ZURICH UNIVERSITY OF APPLIED SCIENCES

MASTER THESIS

Automated image labeling and marker-less deep-learning-based 3D detection for surgical wires

Author:
Loran Avci [1]

Supervisors:
Prof. Dr. Helmut Grabner [1]
Marco Von Atzigen [2,3]

[1]Institute of Data Analysis and Process Design, Zurich University of Applied Sciences, Zurich, Switzerland
[2]Research in Orthopedic Computer Science, Balgrist University Hospital, University of Zurich, Zurich, Switzerland
[3]Laboratory for Orthopaedic Biomechanics, ETH Zurich, Zurich, Switzerland

Submitted in fulfillment of the requirements
for the degree of
Master of Science in Engineering

August 9, 2022

Declaration of Authorship

Master Thesis at the School of Engineering

By submitting this Master thesis, the undersigned student confirms that this thesis is his/her own work and was written without the help of a third party. (Group works: the performance of the other group members are not considered as third party). The student declares that all sources in the text (including Internet pages) and appendices have been correctly disclosed. This means that there has been no plagiarism, i.e. no sections of the Master thesis have been partially or wholly taken from other texts and represented as the student's own work or included without being correctly referenced. Any misconduct will be dealt with according to paragraphs 39 and 40 of the General Academic Regulations for Bachelor's and Master's Degree courses at the Zurich University of Applied Sciences (Rahmenprüfungsordnung ZHAW (RPO)) and subject to the provisions for disciplinary action stipulated in the University regulations.

Schaffhausen,August 9, 2022, Loran Avci

Acknowledgements

I express my deepest gratitude to Marco von Atzigen, who has supported me with his knowledge and expertise in executing this thesis. Without his patience and feedback, completing this work would not have been possible.

Thanks are also due to Dr. Helmut Grabner, who supported this work both technically and morally.

Special thanks to the Balgrist university hospital and the ROCS team for providing the infrastructure and know-how required for this thesis.

Abstract

Augmented reality in surgery is becoming increasingly essential to support surgeons. Larger surgical tools are already detected to support the surgeon's work, but smaller objects are still mostly neglected. Surgical wires, called K-wires, are used in various surgical disciplines. The tracking of K-wires would thus facilitate computer-assisted surgical navigation. We propose a novel markerless concept for tracking K-wires that can be used for surgical navigation. Our system generates an unlimited amount of annotated stereo image data quickly. By applying deep learning methods, the acquired data can be leveraged to detect the K-wires. The system has been developed and evaluated on a lumbosacral spine phantom. Our method achieved a mean error of 6.50 ± 2.30 mm and 6.40 ± 3.69° in the marker-less ground-truth generation. The deep learning-based markerless K-wire detection for computer-generated images achieved a mean error of 6.11 mm ± 1.33 mm to the ground truth labels. Our results show that our markerless image labeling method can be used in the future to generate large amounts of data with good quality. Although we could not evaluate the detection of K-wires with the recorded data due to inadequate stereo encoding, we have shown that our stereo neural network performs well in marker-free detection for computer-generated images.

Contents

Chapter 1

Introduction

The use of augmented reality (AR) in surgery and the training of surgeons with the help of virtual reality (VR) is currently receiving considerable attention (University Hospital, 2022). In previous research, larger surgical tools such as drills could be detected using deep learning methods (Koskinen et al., 2022). The tracking of surgical tools (cf. fig. 1.1) is an important part of surgical navigation, while the placement of pedicle screws is one of the most common areas of surgical navigation in spinal surgeries (von Atzigen et al., 2022; Liebmann et al., 2019). Smaller objects such as pedicle screws could be detected by combining deep learning methods and the Microsoft HoloLens (von Atzigen et al., 2022). This combination is less expensive than conventional tracking hardware and mitigates the line-of-sight problem since a HoloLens is worn directly by a surgeon. Current **surgical navigation** methods require high-end stereo cameras and physical markers that must be sterilized and calibrated for each surgical procedure (Hofstetter & Nolte, 1999; Siston et al., 2007). Furthermore, cameras need direct access to the surgical field to ensure continuous marker detection. These limitations reduce the clinical acceptance of navigation-assisted surgery.

