between the moving and fixed sets using `MyMetrics.nearest_neighbor`.

At each iteration, `best_fit_transform` computes the least-squares optimal rigid transformation via SVD, which is then applied through matrix multiplication. Iterations continue until convergence or a maximum iteration count is reached. The final transformation matrices are returned.

19.5. Estimate Regions of Interest

The `EstimateRoiFunction` operator identifies anatomical regions where affected and reference bones diverge. The functions `nearest_neighbor` and `find_divergence_point` are called sequentially to detect divergence along the centerlines. A region is considered divergent if the distance exceeds a defined threshold for at least 15 consecutive points, ensuring detection of true anatomical differences rather than local noise.

The proximal ulna joint is excluded from the analysis because this region naturally shows divergence. Once divergence points are identified, they are visualized as spheres. Finally, the coordinates are converted into local space using `get_global_or_local` and stored in the affected bone's custom properties as `roi_prox` and `roi_dist`.

20

Optimalisation Problem Definition

This panel orchestrates the multi-round optimisation workflow, from broad exploration to fine local search, while ensuring the core computation remains independent of Blender-specific operations.

20.1. Optimisation Operator

Due to the iterative nature of optimisation algorithms, it is undesirable to interact with Blender attributes during runtime. Therefore, all necessary object and context data is extracted in advance by `_prepare_obj_data` and `_prepare_context_data`. In the current implementation, only a single optimisation round is utilised, although the framework is designed to support multiple rounds with progressively refined search settings. Results are stored in dictionaries, grouped into clusters, and exported as JSON files.

20.2. Optimisation Round

Each round uses the Pymoo `minimise` function. Before the run begins, supporting modules define the problem, configure algorithms, and handle constraint setup. Upon completion, the `SelectResults` class is used to identify suitable results from the Pareto front and `my_visualiser` to create Blender objects of the best solution.

20.3. Parallelisation

Parallel execution can be used to optimise performance, particularly for computationally expensive evaluations. Two main strategies are implemented, with function-level parallelisation adopted in the final methodology, although its impact on performance with new lightweight objectives has not yet been evaluated:

1. **Island-level parallelisation:** Each "island" evaluates a subset of candidate solutions independently before calling the problem. This allows distributing the workload across multiple CPU cores.
2. **Function-level parallelisation:** Within `MySharedProblem`, the execution of all evaluation functions (`all_functions`) can be parallelised using `ThreadPoolExecutor`. Each future handles a single height interval.

The number of parallel processes should not exceed the number of CPU cores, and memory usage should ideally stay below 85% to avoid performance degradation.

```
with ThreadPoolExecutor(max_workers=n_cpu_used) as executor:
    futures = []
    for future_idx, (min_height, max_height) in enumerate(intervals):
        futures.append(executor.submit(_optimalisation_round_0_idx,
                                       min_height, max_height, ...))
```

20.4. Problems

Two problem classes are defined, both subclasses of `pymoo`'s problem types:

- `MyProblem`: Handles batch evaluation of candidate solutions. All candidates in a batch are evaluated simultaneously, which is efficient when evaluations are relatively fast and independent. This is used in the final implementation.

- `MyElementwiseProblem:` Handles individual evaluation of candidates, one at a time. This is useful for the island method, where each "island" or subpopulation can evaluate candidates independently and in parallel. It also allows fine-grained error handling and assigning penalties per candidate.

Both problem types define the number of decision variables and share a common evaluation object. `_evaluate` is called by `pymoo` with a batch of candidate vectors, which are then passed to `MySharedProblem.my_evaluate(x, out)`.

Each candidate `x` is parsed into a set of `ParamData` instances, one for each osteotomy. Each of these contains:

- Two cutting planes: `plane_a` and `plane_b`, both defined by `PlaneParamData` with height, x-tilt, and y-tilt.
- A translation direction, magnitude, and rotation value.

The interpreter classes apply the osteotomy using these parameters, returning an `OtomyDataGroup` object containing planes A and B, the transformation matrices, and the type of osteotomy. Objective and constraint scores (`out["F"]` and `out["G"]`) are computed using `MyObjectives` and `MyConstraints`.

Error Handling:
If any errors or invalid data (e.g., NaNs) occur during evaluation, default penalty values are assigned to ensure robustness of the optimizer. Every `N` evaluations, selected candidates may be visualised to support debugging or interpretation.

