The Johns Hopkins University
EN.601.656 Advanced Computer-Integrated Surgery Course Project
Instructor: Russell H. Taylor

Real-time Integration of Fully Automatic 2D/3D Pelvic Registration with Robotic X-ray Acquisition

Group 7 Final Report

Jiaming Zhang
jzhan282@jhu.edu

Zhangcong She
zshe1@jhu.edu

Mentors: Benjamin Killeen; Prof. Mathias Unberath

Contents

1 Introduction

1.1 Background

2D/3D Registration is essentially solving the Perspective-n-Point (PnP) projection problem, which is the problem of estimating the pose of a calibrated camera given a set of n 3D points in the world and their corresponding 2D projections in the image [1]. The resulting camera pose consists of 6 degrees of freedom (DOF) which are made up of the rotation (roll, pitch, and yaw) and 3D translation of the camera with respect to the world. Figure 1 demonstrates the basic projection relationship between the coordinate frame of 2D images and a 3D body.

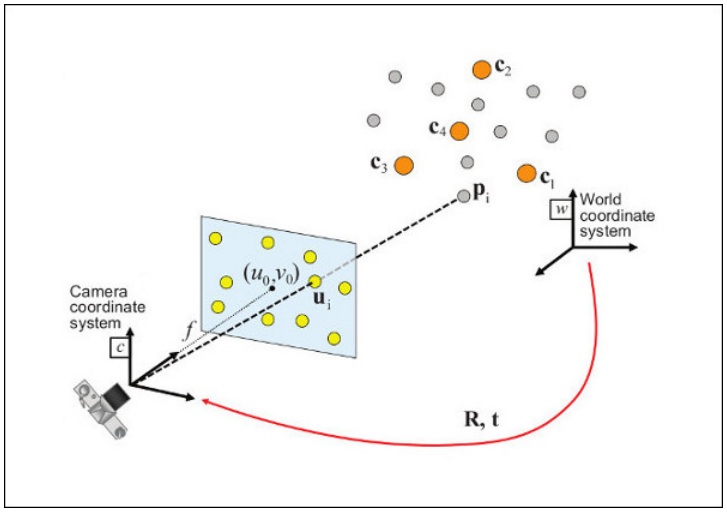

Figure 1: The projection of the reference points. [2]

Points expressed in the world frame X_w are projected into the image plane $[u, v]$ using the perspective projection model Π and the camera intrinsic parameters matrix A. The coordinates of the points on the projection plane are expressed by Eq. 1.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A\Pi\ {}^{c}T_{w} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{1}$$

$$A\Pi\ {}^{c}T_{w} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Registration is used to align the pre- and intra-interventional data both before and during an operation so that corresponding anatomical structures in the two data sets are aligned [3]. In other words, registration is concerned with bringing the pre-intervention data (patient's images or models of anatomical structures obtained from these images and treatment plan) and intra-intervention data (patient's images, positions of tools, radiation fields, etc.) into the same coordinate frame. Intraoperative 2D/3D registration is a commonly used technique in image-guided radiation therapy (IGRT), image-guided radiosurgery (IGRS), and image-guided minimally invasive therapy (IGMIT). for finding a rigid pose of a 3D image so that it aligns with the target 2D fluoroscopic image. Consequently, the rigid pose is used to determine the frame transformation between the preoperative image and the intraoperative patient's coordinate system [4] (see Figure 2).

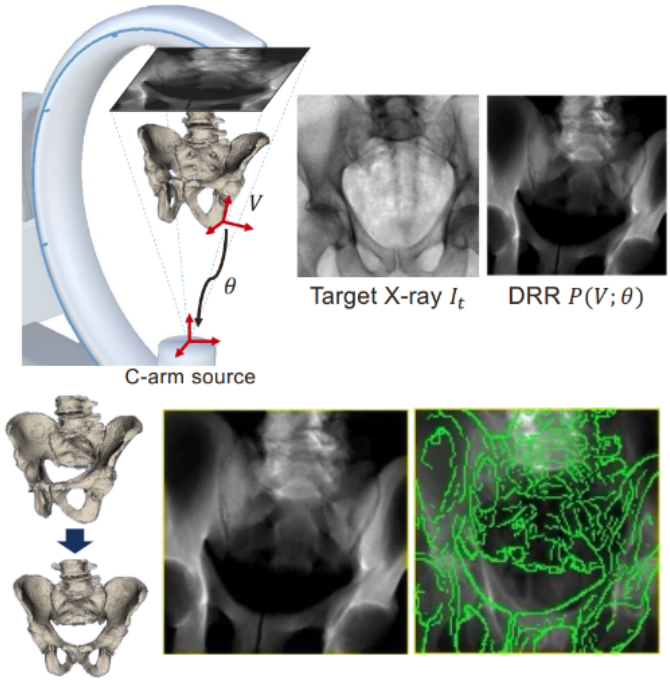

Figure 2: 2D/3D Registration for C-arm fluoroscopy and preoperative CT.

In image-guided minimally invasive surgery, the registration of preoperative and intraoperative data and instrument tracking provide surgeons with information about the current position of the instruments relative to the planned trajectory, nearby vulnerable structures, and the ultimate target. In image-guided endoscopy, 3D virtual images of the anatomy and pathology are generated from pre-interventional images and registered to real-time live endoscopic images to provide augmented reality which enables the display of anatomical structures that are hidden from the direct view by currently exposed tissues. In interventional radiology, registration of the pre-interventional image to the X-ray fluoroscopic or Ultrasound image allows visualization of tools, like catheters and needles, in 3D which can significantly improve guidance. In external beam radiotherapy, registration of planning CT images and daily pre-treatment images allow precise patient positioning, which is of utmost importance for exact dose delivery to the target and for avoiding irradiation of healthy critical tissue.

The methods for registering 2D/3D can be categorized based on the nature of their registration basis. Regardless of the strategy for achieving dimensional correspondence, 2D/3D registration methods can be classified as extrinsic, intrinsic, and calibration-based. The intrinsic methods are further classified as feature-, intensity- or gradient-based. Feature-based 2D/3D registration methods are concerned with finding the transformation that minimizes the distances between 3D features, extracted from the pre-interventional image, or a model, and corresponding 2D features. The features are geometrical entities like isolated points or point sets, forming a curve, contour, or surface. Extraction of geometrical features by image segmentation greatly reduces the amount of data, which substantially accelerates the speed. These geometrical features are called intensity features in some state-of-the-art registration solvers [5]. Intensity-based registration performs an optimization over the relevant pose parameters, using an objective function that compares simulated radiographs, commonly referred to as Digitally Reconstructed Radiographs (DRRs), with the intraoperative image. The optimization formula is expressed by Eq. 3

$$\{\theta_m, m \in \{0, ..., M\}\} = \arg\min_{\theta_m} \sum_{n=0}^{N} \mathcal{S}(I_n, \sum_{m=0}^{M} \mathcal{P}(V_m; \Theta_m)) + \mathcal{R}(\theta_m) \tag{3}$$

Here, V_m means a set of 3D volumetric data and I_m represents 2D X-ray images. θ_m is the object pose. $\mathcal{P}$ means the DRR projection operator; $\mathcal{S}$ means the similarity function;

$\mathcal{R}$ stands for regularizer function. The similarity function $\mathcal{S}$ can be determined by multiple features of the image, including anatomical landmarks, intensity features and gradient features [3]. Grupp utilized landmark positions in the initialization procedure and then solve the optimization problem using Covariance Matrix Adaptation: Evolutionary Search (CMA-ES), and Bounded Optimization by Quadratic Approximation (BOBYQA) [5] in his implementation of xReg. The intensity features are detected and visualized in Figure 3.