Kirschner wires, called **K-wires**, are surgical wires often used in orthopedic and plastic surgery. The application possibilities of these K-wires are very versatile. Using K-wires in fracture fixation is associated with features such as simplicity, speed, and low cost (Kang et al., 2015). Due to their properties, K-wires are used in various standard surgical treatments for procedures on the spine, knees, and other joints (Ref: 99.MY46C.12; 99.81.12US; 99.101.12US; Medacta SA, Switzerland). The detection of thinner objects in the operating room has not yet been addressed in detail. Nevertheless, surgical guidewires have already been tracked using optical markers(Kriechling et al., 2020).

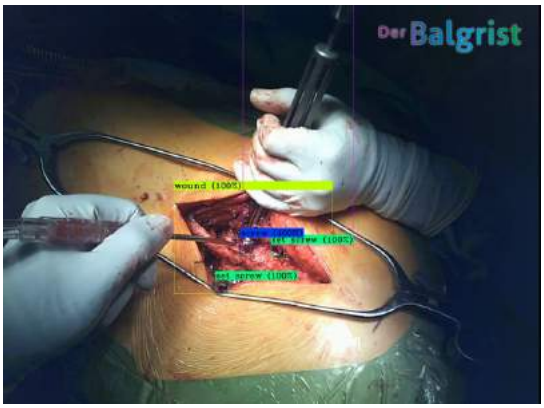


FIGURE 1.1: The illustration shows a conceptual design for detecting surgical tools in the surgical field during computer-assisted surgical navigation. (Image Ref. (Balgrist, 2022))

The overall incidence of complications for **spinal surgery** is high, at about 15% (Nasser et al., 2010). Because of the increased risk of injury, spine surgery was one of the first disciplines to use surgical navigation to operate more safely (Merloz et al., 1998). K-wires are used for temporary or definitive osteosynthesis, temporary joint fixation, or as guidance for other implants such as cannulated screws (Huber, 2008).

Different **marker-free** methods have recently been introduced. These include marker-free surgical navigation for spinal rod bending implants using stereo images or marker-free registration (von Atzigen et al., 2022; von Atzigen et al., 2021) and bone tracking in orthopedic surgeries where RGB-depth images were used as input instead of stereo images (H. Liu & y Baena, 2020). In our work, we prefer stereo images to RGB-depth images, as they have already produced more promising results (von Atzigen et al., 2022; von Atzigen et al., 2021). Furthermore, we suppose detecting K-wires with RGB-depth data is difficult because they are mostly smooth, have no texture, and are very narrow.

Object recognition and detection has been a highly researched area in computer vision for many years and has therefore already produced several approaches that could contribute to tackling the problem of this thesis. In the past, convolutional neural networks (CNN) (Lecun et al., 1998) have achieved great success in image recognition. Examples for object detection algorithms are, Faster R-CNN (Ren et al., 2015), Single shot multibox detector (W. Liu et al., 2016) and YOLO (Redmon et al., 2015; Redmon & Farhadi, 2016, 2018). Deep learning methods are already used in the medical environment to classify anatomical structures and surgical actions in gynecologic surgery (Petscharnig & Schöffmann, 2018), tool segmentation in stereo endoscopic surgeries (Mourgues et al., 2001), and self-supervised depth estimation in robotic surgery (Ye et al., 2017). All these deep learning models are highly data-intensive. These models are trained with hundreds of thousands of images. Therefore, a big focus of this work is the **automated generation of labeled data**.