20.5. MyOtomyInterpreter

This class orchestrates the osteotomy sequence by iterating through parameter sets (`param_dataset`) and applying corresponding geometric transformations to the proximal shape data (`aff_prox_shape_data`). Each iteration modifies the shape data, carrying over results into the next osteotomy step, thus simulating cumulative deformational effects. The output consists of the `otomy_datagroupset`, `aff_prox_shape_i_data`, and `aff_dist_shape_i_data`, containing metadata for each cut, and the final transformed proximal and distally aligned bone geometry.

- For each parameter set, two osteotomy planes (`plane_a` and `plane_b`) are generated via `MyPlaneInterpreter`.
- For the final osteotomy, plane B is derived relative to the aligned distal bone using `MyDistalPlaneInterpreter`.
- A Boolean flag (`opening_bool`) indicates whether the osteotomy is opening or closing.
- `calculate_transformation_matrices()` computes the 4x4 matrices aligning `plane_a` to `plane_b`.
- Attributes (like `centerline`) are split into proximal and distal segments via `split_coords()`.
- `_transform_attributes(...)` applies transformations.
- The segments are merged and reassigned into the `ShapeData` structure.

20.6. MyDistalPlaneInterpreter

- Alignment of distally aligned bone: `get_basic_distalplanedata` retrieves plane data of `plane_d0` and `plane_d`, the distal plane aligned with the distal axis, and its transposed variant defined by the `ParamData`. Both planes are stored as `PlaneData` objects. `set_configuration_distal_part` applies the distal transformation to mesh and landmark data. The distally aligned affected bone is transformed based on the displacement between `plane_d0` and `plane_d` using Singular Value Decomposition (SVD) alignment, creating a slight misalignment with the reference bone.
- Alignment of distal bone segment: `self.matrix_from_distwaypoints` obtains the translation matrix from the proximally aligned bone to the distally aligned bone, used by `get_plane_target_data` to transform plane A to the position of plane B. `determine_openingbool_by_planedata` classifies the osteotomy as opening or closing based on intersection analysis of both planes with the centerline.

20.7. Repairs

The `_do` method is called for each generation of candidate solutions, iterating over every individual.
`get_plane_heightset` extracts and returns the heights and sets Boolean flags indicating whether the original pair was in an "opening" configuration (i.e., proximal height $<$ distal height). All heights are globally sorted to avoid overlap and ensure correct ordering across osteotomies.

- `implement_max_distance`: Prevents overly large openings or shortenings by clamping the difference between plane heights. To maintain divergence, the new relative plane height is determined based on the initial height. By subtracting the amount of thresholds that fit into the difference, the relative height

is transformed into a height within the threshold.
- `implement_min_distance`: Ensures adequate spacing between adjacent osteotomies, especially between the last two segments.

`order_plane_height` determines the final order of a plane pair based on constraints or original configuration. For the final osteotomy, the distal plane is fixed based on input parameters. Each modified plane height is written back into the design matrix X, ensuring that the repaired solution is valid for further evaluation.

20.8. Bounds

The `get_opt_bounds` method constructs a set of lower and upper bounds for each correction osteotomy. Since the parameters of the second plane are interpreted differently at the last osteotomy, the bounds are set accordingly to enable tighter limits.

After populating the bounds dictionary for each osteotomy, the bounds are flattened into a 2D NumPy array of shape `(n_parameters, 2)`, with each row representing `[lower, upper]` bounds for one parameter. This structure is suitable for direct use in evolutionary or gradient-based optimizers.

20.9. Objectives

The `MyObjectives.execute()` method returns a dictionary of computed objective values such as alignment error, surface mismatch, and cut-plane conformity. If the weight of a data category (e.g., centerline) is non-zero, the `MyMetrics.execute` function is called to compute specified submetrics. Then, the metric of the data category is defined as:

$$metric["mesh"] = weights\_tot[key] * submetric[key]$$

To compute surface alignment, the `get_cut_surface_objective` method slices the mesh along the specified planes and transforms the resulting sections using the bone's transformation matrices.
`reorder_points_in_circle` and `resample_closed_curve` ensure the metrics can be used effectively.