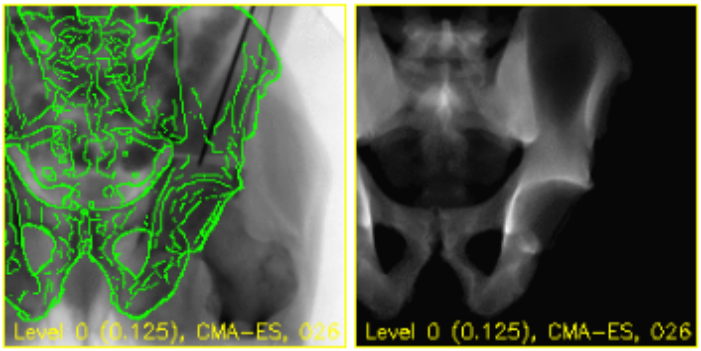

Figure 3: Detected intensity features of CT scan visualized on the X-ray (left) and generated DRR image (right).

With the evolution of deep learning methods, convolutional neural networks (CNN) have excelled in detecting anatomical landmarks on fluoroscopic images [6]. Fine-tuned models can produce satisfactory results with high precision in a short time compared to conventional manual methods. In addition, as the robotic imaging system develops, solving registration problems becomes more efficient since the robotic platform used to produce 2D X-ray images and 3D CT scans could properly initialize the searching bound for optimization using its inherent frame transform information. Loop-X is an advanced automatic robotic imaging system used for high-res X-ray and CT imaging tasks. However, developer tools for such robotic systems are not sufficient especially in terms of 2D/3D registration.

1.2 Significance

In minimally invasive surgery, clinicians use intraoperative fluoroscopy to overcome the occlusion and ascertain the poses of anatomy, surgical instruments, or artificial implants[7]. 2D/3D registration leverages fluoroscopy to align the preoperatively obtained CT scans to

the intraoperative patient's anatomy. Image-based 2D/3D registration is a fiducial-less approach [8].

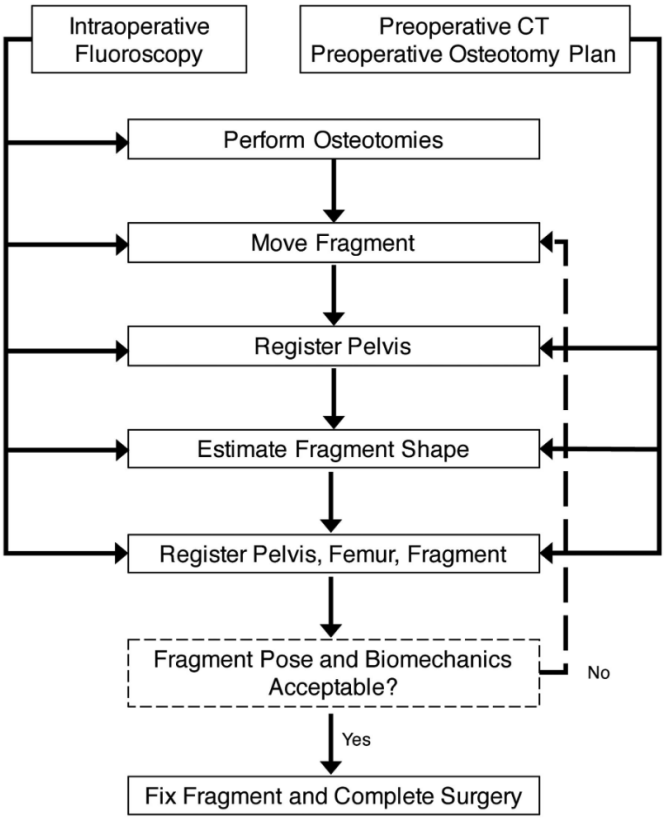

Figure 4: Level workflow detailing the steps performed during the surgery and the data required for each in periacetabular osteotomy fragments. The registration and shape estimation steps are the primary focus of this part [9].

Compared to the traditional 2D/3D registration method, image-based registration doesn't need physical contact, such as screws. During the operative process, it only needs real-time X-ray images, which would be used to solve the registration with the anatomical region of interest in the patient. In conclusion, this method greatly reduces the additional damage to the patient's body and shortens the preparation time for the operations. The registration process for X-ray images and CT scans is a challenging task that requires specialized knowledge and experience. It has traditionally been considered highly complex, and skilled personnel usually dedicate significant time to complete it. That problem is major because performing registration tasks relies on multiple subroutines, where packages or software

used in these subroutines may be not well-cooperated. One typical workflow of solving the registration problem is shown in Figure 4.

These packages are usually developed under various environments and are sometimes not compatible with each other. Some of them are poorly documented which makes users difficult to employ and maintain. The overall problems can make it challenging for surgeons to learn and use existing programs to solve 2D/3D registration problems during intraoperative procedures. Even though some researchers have proposed fully automated solutions, the existing software is closed source, making it hard to reconfigure and second-develop.

1.3 Goal

Based on the aforementioned motivations, this project aims at designing a new unified and modularized architecture in Python that provides developer interfaces for future modularization and development. We proposed that the users could modularize the optimum landmark-detector and registration-solver programs and solve the registration problem with simple command line inputs or code construction. By employing a modular design, we aim to make the registration process more accessible and efficient for surgeons, ultimately reducing the time and effort required to complete it.

In summary, this project seek to integrate the state-of-the-art 2D landmark detection model and advanced registration solver to perform intraoperative anatomy alignment tasks with the following features:

- The Python package of integration should be implemented on top of the proposed unified architecture and tested over sequential X-ray images and the corresponding annotated CTs for hip surgery.
- With the package, we hope to streamline the registration process for real-time intraoperative applications.
- As a course project for CIS 2, we have provided sufficient documentation and tutorials for the package so that other researchers and users could incorporate it in their own research projects and/or insert other advanced modules into the proposed package.