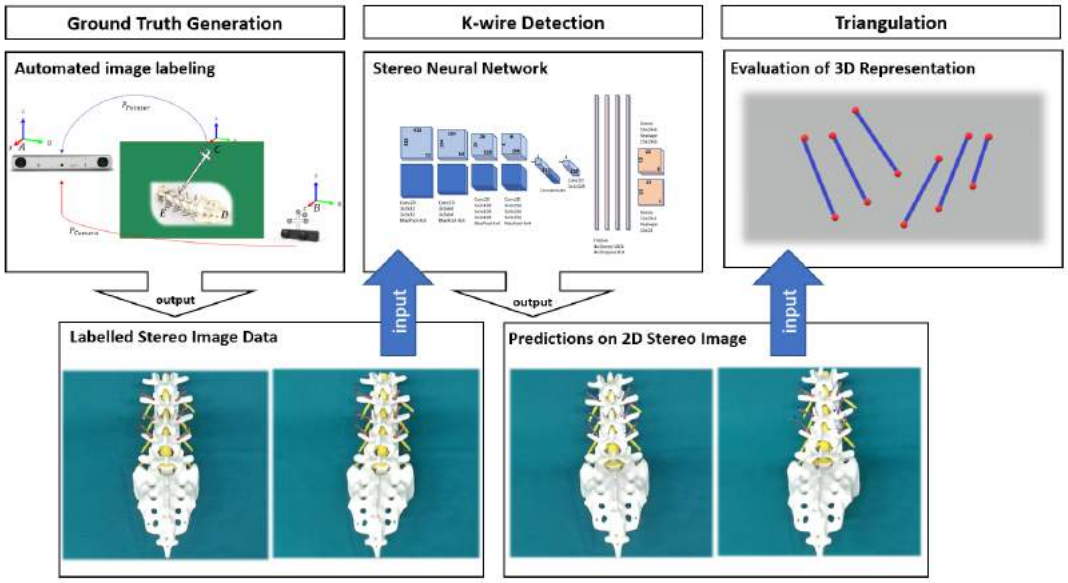


FIGURE 1.2: Our proposed experiment shows how images can first be labeled automatically (left). The CNN model is then trained with the obtained data to receive 2D predictions (center). These can be triangulated to evaluate the model's performance in 3D space (right).

This thesis aims to develop a marker-free method that allows for detecting smaller and thinner surgical objects, in our case, K-wires. The main focus of this work is on the data generation process. For this purpose, a pipeline is implemented with which it is possible to generate stereo images which automatically label the K-wires to be tracked. In the next step, a stereo neural network is developed to detect and predict objects on the images (cf. fig. 1.2). It is also not sufficient to detect the objects and frame them with a bounding box since we are interested in the trajectory of a K-wire. They must be recognized as a whole and represented as such to avoid correspondence matching. Our performance evaluation is handled by comparing the 3D detections with our spine phantom's 3D model.

Chapter 2

Ground Truth Generation

This chapter covers the entire pipeline for generating automated labeled images. First, a camera model is presented on which the calculations are based. Then, the experimental setup for automated data acquisition and labeling is explained.

2.1 Perspective Projection Model

This thesis requires a mathematical model to transform a point in the three-dimensional world into two-dimensional pixel coordinates. A simple but widely used model is the pinhole camera model. A pinhole camera consists of a camera body with a tiny hole through which light enters. The light that enters creates an image on the back of the camera body, which is upside down. A schematic illustration of the pinhole camera model can be found in the appendix (cf. fig. A.1). In the context of computer vision, a simplified or idealized model is used, which has an infinitely small pinhole and, therefore, perfect sharpness (Thormälen, 2021). Furthermore, we assume that the image is displayed on an imaginary image plane in front of the projection center so that the image is no longer upside down. To find the pixel coordinates (u, v) of a point P_w in a three-dimensional space, we require three coordinate systems interacting, which will be explained in the following subsections.

2.1.1 World Coordinate System

The world coordinate system corresponds to the coordinate system of the space in which a camera locates itself. The world coordinate system has an origin with coordinates $(0, 0, 0)$. This origin is arbitrary. Furthermore, the world coordinate system has an x, y, and z-axis. With the help of these two specifications, we can define any point in the world coordinate system by measuring the distance of the point P_w from the origin along the x, y, and z-axis. The coordinates of any point P_w thus correspond to the coordinates (X_w, Y_w, Z_w) in the world coordinate system (cf. fig. 2.1).