20.10. Constraints

Constraints are computed by invoking a series of private methods. Results are collected, and `_handle_nans` replaces NaNs with large penalty values to avoid optimizer errors. The final output is a flat array of constraint violation values. Each constraint is reduced to the following form for continuity:

$$constraints\_dict['length'] = [(abs(value - target) - tol) * weight]$$

20.11. Samplers

The class is initialized with a `MyProblem` object, from which it extracts the number of osteotomies and the lower and upper bounds for decision variables. This ensures consistent sample generation aligned with the defined optimization problem.

- `set_samples_1` uses Latin Hypercube Sampling (LHS). Each row represents a complete osteotomy configuration. The `_set_samples_to_zero` method masks certain variables with zero.
- `set_samples_2` builds upon previous optimization results by adding Gaussian noise from a truncated normal distribution defined by each variable's bounds.
- `set_samples_3` performs local exploration by adding noise scaled to 5% of the total range, shifted appropriately. The resulting sample matrix maintains all values strictly within bounds via `np.clip`.

20.12. Outputs

- `MyOutput`: If verbose display is enabled, it shows live summary statistics in the console during optimization.
- `MyPlotter`: Uses output data from the callbacks to produce: 2D and 3D Pareto front plots, objective and constraint evolution plots, and parameter convergence scatter plots. `save_plot` stores plots to disk using paths resolved through the Blender API (`bpy.context.scene.ui_properties.addon_directory`).

20.13. Convergence Tracking

- Saving History: Stores all generations for post-analysis, which is memory-intensive.
- Callback: Collects selected metrics at runtime. Access via `res.algorithm.callback` keeps the original algorithm object unaltered.

- Checkpoints: Periodically store the algorithm state to facilitate resuming after interruptions.
- Verbose Output: Real-time logging of progress and performance.

20.14. Results

The `Result` object summarises optimisation outcomes. If no feasible solution is found, it is set to `None`. For multi-objective problems, a Pareto front of non-dominated solutions is returned.

20.15. Visualisation

For visualising the preoperative situation, the osteotomy planes are positioned relative to the original bone geometry, without applying transformations to the mesh itself. The final osteotomy presents a special case: while Plane A remains unchanged, determining the correct representation of Plane B is more complex.

- For most osteotomies, the preoperative Plane A and B are known directly by applying the parameters to the centerline.
- For the final osteotomy, Plane B is reconstructed. First, all osteotomies are performed except the last. Then, a second version of the affected bone is aligned distally. The post-operative position of Plane B is created based on the first parameter set, applied to the distally aligned bone.
- When a single osteotomy is applied, this plane position is the preoperative position. If multiple osteotomies are applied, the planes do not represent the positions where the bone should be cut in the original configuration. `MyRevOtomyInterpreter` reverses the applied transformations.
- Once the plane positions are known prior to transformations, they can be used to cut the Blender object and plan the PSI. The Blender objects are cut and transformed by functions from `MyStlVisualiser`, based on the `openingboolset`, which holds a list of Booleans indicating whether each osteotomy should be opening. The bone is only cut at plane A (opening), or the bone between plane A and B is removed (closing).
- The bone parts are then removed.

20.16. MyDataVisualiser

The class `MyDataVisualiser` offers methods such as `visualize_osteotomy_result`,
which calls `create_mesh_from_coords` and `create_centerline` to generate mesh objects and curves from 3D coordinates representing the bone and centerline. Additionally, `create_plane_mesh_from_data` and `create_plane_mesh_from_points` generate planar mesh objects from plane definitions or points.

20.17. MyStlVisualiser

Planes created by `generate_osteotomy_planes` are used by `cut_otomy_parts` to cut the bone objects. This function delegates to the correct methods based on a Boolean flag indicating whether it is an opening or closing osteotomy.

- Opening Osteotomy: `_cut_into_parts_opening` performs a cut at Plane A; the proximal bone loses mesh distal to A and the distal bone loses mesh proximal to A.
- Closing Osteotomy: Private methods `_create_osteotomy_objects` and `_cut_into_parts_closing` remove the bone between Plane A and Plane B. The proximal bone loses mesh distal to B and the distal bone loses mesh proximal to A.

The osteotomy procedure involves three main cutting steps:

1. A convex hull of the bone is created to cover the entire cut region, then trimmed to define the region of bone to be removed.
2. This object is used to subtract mesh from duplicates of the bone, ensuring clean removal via Boolean subtraction.
3. Finally, the osteotomy planes are used to split the bone into separate proximal and distal objects.