2 Technical Approach

2.1 Overview

As introduced in Section 1, the ultimate goal of this project is to implement a open source package to accelerate the process of 2D/3D registration. The package should be highly compatible and reconfigrable to adapt different submodule substitutions. It should be deployed via Python Package Index (PyPI) [10] and/or other Python package platform. PyPI is a central directory that hosts software packages developed for the Python programming language. PyPI serves as a distribution platform where developers can publish their Python libraries, frameworks, and tools, making them available for others to download and use in their own projects.

Limited by the course period, the package XREGI only support SyntheX and xReg as the landmark detector and registration solver respectively. The implementation is available on GitHub https://github.com/shez12/xregi and can be easily installed on any python environment using '*pip install xregi*' command. In the following sections, we will detail the theoretical and practical techniques used in XREGI.

2.2 2D Landmark Detection

Anatomical landmarks are biologically meaningful locations in anatomy that can be readily detected and enable correspondence between specimens and across domains. Landmark or key point detection is well understood in computer vision, where robust feature descriptors disambiguate correspondences between multiple 2D images, finally enabling purely image-based pose retrieval [11]. In this project, we rely on a well-established backbone CNN network, namely TransUnet, for landmark detection tasks. The TransUnet architecture is adjusted and fine-tuned to perform accurate inference on the given X-ray datasets.

Specifically, the model will first extract the segmentation features and then obtain a feature map where each channel estimates the heatmap of a landmark. Both segmentations and heatmaps are used to estimate the locations of anatomical landmarks in pixel coordinates. The estimated location of each landmark is defined as the candidate location with maximal heatmap intensity. Figure 5 gives an example of the process and results of

using TransUnet to detect the landmarks on a cadaveric pelvic.

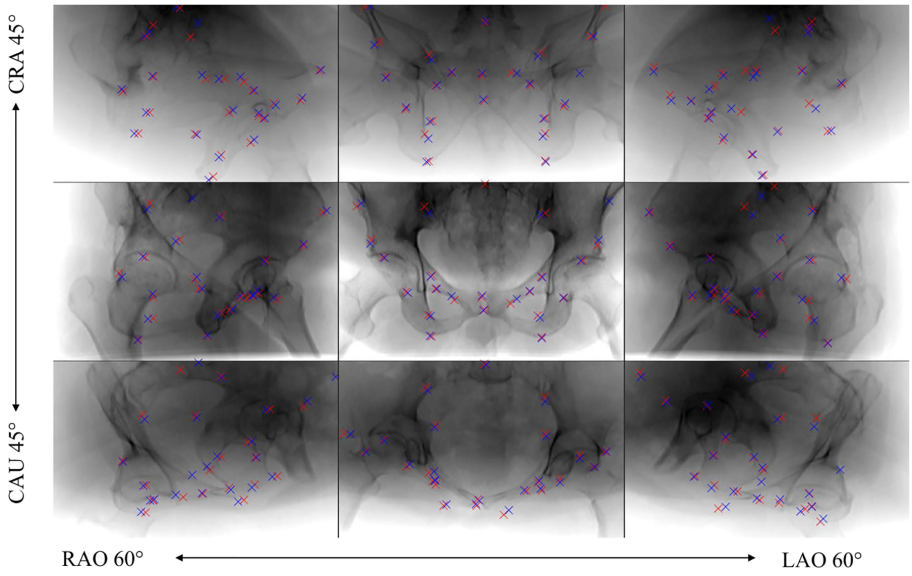

Figure 5: Predicted landmark positions for example projection images sampled across the sampled spherical segment of the synthetic test data set. Ground truth positions are marked with blue labels and automatic detection with red labels. Note that each projection image is processed independently [11]

To derive a well-trained model, Gao et al. [7] have proposed a synthetic data paradigm, SyntheX, to enrich the dataset and improve the model inference accuracy. The SyntheX framework is designed to develop AI algorithms for analyzing X-ray images in a way that can be applied to a variety of situations. The framework accomplishes this by using synthetic data that is generated from annotated computed tomography (CT). The process involves simulating X-ray image formation based on CT, and then training the AI models using domain randomization. The resulting models created by SyntheX are able to maintain their performance under domain shift, making them suitable for evaluation and deployment in real-world clinical X-ray settings. By using this approach, the SyntheX framework allows for the development of generalizable AI algorithms for X-ray image analysis that can be used in a variety of applications.

The source code of SyntheX is available on GitHub. In this project, we mainly use a pre-trained model proposed by SyntheX to find 2D landmarks on X-ray images.

2.3 3D Landmark Detection

Because our project focuses on the hip surgery process, we will introduce the method we use to annotate the 3D landmarks on the pelvis. Using the procedure described in [12], lower torso 3D CT scans are resampled to have 1mm isotropic spacing. Segmentations of the pelvis, femurs, and vertebrae are obtained semi-automatically. A total of 14 landmarks are manually annotated in 3D: the left and right (L./R.) centers of the femoral head (FH), L./R. greater sciatic notches (GSN), L./R. medial obturator foramen (MOF), L./R. inferior obturator foramen (IOF), L./R. superior pubis symphysis (SPS), L./R. inferior pubis symphysis (IPS), L./R. anterior superior iliac spine (ASIS). The 3D landmarks are illustrated in the Figure 6. In this figure, three views of the 6 anatomical structures and 14 landmarks to be annotated in 2D fluoroscopy are displayed. All landmarks are bilateral with left (L.) and right (R.) denoted.

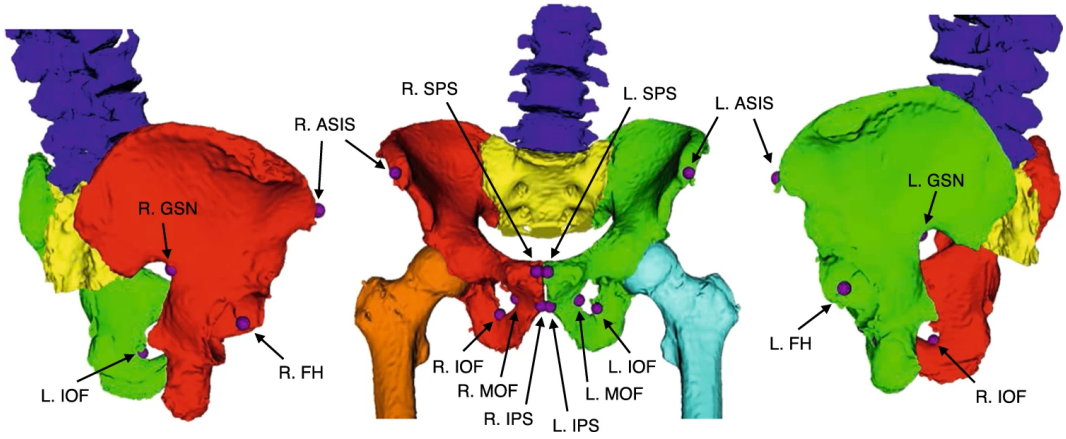

Figure 6: 3D landmarks annotated on he CT segmentation of lower torso. The L. hemipelvis is shown in green, the R. hemipelvis in red, L. femur in cyan, R. femur in orange, vertebrae in blue, and upper sacrum in yellow. Each landmark is overlaid as a purple sphere. [5]

2.4 2D/3D Registration

Our approach to 2D/3D registration of single-view fluoroscopy and CT builds upon the multiple-resolution, multiple-component, 2D/3D, intensity-based registration pipeline introduced in [5]. The pipeline solves the registration problem using xReg. xReg is a C++

library developed by Dr. Robert Grupp, with an online and intraoperative registration strategy. It leverages image intensities and CNN features to compute the registration matrix between the CT frame and the X-ray frame. In this project, the registration tool we use is designed to register a CT scan of a single 3D object, such as the pelvis, with a single 2D fluoroscopy view. The registration process involves two steps: first, the tool identifies anatomical landmarks in both the 3D volume and the 2D view and performs a paired point registration. Second, a multiple-resolution intensity-based registration is conducted, using the paired point registration results as an initial estimate.