2.1.2 Camera Coordinate System

Assuming our camera is located in the space defined by our world coordinate system at an arbitrary point (X_w, Y_w, Z_w), this camera can have a translation and a rotation relative to the world coordinate system. A vector can describe a translation with three elements and a rotation by a matrix with nine elements. Thus, a point P_w with world coordinates (X_w, Y_w, Z_w) can be described by a point P_c in the camera coordinate system (X_c, Y_c, Z_c) using a rotation matrix $\mathbf{R}$ and a translation vector t. The rotation aligns the axes of the coordinate systems, while the translation aligns the origin of the two systems. The relationship between these two coordinates can thus be summarized as:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \mathbf{R} \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + t \tag{2.1}$$

where the coordinates follow a Euclidean geometry. For a projective geometry, the coordinates are homogenized. Homogeneous coordinates allow us to perform common operations such as translation, rotation, scaling and perspective projection to be implemented as matrix operation. Thus, we obtain the following system of equations:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.2}$$

which can be written as:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = [\mathbf{R}|t] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.3}$$

Where $[\mathbf{R}|t]$ is also called the extrinsic matrix.

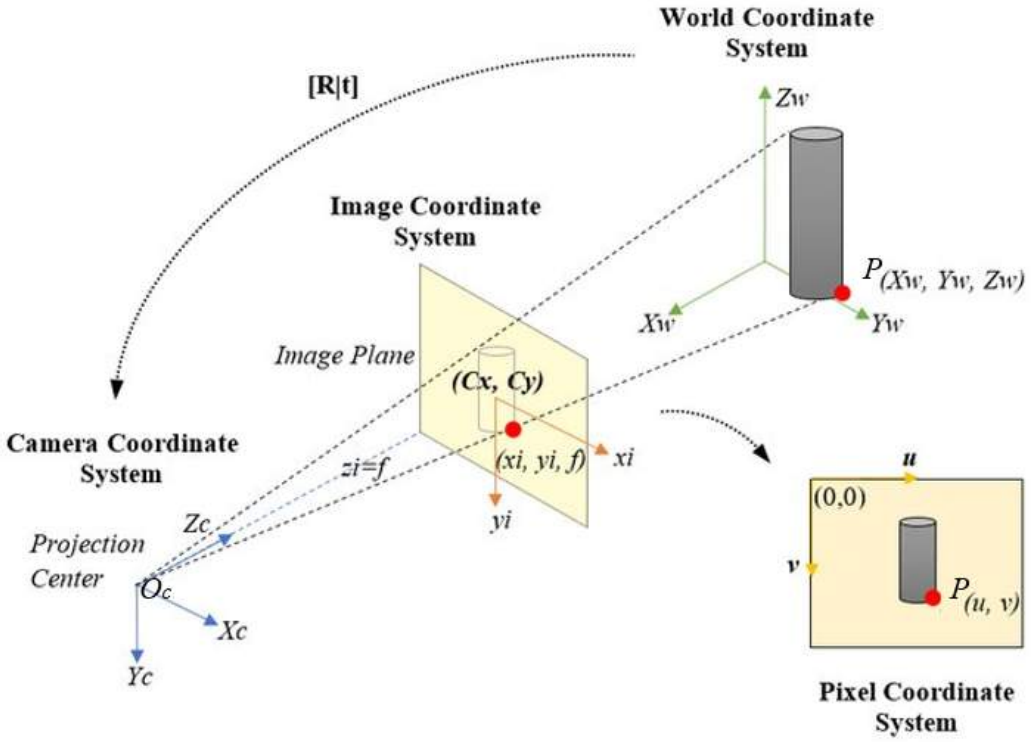


FIGURE 2.1: The world coordinate system, shown here in green, is our origin. The point P_w can be mapped through a rotation matrix $\mathbf{R}$ and a translation vector t in a camera coordinate system, here blue. The projection of the point P_w meets the image plane (orange) at the point, (x_i, y_i, f) which is the equivalent of (u, v) in the pixel coordinate system (yellow). (Image Ref. (Ortiz et al., 2017))

2.1.3 Image Coordinate System

The projection center, also called optical center, which represents the pinhole of the pinhole camera model, is defined by O_c (cf. fig. 2.1). The optical axis intersects the image plane at the principal point, described by c_x, c_y. It must be mentioned that the principal point c_x, c_y does not have to be in the center of the image coordinate system. The image plane is located at a distance of the focal length f from the optical center O_c along the optical axis. The projection of the three-dimensional point P_w in the world and camera coordinate system corresponds to the point (x, y) on the image plane. Due to the main theorem of similar triangles, we can state the relation:

$$x = f\,\frac{X_c}{Z_c},\; y = f\,\frac{Y_c}{Z_c} \tag{2.4}$$

Since the pixels of a camera do not have to be quadratic, we can address this by two different focal lengths f_x and f_y. The axes of the camera sensor may not be orthogonal, and therefore a skewness parameter γ is included. In summary, these definitions result in the camera matrix $\mathbf{K}$, also called the intrinsic matrix:

$$\mathbf{K} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

the image coordinates (u, v) are thus derived from the equation:

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \tag{2.6}$$

where:

$$u = \frac{u'}{w'},\; v = \frac{v'}{w'} \tag{2.7}$$

Thus, we can summarize that a three-dimensional point in the world coordinate system can be transformed into a point in the camera coordinate system using the extrinsic matrix (rotation and translation). This point can then be projected into a two-dimensional point on the image plane using the intrinsic matrix (internal camera parameters). In matrix notation, this leads to the following equation:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \mathbf{K}\,[\mathbf{R} \mid t]\,P \tag{2.8}$$

2.1.4 Calibration

The presented camera model described by equation 2.8 is only correct for perfectly manufactured lenses, which usually is not the case. Most cameras have a characteristic called lens distortion due to their design. The lens distortion issue can be divided into two categories: radial and tangential. Radial distortion is a result of the shape of the lens. Tangential distortion is a result of the construction process of the camera as a whole. The two distortion types are also called barrel or "fish eye" effect and pincushion effect. A schematic illustration of this can be found in the appendix (cf. fig. A.2).

In order to correct the effect of distortion and primarily inaccuracies during the manufacturing of the lenses, we can calibrate our camera to retrieve our intrinsic

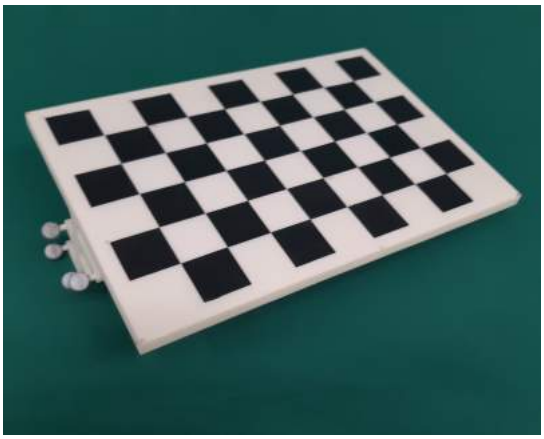


FIGURE 2.2: The physical checkerboard pattern with which the camera was calibrated also has a 3D marker which can be tracked by fusionTrack 500.

camera parameters, including the distortion parameters, with which the image is finally corrected (Bradski & Kaehler, 2008). The calibration process of the camera was modified in our work so that the points picked up with the pointer tip correspond to the checkerboard corners. This calibration method was done as follows:

1. fusionTrack 500 detects the 3D markers of the chessboard (cf. fig. 2.2) and the pointer (cf. fig. 2.4b).
2. With the pointer, we retrieve the corners of the chessboard pattern and store these 3D coordinates.
3. After all corners are scanned, the ZED Mini (cf. fig. 2.4a) starts and captures an image of the chessboard.
4. We use a pre-built algorithm to find the pixels of the chessboard corners of the captured image to get the 2D coordinates.
5. We retrieve the camera intrinsic matrix and the distortion coefficients of our camera.
6. By the Perspective-n-Point (PnP) pose computation, we compute the rotation and translation that minimizes the reprojection error from 3D-2D point correspondences.
7. The process is repeated from different angles, and the resulting rotation matrices and translation vectors are averaged.

The position of our pointer will then be corrected with the resulting rotation and translation to minimize the error during the data generation process explained in section 2.2.2.