20.18. Intersection Segments

In some osteotomies, the cutting planes (A and B) intersect within the bone volume, requiring special handling of overlapping mesh regions:

- Opening Osteotomy: If the distal part moves into a region already occupied by proximal bone mesh, this overlapping segment must be removed. This region lies *proximal to A* and *distal to B* and is referred to as `a_to_b`. It is added to one of the final cutting objects (`otomy_obj_prox_total` or `otomy_obj_dist_total`).

- Closing Osteotomy: In some cases, mesh in the overlapping region is reused in both the proximal and distal bone. To prevent duplication, the region *distal to A* and *proximal to B*, denoted `b_to_a`, must be removed from one of the parts.

`transform_otomy_parts` applies a series of matrices to the osteotomy segments. Matrix inversion is supported by `get_inverse_matrixdata` and `safe_inverse`. The functions `transform_part` and `transform_cos` handle the application of 4x4 transformation matrices to vertex coordinates of Blender mesh objects. `get_mesh_cos` retrieves the world-space vertex coordinates of a mesh.

20.19. Select Result

Users can navigate through multiple optimisation results using `MoveResultUpOperator` and `MoveResultDownOperator`, which increment or decrement the selected result index and invoke `SetSelectedResultOperator`. This updates the optimisation property `selected_result` and invokes `ActivateCollectionOperator`, ensuring that only the active result's collection is visible.

20.20. Position Plate

For surgical planning, the `PlatePositionOperator` launches a modal interface that lets users click on the bone surface to position the marker using ray casting (`context.scene.ray_cast`). Snapping is enabled via `OBJECT_OT_enable_snap.execute`.

20.21. Select Representative Results

The `SelectResults` class implements a pipeline to extract a set of representative solutions. Its workflow is initiated through the

Part V:
Discussion and Future Outlook

This chapter provides a summary of the key findings of this thesis and offers recommendations for future research and potential improvements. It includes reflections on the lessons learned and discusses the potential clinical impact of the developed tool.

21

Discussion and Future Outlook

This chapter discusses the outcomes, technical design decisions, and clinical relevance of the 3D-SurgiGen Add-on. It synthesises findings from the development and evaluation phases, reflecting on how the project addressed its overarching goal: To develop an open-source semi-automated surgical planning tool for forearm osteotomies, improving surgical planning efficiency.

The discussion follows the project's core objectives, translating clinical needs into algorithmic design, developing a user-centred workflow, establishing a robust and extensible architecture, and evaluating usability and clinical potential.

21.1. Translation of Clinical Needs into Algorithmic Design

The Add-on was designed to address clinical priorities such as aligning the overall shape, ensuring accurate joint alignment, and generating feasible osteotomy configurations. By embedding these objectives within a multi-objective optimisation framework, the system supports the creation of realistic surgical plans that respect clinical constraints.

Unlike earlier optimisation-based methods, which often relied on full mesh transformations or computationally intensive surface comparisons, this implementation focuses on the centreline and osteotomy region only. This approach maintains alignment accuracy while substantially reducing computational complexity, enabling interactive user involvement during optimisation.

The algorithm generates multiple feasible alternatives, allowing the clinician to explore trade-offs interactively. This hybrid approach combines algorithmic efficiency with clinical expertise, ensuring that final plans remain both geometrically sound and clinically practical.

Previous methods for osteotomy optimisation often relied on computationally intensive strategies. The cut-surface minimisation approach proposed by Carrillo et al. (2017) optimises a fundamentally different objective from the osteotomy-surface optimisation presented here. Their method minimises the area of the cut surface by approximating it as an ellipse derived from principal component analysis of projected points. However, Carrillo et al. (2020) later concluded that this metric is insufficient to capture clinically meaningful alignment.

The bone protrusion method quantifies deviations between fragments by identifying regions where the bone surface diverges from a reference target within a predefined window surrounding the osteotomy plane [1]. While effective, this approach requires partial transformation of mesh data and computation of point-to-surface distances, making it computationally expensive. Carrillo et al. (2017, 2020) reported run times of 1 h 44 min and 85.38 $\pm$ 33 min, respectively, as their optimisation also considered plate and screw positioning.

Previous work by this group used bone protrusion as the sole metric, with run times of 74–81 minutes [3]. Such durations may not be suitable for interactive workflows that allow users to adjust optimisation settings based on intermediate results. In contrast, the current method achieves similar clinical outcomes in approximately 2 minutes by transforming only the intersections of osteotomy planes with the preoperative mesh.