Instead of treating landmark features and intensity features separately, the detected landmark locations may be incorporated into a robust reprojection regularizer for intensity-based registration. The regularizer is defined in Eq 4

$$\mathcal{R}(\Theta_P) = \frac{1}{2\sigma_l^2} \sum_{l=1}^{N_l} ||\mathcal{P}(p_{3D}^l; \Theta_P) - p_{2D}^l|| \tag{4}$$

As with the PnP approach, non-uniform patch weightings are applied using segmentation. Using one of the estimated 2D landmarks locations, the single landmark initialization is used to calculate an initial pose of the pelvis. The solver first uses a Covariance Matrix Adaptation: Evolutionary Search (CMA-ES) [13] optimization, followed by the Bounded Optimization by Quadratic Approximation (BOBYQA) [14] at a finer resolution without patch weightings or regularization.

After the program reaches the convergence, it saves the pose estimate to disk and optionally generates debug information for troubleshooting or verifying the success of the registration process. The registration result can be visually inspected by generating a projection figure using the visualization functionality provided by xReg.

2.5 Pipeline Workflow

As shown in Figure 7, our designed pipeline is composed of intraoperative and preoperative subroutines. First of all, in the preoperative period, our pipeline requires annotated CT scans of the anatomy involved in the operation. Users need to perform the segmentation and landmark detection process on the CT scan. Our package doesn't provide tools for segmentation in CT. Users are encouraged to employ Total Segmentator [15], a tool for robust segmentation of all important anatomical structures of the human body in CT

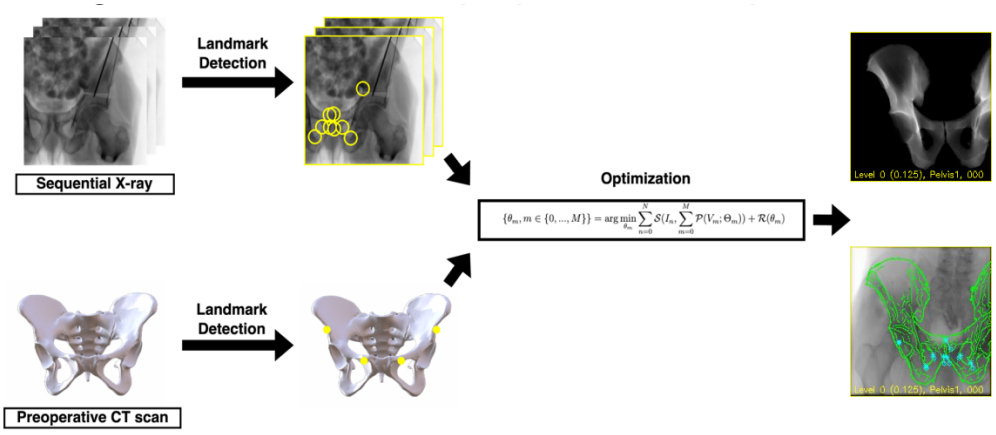

Figure 7: Pipeline workflow

images, to segment CT scans. Once complete the segmentation, users can annotate the 3D landmarks manual using the guidance described in Section 2.3. It is highly recommended that the manual landmark annotation is done by professionals because the coordinates of the landmarks would be used for obtaining the initial registration estimates of the pelvis.

Then, given X-ray images, 2D landmarks on the X-ray images, CT scan, the segmentation of the same CT, and 3D landmarks as input, xReg would act as registration solver to solve the projection matrix between CT frame and X-ray frame, also named as F^{CT}_{X-ray}. The result helps to reproject the mesh of the anatomy (generated from corresponding CT scans) on the input X-ray images to visually inspect the quality of the alignment (see Figure 8).

2.6 XREGI Program Architecture

With the proposed pipeline workflow, we have implemented a package to copy with the registration problem. As we mentioned in previous sections, we aim to provide a new unified and modularized architecture in Python that gives developer interfaces for future modularization and development. To modularize our program, our package is formulated as a combination of three abstract classes, **LandmarkDetector**, **RegistrationSolver** and **Regi2D3D**. Each abstract class represents a subroutine involved in the 2D/3D registration task.

LandmarkDetector class is designed to process X-ray images. It has been pro-

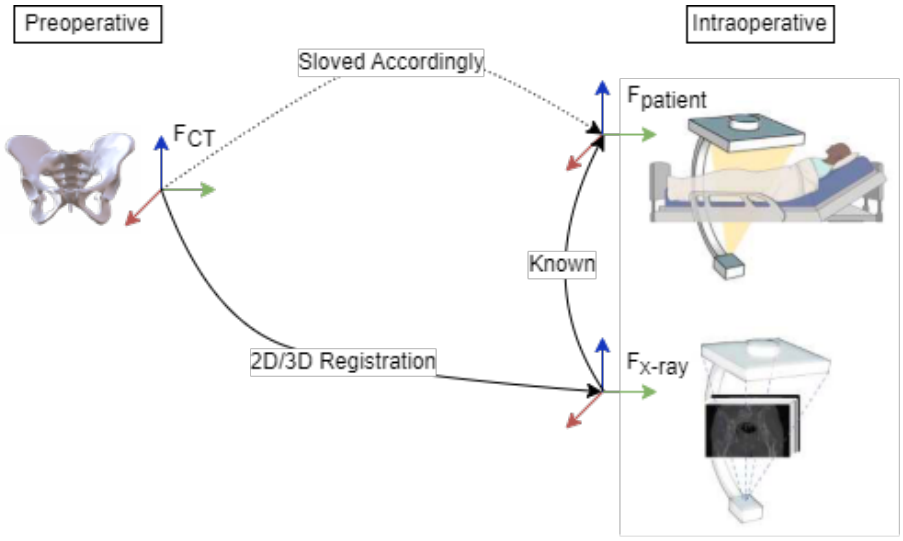

Figure 8: Coordinates system relationships between intraoperative patients' anatomy, intraoperative X-ray images and preoperative CTs.

grammed to require only image data as input, usually in the form of a NumPy array. The expected output of this process is 2D landmarks which are saved in a CSV file for further use. The primary goal of this class is to accurately identify and locate key landmarks within X-ray images, providing essential data for subsequent analysis.