2.2 Automated Image Labeling

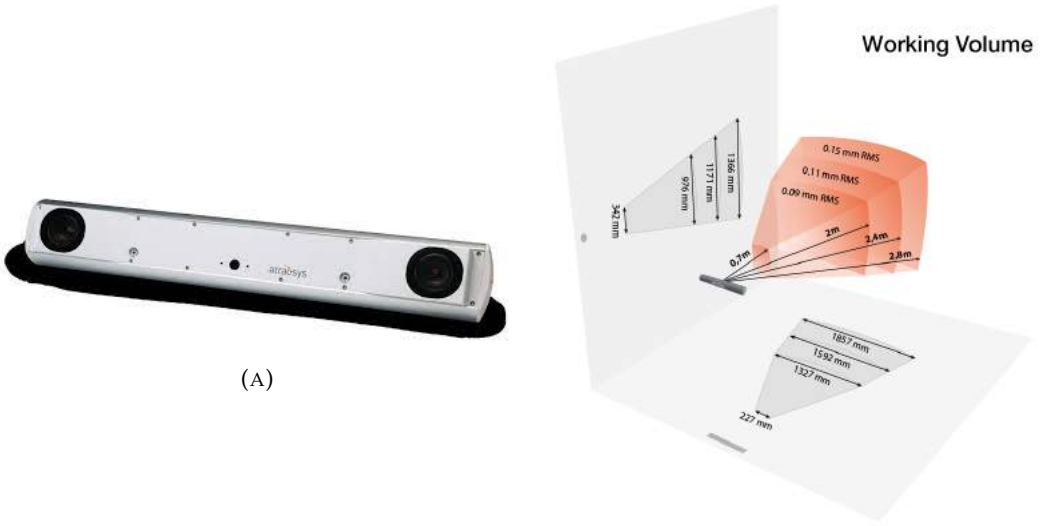
Training deep neural networks requires large amount of data. Obtaining large amounts of labeled medical image data is a crucial problem. Therefore, this thesis heavily focuses on developing a system to automatically generate labeled stereo image data. This section explains the components used in the experimental setup and shows the data acquisition process.

2.2.1 Components

For the experiment, four essential components are used. A device is needed to track objects as accurately as possible. Furthermore, a camera is required which can record stereo images. For markerless detection, a pointer is necessary to mark the desired objects. Finally, a phantom is needed in which the K-wires to be tracked are mounted.

fusionTrack 500

The tracking system used is the fusionTrack 500 by Atracsys (Atracsys SA, Switzerland) (cf. fig. 2.3a), which is a real-time optical pose tracking system specifically designed for detecting and tracking reflective spheres in real-time video streams with high accuracy and a measurement rate of 335 Hz (cf. fig. 2.3b). The fusionTrack consists of two infrared cameras that observe the reflective spheres (fiducials) and can determine their position and rotation using triangulation. The fusionTrack builds the origin of our world coordinate system and is fixed on a tripod.



(A)

(B)

FIGURE 2.3: (A) The tracking device used in our experiments was the fusionTrack500 manufactured by Atracsys. (B) The working volume shown indicates the accuracy of fusionTrack up to 2.8 m (Image Ref. (Atracsys, 2022)).)

ZED Mini

The images used to train the network were taken with the stereo camera ZED Mini from Stereolabs (Stereolabs Inc., San Francisco, CA, USA) with a resolution of 2560x720 (1280x720 per frame) pixels at 60 frames per second. The stereo camera captures RGB images and can also estimate depth using stereo depth sensing. This work does not consider depth information because K-wires are too narrow to be represented by depth estimation adequately. The ZED Mini is in a 3D printed case which has a marker on which four fiducials are placed. The markers can be tracked by fusionTrack to get the camera's location in real-time. The camera is not fixed and is moved manually to capture images during the recording process.

(A) (B) (C)

FIGURE 2.4: The objects tracked by fusionTrack: (A) ZED Mini stereo camera in the custom case with 3D marker, (B) 3D printed pointer with 3D marker, (C) Spine phantom with 6 K-wires. The silver spheres fixed to the camera and the pointer have a specific geometry assigned an ID for fusionTrack to distinguish them in real-time.

Pointer

The pointer is a 3D printed pointer that has the shape of a pen. It has a tip at one end and a marker at the other end, to which four fiducials are attached which can be localized by fusionTrack. The origin of the pointer coordinate system must be at the pointer tip for our method.