The method used in this thesis aligns the osteotomy surface with the bone centreline to minimise bone protrusion while maintaining sufficient surface continuity. This metric reduces gaps in opening-wedge osteotomies, optimises alignment in closing-wedge osteotomies, and can be extended to hybrid osteotomies. Future refinements could categorise points on the osteotomy plane according to their relevance for gap minimisation

or plate positioning, enabling more targeted evaluation for complex geometries.

21.2. User-Centred Workflow and Interaction Design

A major focus of the project was to design an intuitive workflow that reflects the logic of preoperative planning. Through iterative development and testing, the interface was structured to guide users through each stage of the process, from landmark annotation to optimisation and plan selection.

Usability testing confirmed that clinicians could quickly learn to operate the tool, though several interface limitations were identified. These include restricted control over small manual adjustments, limited visualisation flexibility, and terminology that was occasionally non-intuitive. Nevertheless, the design successfully achieved a balance between automation and user control, ensuring that clinical judgement remains central to decision-making.

One of the implemented workflow strategies is creating a effecitve system to navigate between files, allowing the user to quickly navigate though optimisation results. However, using the branching functionality required explicit instruction and used minimally during planning for the evaluation fase. Requiring to create a new run before optimising the other bone was reported to be confusing. Moreover, the current implementation of file management is not on a level yet to safely implement in clinical practise

21.3. Technical Architecture and Extensibility

The project emphasised modularity, maintainability, and transparency. Rather than adapting previous codebases tailored to femoral osteotomies, a new architecture was developed from first principles, while reusing suitable code segments. The framework incorporates clear separation of concerns, consistent naming conventions, and structured data handling.

This foundation ensures scalability and facilitates integration of future features such as multi-bone alignment, saw loss representation, or implant positioning. The use of Blender as an open-source platform further enhances accessibility, enabling future research groups to extend or adapt the software to their specific needs.

21.4. Validation and Performance

The 3D-SurgiGen Add-on was prospectively evaluated in five clinical cases involving deformities of both the radius and ulna. Semi-automated planning produced a comparable number of clinically acceptable plans to manual planning, with similar rates of failure and weaknesses. Planning time was not reduced (median difference: 7 minutes, range: 4–17). However, the semi-automated workflow enabled broader exploration of potential osteotomy configurations within comparable timeframes.

These findings demonstrate the feasibility of semi-automated optimisation-based planning. While clinical efficiency was not yet improved, the tool effectively generated feasible osteotomy planes and supported surgeon-specific preferences. Limitations include the small sample size, the exclusive inclusion of midshaft deformities, and the absence of saw loss representation or advanced manual adjustment tools.

21.5. Clinical Implications

The Add-on provides an interactive environment for exploring osteotomy options. It is best positioned as a complementary decision-support tool rather than a replacement for manual planning. By allowing surgeons to visualise, compare, and select from multiple optimised solutions, the tool supports clinical reasoning and fosters confidence in the final surgical plan. Its open-source nature further supports collaboration, training, and adaptation across different institutions.

21.6. Strengths and Limitations

A major strength of this work lies in the integration of a fully open-source, modular, and extensible system architecture that bridges computational methods and surgical practice. Direct comparison between manual and semi-automated planning in matched cases provided valuable insight into usability, workflow efficiency, and optimisation behaviour.

However, several limitations remain. The study included only five cases and focused exclusively on single osteotomies on midshaft deformities, limiting generalisability. The strong morphological recognisability of the bones constrained anonymisation, introducing potential observer bias. In addition, several functional limitations, such as the absence of explicit bone segment removal, non-intuitive interface elements, and limited interactive parameter control, should be addressed in future iterations.

21.7. Future Directions

Future research should expand validation to a larger and more diverse cohort, including distal and proximal forearm deformities. Technical extensions could include simulation of saw loss, implementation of multi-osteotomy options for the forearm, specific control over bone alignment for plate positioning, improved representation of clinically relevant regions such as the radial bow, integration of plate and screw positioning, and constraint-based guidance to restrict osteotomy planes to anatomically safe areas.