In our implementation, we inherit the LandmarkDetector class with the features developed in SyntheX. First, a well-trained model (SyntheX checkpoint file) is loaded to the LandmarkDetector abstract class. And then, in LandmarkDetector class, we choose to load SyntheX as the detection method. At this point, the checkpoint file would be uploaded to the main program and the class would begin to read the X-ray images in a given path. Due to the modularized structure of our program, we could substitute or add extra X-ray images without loading the SyntheX checkpoint file again, which reduces the performing latency.

Nevertheless, users have the flexibility to load their desired detector program onto this component through inheritance. Other programs may use their own checkpoint models or unique input so that we create a config file that supports users to modify our program's input path configuration.

RegistrationSolver class allows for the seamless integration of various registration solver functions or programs. Typically, the inputs required for this process include X-ray

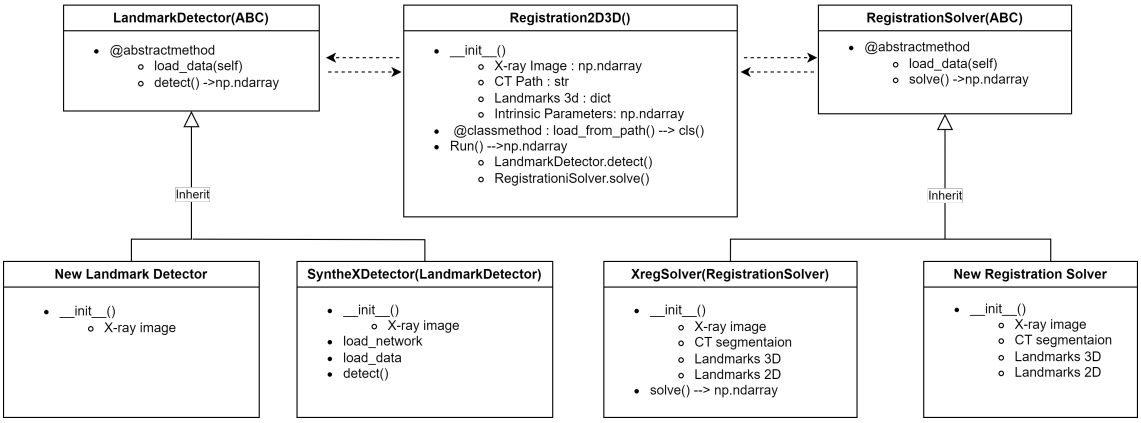

Figure 9: Program architecture

images, 2D landmarks, CT scans, 3D landmarks, and CT segmentations. The output of the RegistrationSolver class is the projection matrix between the CT and X-ray images, providing valuable insights for further analysis. Our package inherits the class based on the features of xReg. As mentioned in Section 2.4, xReg is a C++ library. Here we launch the binary files built from C++ source code using the *subprocess* package in Python to integrate the xReg functionalities.

When running our package, all mandatory data is first processed through **Regi2D3D**, which seamlessly transfers the data to its respective components. Regi2D3D will call the functionalities sequentially according to our workflow. It manages the inputs and the outputs of the other modules and processes them correspondingly. For example, the landmarks detected by the SyntheX detector are arranged in pixel coordinates in a downscaled range. The coordinates array needs to be transposed and scaled back in order to represent the Cartesian coordinates on the corresponding X-rays.

In addition to serving as a "manager" for our detector and solver components, the Regi2D3D class also provides users with flexible interfaces to choose from different registration methods that were loaded in the previous two classes without changing the basic workflow. This highly modularized design has made the 2D/3D registration problem accessible and easy to reconfigure.

2.7 Data Exchange

To facilitate the data exchange for the aforementioned modules, we employed a special data structure as the container to hold and transfer the data. The data container is designed as a Hierarchical Data Format (HDF). HDF file uses a "file directory"-like structure that allows to organize the data within the file as an embedded dictionary. For the input of the landmark detector, the *detector_input.h5* stores a set of raw X-ray images and the name of the landmark that we want to locate. For the input of the registration solver, *solver_input.h5* contains the X-ray images, the pixel coordinates of the detected 2D landmarks, intrinsic camera parameters, and image dimensions. With proper encoding and decoding processes, which are all provided in our package, the intermediate data could be efficiently managed and utilized. Besides, we have provided several overloaded data import functions, including loading data from memory and loading data with file paths, so as to support different data-loading requirements.

Besides, we provided a data structure called **LandmarkContainer** to cope with the sophisticated naming issues for the landmarks. The landmark container stores the landmarks in a way that the values and their label names are aligned as key-value pairs as in the dictionary. And the users can retrieve any stored landmarks with specified naming conventions. For example, SPS-l is interpreted as "{ANATOMYNAME} {Delimiter} {Bilateral Side}" pattern and R.gsn is identified as "{Bilateral Side} {Delimiter} {Anatomyname}". As a result, users are able to define their own naming patterns and convert the patterns to the required format of other modules with the help of **LandmarkContainer**.

2.8 Example Case: SyntheX and xReg

To illustrate the versatility of our **LandmarkDetector** and **RegistrationSolver** components, we have successfully integrated SyntheX and Xreg into the respective modules. Despite SyntheX being an AI-based model that requires a well-trained checkpoint model as input, the output of SyntheX is still 2D landmarks. This implies that any landmark detector program with image data input and 2D landmark output can be integrated seamlessly with our program.

2.8.1 Run SyntheX to Detect Landmarks

Here we present one example to demonstrate the accessibility of our package. Two interaction methods are supported in the existing framework, namely command line interactions and class object instantiations. Command line interactions basically require the user to specify the path of the input files. The files used as an example include:
the path of the input xray image

```
\User\xregi\data\xray_01.png
```

the path of the pre-trained model checkpoint file

```
\User\xregi\data\checkpoint001.pt
```

two SyntheX input h5 files are

```
\User\xregi\data\SyntheX_input.h5
\User\xregi\data\SyntheX_label_input.h5
```

Therefore, we can run the LandmarkDetector part (SyntheX version) using

```
python test_ensemble.py \User\xregi\data\SyntheX_input.h5
\User\xregi\data\SyntheX_label_input.h5
--pats 1
--nets \User\xregi\data\checkpoint001.pt
```

and the path of the ouput 2D landmarks file is designed to be

```
\User\xregi\data\2d_landmarks.csv
```

2.8.2 Run xReg to Solve Registration

In xReg, 2D/3D registration solver requires a common set of inputs, including X-ray images, X-ray landmarks, CT scans, CT segmentations, and 3D landmarks. X-ray image data is in the form of a numpy array, whereas CT scans and CT segmentations are in the nii.gz format, and CT landmarks are typically in fcsv format. From previous paragraph, we know the X-ray file path and its landmarks file path.
Additionaly, the path of the CT scan is

```
\User\xregi\data\ct_01.nii.gz
```

The path of the CT segmentation is

```
\User\xregi\data\ct_01_seg.nii.gz
```

The path of the 3D landmarks is

```
\User\xregi\data\ct_01_landmarks.fcsv
```

and we would read and save *2d_landmarks.csv* and *xray_01.png* in form of h5 format to fit xreg requirement in the following path

```
\User\xregi\data\xray_01.h5
```

Finally, the RegistrationSolver part (Xreg version) can be run by

```
xreg-hip-surg-pelvis-single-view-regi-2d-3d \User\xregi\data\ct_01.nii.gz
\User\xregi\data\ct_01_landmarks.fcsv
\User\xregi\data\xray_01.h5 output1.h5
output2.h5 -s \User\xregi\data\ct_01_seg.nii.gz
```

Here *output1.h5* and *output2.h5* are output files about computed projection matrix and debug information created by xReg program.