Enhanced control over object visualisation and clear guidance on which parameters should or should not be altered would further improve usability. The project provides a structured codebase designed with future development in mind for other bones and anatomical regions. As the majority of parameters are currently defined in a dedicated file for the radius and ulna, these can be relatively easily adapted to the requirements of other bones. By combining precise control of distal alignment with the preservation of multi-level osteotomy functionality, the tool offers promising opportunities for broader clinical application. By providing an elaborate technical document that not only explains the choices made but also describes alternative approaches and aspects that could further enhance development, optimal knowledge transfer is established, enabling future software improvements and adaptation to the requirements of other bones.

21.8. Niche Opportunities

Excessively large gaps between adjacent bone segments can impede healing or lead to implant failure due to instability. In surgical practice, such gaps are often filled with structural bone grafts to support bone healing. During evaluation, it was emphasised that the current research scope does not encompass these advanced planning scenarios. While automation of bone graft planning has been explored in cranio-maxillofacial surgery, for example, reconstructing the mandible with vascularised fibula grafts, comparable automated workflows for long-bone reconstruction have not yet been established to the authors' knowledge.

The 3D-SurgiGen Add-on framework may be well-suited for extending its functionality to support preoperative planning in these scenarios. By enabling simultaneous interactive visualisation of both donor and recipient bones, the recipient defect can be defined using two osteotomy planes to specify the segment to be removed. This segment can then be projected onto the donor bone to identify candidate grafts. The workflow would need to optimise both the size of the donor fragment and the subsegment used as a graft, ensuring appropriate correction of both bones. Determining a suitable reference scale is pivotal in such optimisation problems, and the Add-on's interactive environment provides a platform for exploring scaling and alignment adjustments.

As the recipient's model is scaled down, the required graft size decreases. An approximate scaling factor can be estimated once the distal portion of the donor sufficiently bridges the graft-induced gap. Subsequent optimisation could then be explored using the scaled reference models. The method presented for representing osteotomy surface borders could be incorporated as additional objectives to characterise postoperative contact regions between adjacent bone segments of both the radius and ulna.

21.9. Concluding Remarks

This thesis demonstrates the feasibility of semi-automated, optimisation-based preoperative planning for corrective forearm osteotomies. The 3D-SurgiGen Add-on provides a structured and efficient framework that supports both accuracy and flexibility by combining algorithmic automation with clinical input. While further refinement is needed before clinical use, this work establishes a foundation for future open-source surgical planning systems that aim to standardise preoperative workflows and support surgeon decision-making.

References

[1] F. Carrillo et al. "An automatic genetic algorithm framework for the optimization of three-dimensional surgical plans of forearm corrective osteotomies". In: *Med Image Anal* 60 (2020), p. 101598. ISSN: 1361-8423 (Electronic) 1361-8415 (Linking). DOI: 10.1016/j.media.2019.101598. URL: https://www.ncbi.nlm.nih.gov/pubmed/31731091.

[2] F. Mauler et al. "Prediction of normal bone anatomy for the planning of corrective osteotomies of malunited forearm bones using a three-dimensional statistical shape model". In: *J Orthop Res* 35.12 (2017), pp. 2630–2636. ISSN: 1554-527X (Electronic) 0736-0266 (Linking). DOI: 10.1002/jor.23576. URL: https://www.ncbi.nlm.nih.gov/pubmed/28390188.

[3] F.M.A. Koopman et al. "Automation and Optimization of Computer-Assisted Surgery Planning of Corrective Osteotomies for Forearm Malunions". In: (2023).

[4] G. Wu et al. "ISB recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion–Part II: shoulder, elbow, wrist and hand". In: *J Biomech* 38.5 (2005), pp. 981–992. ISSN: 0021-9290 (Print) 0021-9290 (Linking). DOI: 10.1016/j.jbiomech.2004.05.042. URL: https://www.ncbi.nlm.nih.gov/pubmed/15844264.

[5] H. Hoekstra et al. "Corrective limb osteotomy using patient specific 3D-printed guides: A technical note". In: *Injury* 47.10 (2016), pp. 2375–2380. ISSN: 1879-0267 (Electronic) 0020-1383 (Linking). DOI: 10.1016/j.injury.2016.07.021. URL: https://www.ncbi.nlm.nih.gov/pubmed/27498242.

[6] J. Cognet et al. "Distal radius malunion in adults". In: *Orthop Traumatol Surg Res* 107.1S (2021), p. 102755. ISSN: 1877-0568 (Electronic) 1877-0568 (Linking). DOI: 10.1016/j.otsr.2020.102755. URL: https://www.ncbi.nlm.nih.gov/pubmed/33316441.

[7] J.G.G. Dobbe et al. "Patient-tailored plate for bone fixation and accurate 3D positioning in corrective osteotomy". In: *Medical & Biological Engineering & Computing* 51.1–2 (2012), pp. 19–27. DOI: 10.1007/s11517-012-0959-8.

[8] M. J. Richard et al. "Malunions and nonunions of the forearm". In: *Hand Clin* 23.2 (2007), pp. 235–43, vii. ISSN: 0749-0712 (Print) 0749-0712 (Linking). DOI: 10.1016/j.hcl.2007.02.005. URL: https://www.ncbi.nlm.nih.gov/pubmed/17548014.

[9] N. Andrey et al. *Linear algebra. points matching with SVD in 3D space*. Nov. 2019. URL: https://medium.com/machine-learning-world/linear-algebra-points-matching-with-svd-in-3d-space-2553173e8fed.

[10] N. Gunantara et al. "A review of multi-objective optimization: Methods and its applications". In: *Cogent Engineering* 5.1 (Jan. 2018), p. 1502242. DOI: 10.1080/23311916.2018.1502242.

[11] O. Fujita et al. "Metrics based on average distance between sets". In: *Japan Journal of Industrial and Applied Mathematics* 30.1 (June 2012), pp. 1–19. DOI: 10.1007/s13160-012-0089-6.

[12] P. Jayakumar et al. "Reconstruction of malunited diaphyseal fractures of the forearm". In: *Hand (N Y)* 9.3 (2014), pp. 265–73. ISSN: 1558-9447 (Print) 1558-9455 (Electronic) 1558-9447 (Linking). DOI: 10.1007/s11552-014-9635-9. URL: https://www.ncbi.nlm.nih.gov/pubmed/25191155.

[13] R. Shiode et al. "Reproduction of forearm rotation dynamic using intensity-based biplane 2D-3D registration matching method". In: *Sci Rep* 14.1 (2024), p. 5518. ISSN: 2045-2322 (Electronic) 2045-2322 (Linking). DOI: 10.1038/s41598-024-55956-z. URL: https://www.ncbi.nlm.nih.gov/pubmed/38448504.

[14] S. Kim et al. "Rotation Representations and Their Conversions". In: *IEEE Access* 11 (2023), pp. 6682–6699. ISSN: 2169-3536. DOI: 10.1109/access.2023.3237864.

[15] S. Vijayakumar et al. "Assessing the effectiveness of Moscow prioritization in software development: A holistic analysis across methodologies". In: *EAI Endorsed Transactions on Internet of Things* 10 (2024). DOI: 10.4108/eetiot.6515.

[16] S. Xiuying et al. "The Iterative Closest Point Registration Algorithm Based on the Normal Distribution Transformation." In: (2018).

[17] U. Hassani et al. "A systematic review of optimization algorithms for Structural Health Monitoring and optimal sensor placement". In: *Sensors (Basel, Switzerland)* (). URL: https://pubmed.ncbi.nlm.nih.gov/36992004/.

[18] F.A. Diklan et al.d. "Semi-Automatic Generation of Preoperative Surgical Plans for Complex Femoral Deformity Correction". MSc Thesis. 2023.

[19] *Automate your surgical guide design workflow*. Materialise. n.d. URL: https://www.materialise.com/en/academy/healthcare/mimics-innovation-suite/video-tutorials/automate-surgical-guide-design-workflow (visited on 07/06/2025).

[20] J. Blank and K. Deb. "pymoo: Multi-Objective Optimization in Python". In: *IEEE Access* 8 (2020), pp. 89497–89509.

[21] K. Deb et al. "A fast and elitist multiobjective genetic algorithm: NSGA-II". In: *IEEE Transactions on Evolutionary Computation* 6.2 (2002), pp. 182–197. DOI: 10.1109/4235.996017.

[22] D.F.R. van Loon et al. "Automatic identification of radius and ulna bone landmarks on 3D virtual models". In: *Comput Biol Med* 179 (2024), p. 108891. ISSN: 1879-0534 (Electronic) 0010-4825 (Linking). DOI: 10.1016/j.compbiomed.2024.108891. URL: https://www.ncbi.nlm.nih.gov/pubmed/39047505.