2.8.3 Run with "One Click"

Then, in Regi2D3D, we load SyntheXDetector and XreSolver. Given proper input, the projection matrix will be computed. Overall, our final command to run the example case would be :

```
\User\xregi$ python -m xregi --xray data\xray_01.png --ct data\ct_01.nii.gz
--landmarks data\ct_01_landmarks.fcsv --seg \data\ct_01_seg.nii.gz
```

The users are also welcomed to utilize the features directly through import xregi package into their code implementation. An example is:

```
import xregi

reg = xregi.Regi2D3D()
reg.load(xray_images, CT, CT_segmentation, 3D_landmarks)
projection_matrix = reg.run()
```

3 Result and Discussion

Supported XREGI features will be thoroughly presented, followed by an evaluation section using the cadaveric dataset for single-view 2D/3D registration tasks. All the evaluation process is done in a Ubuntu 20.04 machine, with GTX 1080 GPU. The results are demonstrated in this section.

3.1 Supported Features

For the **SynthexDetector** class, we developed several class members and properties:

- **load()**: load() method allows the users load the data both through memory and local folder by specifying the file name.
- **xray_image, landmark2d**: The loaded data will be handled as the class property so that every part of it can be easily substituted in the run-time, facilitating the real-time applications in the future.
- **detect()**: This function will load the model to construct the network beforehand to speed up the inference process. It also supports substitute the pre-trained model in the run-time.

Like the detector class, the **xregSolver** class also supports multiple data loading ways using the class method **load()**. Moreover, the class method **run()** supports three different arguments to calculate the result, generate the video and render a projection scene respectively.

For the **LandmarkContainer** class, we designed the class properties as

- names: the name of the landmarks, stored as the convention used in [5]
- values: the location of the landmarks
- dimension: indicating whether the landmarks is 2D or 3D
- **get("Template Name")**: this method allows the user to extract the landmarks and their corresponding values with the pattern specified in the "Template Name"

Other involved arguments for running SyntheX and xReg are defined in a configuration file. Most of these arguments are the default paths of files that are created while running the programs. For example, the computed 2D landmarks will be stored in a csv file, which is designed to have default name and default path. We propose that these arguments can be modified to fit the input for other landmark detection and registration solver programs.

3.2 Landmark Detection Tasks

In our experiments on pelvic registration, we utilized sequential X-ray images and highlighted computed landmarks on one example image (as shown in Figure 10). Overall, the landmark detection process worked well, with most of the landmarks being located correctly. In most cases, through manual inspection, the total number of detected landmarks is slightly less than the total number of visible landmarks. We have tried several threshold values until the inference result is satisfactory. Figure 11 shows how the number of detected landmarks varies along with the threshold values.

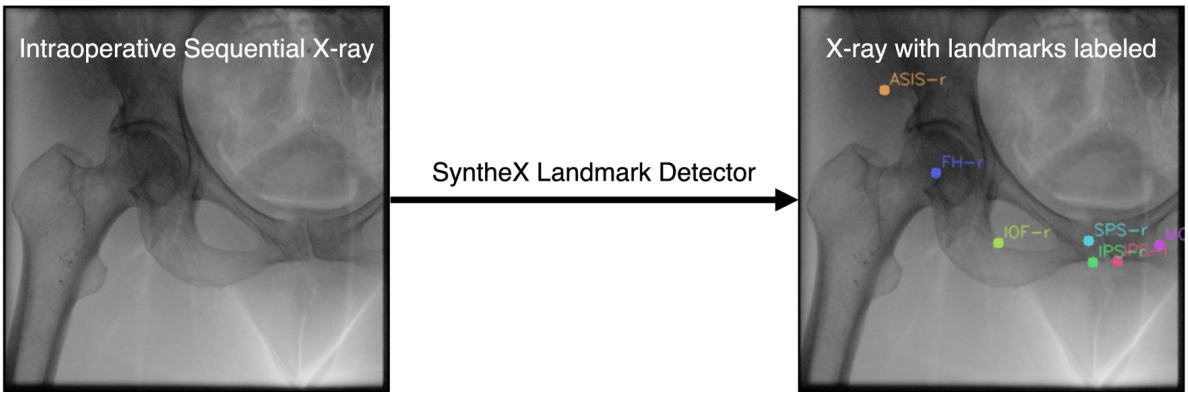

Figure 10: SyntheX Landmark Detection

Even though the threshold arguments fine-tuned, we still encounter some difficulty with the ASIS-r landmark. This phenomenon might be caused by the shade of the surgical tools and/or the occulsions of other anatomies. It should be noted that this discussion is focused on the workflow, and we will not delve into the accuracy of the SyntheX output in this context. For more detailed landmark data, please refer to the appendix. On the other hand, We have tried several images with different resolutions, such as 360 * 360, 768*768,

and 1536*1536. All of them spend less than 4 seconds with exact the same (L2-error is equal to or smaller than 1 pixel) results.

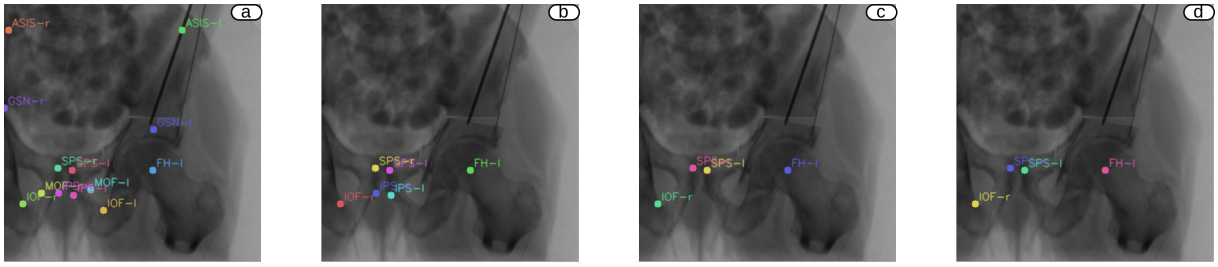

Figure 11: Landmarks With Different Threshold. (a) has a threshold of 0.01; (b) has a threshold of 0.5; (c) has a threshold of 0.65; (d) has a threshold of 0.7. The number of the detected landmarks reduces as we increase the threshold. However, there exist misdetection when the threshold is too low.