[23] *Mastering optimization algorithms in machine learning a comprehensive guide*. Web Page. 2024. URL: https://medium.com/@sayedebad.777/mastering-optimization-algorithms-in-machine-learning-a-comprehensive-guide-2fcab33a6377.

[24] *Open3D: A Modern Library for 3D Data Processing*. Web Page. URL: https://www.open3d.org/.

[25] *Quickstart. Quickstart - Blender Python API*. n.d. URL: https://docs.blender.org/api/current/info_quickstart.html (visited on 07/06/2025).

[26] V. Toğan and A.T. Daloğlu. "An improved genetic algorithm with initial population strategy and self-adaptive member grouping". In: *Computers Structures* 86 (2008), pp. 1204–1216. DOI: 10.1016/j.compstruc.2008.03.001.

Appendix

#	Requirement	Priority
1	**STL file handling**	
1A	UI facilitates the import of STL files.	M
1B	Automatically label STL files to indicate affected and reference bone.	M
1C	Reduces STL file size without compromising essential information.	S
1D	Registers bone sufficiently to avoid optimization convergence to local minima.	M
1E	Effectively visualizes the model output for insightful interpretation.	M
2	**Osteotomy plane optimization**	
2A	Allows manual adjustment of osteotomy planes.	C
2D	Osteotomy planes pass through the bone shaft.	M
2E	Avoids intra-articular cuts.	M
2F	Avoids no-cut zones.	M
2G	Creates cuts within Must-cut zones.	M
2H	Prevents slippage.	M
2I	Prevents shear forces.	M
2J	Restricts osteotomy angle.	M
2K	Avoids large open wedges or gaps.	M
2L	Avoids excessive length shortening.	M
2M	Prevents excessive tissue torsion.	M
2N	Ensures wedge clearance.	M
2O	Minimizes the number of osteotomies.	M
2P	Ensures sufficient bone overlap.	M
2Q	Minimizes bone protrusions, particularly on a 1cm part of the volar side of the radius and dorsal side of the ulna.	C
2R	Optimizes mesh overlap.	M
2S	Minimizes deviation of the centreline.	M
3	**Osteosynthesis plate optimization**	
3A	Allows bending of the plate.	C
3B	Ensures the plate does not interfere with the bone surface.	C
3C	Ensures the plate does not interfere with soft tissues.	C
3D	Ensures the plate does not interfere with joint movements.	C
3E	Centers plate positioning around the osteotomy site.	C
3F	Considers surgical approach preferences.	C
3H	Maximizes the plate's bone contact.	C
4	**Presentation of clinical parameters**	
4A	Displays ulnar variance.	S
4B	Displays radial length.	S
4C	Displays radial angulation.	S
4D	Displays dorsal angulation.	S
4I	Defines the bony landmarks with minimal manual annotation.	C
4J	Facilitates adjusting the annotation of bony landmarks.	M
4K	Automatically determines the anatomical axis.	C

Table 1: Functional requirements for the osteotomy planning addon, prioritised using MoSCoW classification. *Abbreviations:* STL, Standard Triangle Language; UI, User Interface;

#	Requirement	Priority
5	**Usability**	
5A	UI should be intuitive and easy to navigate, with a minimum user training requirement	S
5B	UI should allow users to adjust previously inserted values	S
5C	Users should be warned when performing actions risking loss of progress	S
5D	Users must be able to include or exclude constraints	M
5E	Users could be guided in altering optimization parameters within feasible boundaries	C
5F	Comprehensive user manuals should be provided	S
6	**Efficiency**	
6A	The application should use memory efficiently	C
6B	The run time should ensure good user experience	S
7	**Maintainability**	
7A	The code should follow best practices and standards for readability and maintainability	S
7B	Comprehensive documentation for the codebase and system architecture should be available	S
7C	The system should be modular to allow for easy updates and feature additions	S
7D	The system should include logging and monitoring capabilities to facilitate troubleshooting and performance tracking	S
7E	Error handling mechanisms should be established to manage and report system failures	C
8	**Software distribution**	
8A	The application must only use open-source resources	M
8B	The application should be easy to install on different environments with minimal configuration	S
9	**Backup and recovery**	
9A	The system should perform backups after each calculation or optimization step	S
9B	The system should be capable of restoring from backups	S

Table 2: Non-Functional requirements for the osteotomy planning addon, prioritised using MoSCoW classification. *Abbreviations:* UI, User Interface;