3.3 Solving Projection Matrix Tasks

Figure 12 shows the result of the registration solver of xReg. The intensity features are projected onto the X-ray images to help the user verify whether the registration result is sufficiently correct. The figure proves that our workflow performs well under the experiment setups.

To obtain correct results, we have tried different combinations of the key parameters used in the optimization process. Besides, we have found out that xReg only takes L./R.IOF, L./R.ASIS, and L./R.FH as its optimizing targets. Among the key parameters involved, the extrinsic parameter of the camera affects the result most significantly. Bad assignment of the extrinsics could lead to a poor initialization for the bound of the searching area, which eventually causes failure or completely wrong results.

Again, as our objective is to provide an integration of xReg, the quality of the output of the package will not be discussed here. Nevertheless, with the package implemented, the performance of xReg and other potential methods can be compared and evaluated by a controlled dataset with known ground-truth.

Once the registration problem has been computed, we utilize the estimated registration pose to render the pelvis with respect to the projective coordinate frame. The result of

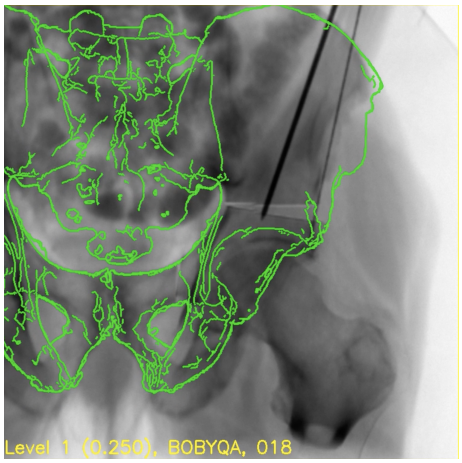

Figure 12: Registration result: The intensity features are projected back on the X-ray images. This movie demonstrates how the optimizer converges upon iterations.

this process is shown in the Figure 13. The successful rendering of the pelvis with respect to the projective coordinate frame demonstrates the versatility of our approach.

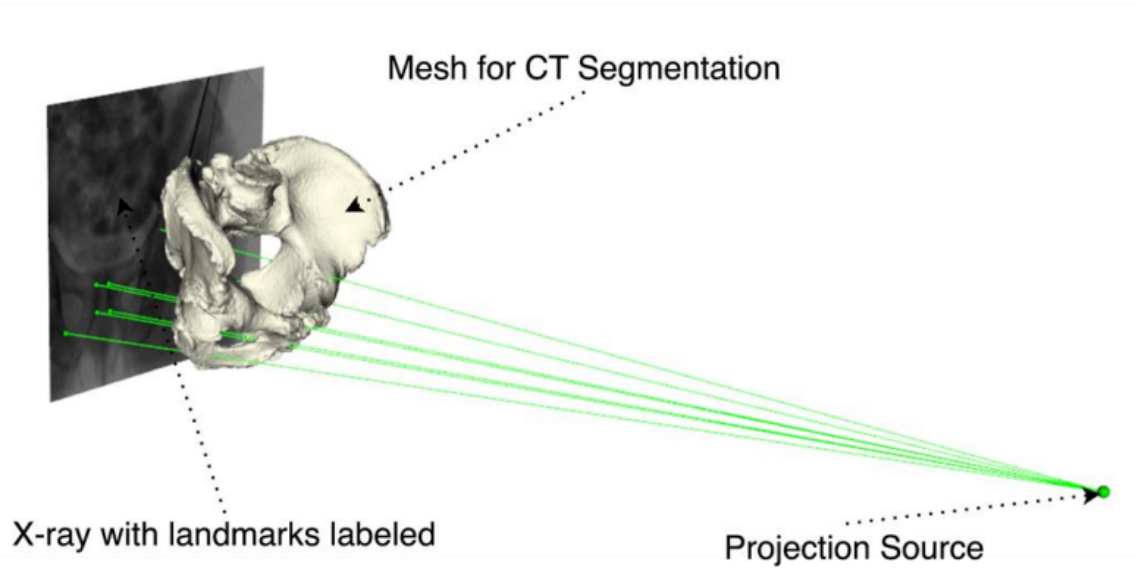

Figure 13: Projection visualization

3.4 Conclusion and Future Work

In conclusion, our key contributions in this project includes:

- We have successfully built a new unified and modularized architecture in Python that provides developer interfaces for future modularization and development.
- By integrating SyntheX and xReg into our program, we have demonstrated our program's compatibility and effectiveness, and have highlighted the potential for its use in a wide range of medical imaging applications.
- The package can be easily installed and utilized with simple terminal commands, making it accessible to users of different levels.

To ensure that users can effectively leverage the package to accelerate their research, we have created detailed documentation for both SyntheX and XREGI. This documentation provides users with the necessary resources to effectively use the package.

In the future, we will keep maintaining XREGI and make it more versatile. We seek to incorporating our implementation with robotics medical imaging platforms to realize real-time image acquisition and registration. Moreover, we plan to employ the registration result derived by XREGI in mixed reality area, such as the use of HoloLens, to facilitate its clinical applications in intraoperative processes. Additionally, we plan to submit our work to a peer-reviewed conference or journal to further validate the effectiveness and potential of our package.

4 Project Management

4.1 Dependencies

The project's dependencies are depicted in Figure below, along with the estimated resolving time for each. To mitigate the risk of original plans failing, alternative plans have been proposed for each dependency. All dependencies are succesfully solved before the deadline.

	Need	Followup	Contingency Plan	Planned	Hard DL	Status
MOCK OR Lab Access	manipulate Loop-X	N/A	Ask Benjamin for access	Feb 06	Feb 10	✓
Loop-X	Generate X-ray and CT Scan	N/A	Ask Benjamin for access	Feb 06	Feb 10	✓
SyntheX	Generate Domain Generalized X-ray Open source github repository	N/A	Request the source code from Dr. Cong Gao	Feb 01	Feb 06	✓
Model Checkpoint	Hyper parameters of SyntheX On private onedrive folder	Keep secured		Feb 08	Feb 12	✓
xReg	Compute registration parameter between CT scan and X-ray Image, open source github repository	N/A	Request the source code from Dr. Grupp	Feb 08	Feb 12	✓
CT DataSet	As a input used in Xreg	Keep secured	N/A	Feb 06	Feb 20	✓
Total Segmentator	Do CT Scan segmentation	Downloaded	N/A	Feb 18	Feb 20	✓
Computers	Our own computer with an environment for software development	Ready to use	"PACKMAN" ARCADE Server	Jan 23	N/A	✓

Figure 14: Dependencies

4.2 Project Deliverable

The project's deliverables are shown below. By the end of semester, all minimum deliverables and most expected deliverables are completed. Due to lack of time, the maximum deliverable is not able to be achieved. All code and its documentation are avaiable in the github page.

	Deliverable	Key Milestones	Status
Minimum	Documentation for SyntheX, provide applicable interfaces	Documentation and well-organized repository for SyntheX	☑
	Automatic Data Acquisition script	Program or device that can extract image data from Loop-X	☑
	A well-documented program integrating previous works	Python Scripts incorporates previous works	☑
Expected	validation of integration program on cadaveric images	Test of program on dicom file from loop-X	☑
	Fully constructed software architecture	Reconstruction of python scripts for future users' modification	☑
	A python pip package for integration software	A python installation package	☑
	A user-friendly GUI	A GUI designed to import input and run the program	☑
	A fully automatic pipeline	well-designed, documented, and automatic pipeline	☑
	~~A view-rendering application for projective visualization~~	~~simulated scene of projective visualization in unity~~	☒
	A report for Validating our application on cadaveric images	well-analysed report about validating program	☑
Maximum	~~Integration with mixed reality visualization of relevant anatomy~~	~~simulation app executable in HMD~~	☒

Figure 15: Deliverables

4.3 Credit and Team Management

Group 7 is consists of 2 team members and two mentors:

- Jiaming Zhang, Group Member, M.S.E. in Robotics
- Zhangcong She, Group Member, M.S.E. in Mechanical Engineering
- Benjamin Killeen, Mentor, Ph.D. candidate in Computer Science
- Mathias Unberath, Mentor, Assistant Professor in the Department of Computer Science

Jiaming is mainly responsible for RegisrationSolver class programming, integration of Xreg and several utility functions that deal with format issues.Zhangcong is mainly in charge of LandmarkDetector class programming, SyntheX integration and several utility functions. Both team members equally contributes to maintaining the repository, the wiki and writing up the documentations.

The team members meet twice a week every Tuesday and Friday to synchronize the progress and make for the next step. Mentor's meeting is held every Monday to present

the outcomes and discuss the unresolved issues. The communications are done via Discord and Email. The source code and documentation are hosted on GitHub.

4.4 Lessons Learned

Having comprehensive documentation is essential for facilitating understanding and collaboration on related projects. Comprehensive documentation provides users with a clear understanding of the capabilities and limitations of a particular software or tool, allowing them to use it more effectively and efficiently.

In addition, having a clear list of dependencies and detailed timeline can indeed help a project proceed smoothly and efficiently. By clearly identifying the dependencies required for a software or tool, developers can ensure that all necessary components are installed and configured properly, reducing the risk of errors or compatibility issues.

5 Acknowledgements

We thank Dr. Cong Gao and Dr. Robert Grupp for making the source code of SyntheX and xReg public.

We thank Prof. Russel Taylor and Prof. Mathias Unberath for their invaluable help and guidance.

We want to convey our special gratitude to Benjamin Killeen for his exceptional guidance and support throughout our CIS 2 project.

6 Appendix

Landmarks name	row	col	Inference Time (sec)
FH-l	294	355	0.0091
FH-r	295	76	0.0077
GSN-l	213	321	0.012
GSN-r	213	101	0.0271
IOF-l	357	294	0.0198
IOF-r	257	140	0.01
MOF-l	351	260	0.0099
MOF-r	349	170	0.0356
SPS-l	336	232	0.0184
SPS-r	333	203	0.0333
IPS-l	359	228	0.029
IPS-r	359	203	0.0218
ASIS-l	184	334	0.0471
ASIS-r	189	32	0.0094

Table 1: 2D landmarks

References

[1] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[3] P. Markelj, D. Tomaževič, B. Likar, and F. Pernuš. A review of 3D/2D registration methods for image-guided interventions. *Medical Image Analysis*, 16(3):642–661, April 2012.

[4] Cong Gao. *Fluoroscopic navigation for robot-assisted orthopedic surgery.* Phd dissertation, August 2022.

[5] Robert B. Grupp, Mathias Unberath, Cong Gao, Rachel A. Hegeman, Ryan J. Murphy, Clayton P. Alexander, Yoshito Otake, Benjamin A. McArthur, Mehran Armand, and Russell H. Taylor. Automatic annotation of hip anatomy in fluoroscopy for robust and efficient 2D/3D registration. *Int J CARS*, 15(5):759–769, May 2020.

[6] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L. Yuille, and Yuyin Zhou. TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation. 2021. Publisher: arXiv Version Number: 1.

[7] Cong Gao, Benjamin D. Killeen, Yicheng Hu, Robert B. Grupp, Russell H. Taylor, Mehran Armand, and Mathias Unberath. Synthetic data accelerates the development of generalizable learning-based algorithms for X-ray image analysis. *Nat Mach Intell*, 5(3):294–308, March 2023.

[8] Y. Otake, M. Armand, R. S. Armiger, M. D. Kutzer, E. Basafa, P. Kazanzides, and R. H. Taylor. Intraoperative Image-based Multiview 2D/3D Registration for Image-Guided Orthopaedic Surgery: Incorporation of Fiducial-Based C-Arm Tracking and GPU-Acceleration. *IEEE Trans. Med. Imaging*, 31(4):948–962, April 2012.

[9] Robert B. Grupp, Rachel A. Hegeman, Ryan J. Murphy, Clayton P. Alexander, Yoshito Otake, Benjamin A. McArthur, Mehran Armand, and Russell H. Taylor. Pose Estima-

tion of Periacetabular Osteotomy Fragments With Intraoperative X-Ray Navigation. *IEEE Trans. Biomed. Eng.*, 67(2):441–452, February 2020.

[10] Python package index - pypi.

[11] Bastian Bier, Florian Goldmann, Jan-Nico Zaech, Javad Fotouhi, Rachel Hegeman, Robert Grupp, Mehran Armand, Greg Osgood, Nassir Navab, Andreas Maier, et al. Learning to detect anatomical landmarks of the pelvis in x-rays from arbitrary views. *International journal of computer assisted radiology and surgery*, 14:1463–1473, 2019.

[12] Shun Miao, Z Jane Wang, and Rui Liao. A cnn regression approach for real-time 2d/3d registration. *IEEE transactions on medical imaging*, 35(5):1352–1363, 2016.

[13] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

[14] Michael JD Powell. The bobyqa algorithm for bound constrained optimization without derivatives. *Cambridge NA Report NA2009/06, University of Cambridge, Cambridge*, 26, 2009.

[15] Jakob Wasserthal, Manfred Meyer, Hanns-Christian Breit, Joshy Cyriac, Shan Yang, and Martin Segeroth. Totalsegmentator: robust segmentation of 104 anatomical structures in ct images. *arXiv preprint arXiv:2208.05868*, 2022.