Spine Phantom

The test object is an artificial spine made by Synbone (SYNBONE AG, Zizers, Switzerland). The phantom contains the coccyx, the sacrum, and the lumbar vertebrae L1 to L5. In addition to bones, the phantom also contains an artificial spinal cord, including spinal nerves and artificial intervertebral discs. The K-wires to be detected are fixed in our spine phantom. The K-wires used in this experiment are steel wires with a diameter of 4 mm. They have a thread at the bottom end, which allows them to be screwed into or out of the spine phantom. On average, the K-wires were 40 mm long.

2.2.2 Experimental Setup

The world coordinate system for our experimental setup is drawn from fusionTrack, which is fixed on a tripod and does not move (cf. fig. 2.5 Letter A). The ZED Mini (cf. fig. 2.5 Letter B) and the fusionTrack are both connected to the same computer. The fusionTrack is directed to a green surgery cloth, our neutral background, which covers the surrounding area and the table-top. On top of the table-top is our spine phantom, our test object (cf. fig. 2.5 Letter D) in which the K-wires are attached, which need to be tracked (cf. fig. 2.5 Letter E).

Recording Process

1. When we start our application, we are shown which markers are visible for the fusionTrack, in our case, the ZED Mini, and the pointer.

2. To start the process in which we collect the points we want to track, we press a designated key.

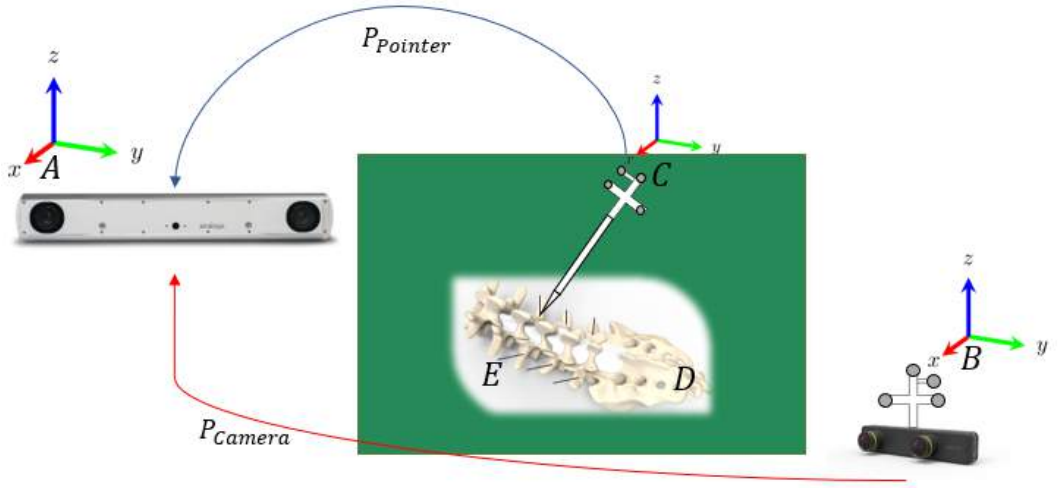


FIGURE 2.5: A schematic illustration of the automated data generation process. (A) Corresponds to our world coordinate system, (B) to our camera coordinate system, and (C) to the pointer coordinate system. (D) represents our spine phantom in which our K-wires (E) are fixed.

3. With the pointer, we can locate the points we want to track. We first hold the tip of the pointer to the start point and then to the endpoint of a K-wire we want to track. The start and end points of a K-wire are saved by pressing a key designated for this purpose.The 3D coordinates of the pointer w.r.t. our world coordinate system, are now stored in a C++ custom application. This process is repeated until all positions of the K-wires have been selected.

4. As soon as all positions of the K-wires are stored, the pointer is no longer used. This is especially important since we do not want artifacts in the recorded frames, such as the tip of the pointer, which would introduce a bias into the machine learning part.

5. The recording of the stereo camera can be started by pressing another button.

   (a) After all positions have been picked up w.r.t. our world coordinate system, we transform them into the 3D coordinates of the moving camera. For this, we express the positions of the pointer relative to the camera coordinate system using the equation:

   $$P_{Camera} = [\mathbf{R}|t]\, P_{Pointer} \tag{2.9}$$

   Where $[\mathbf{R}|t]$ represents the rotation and translation from the world to the camera coordinate system. We use the extrinsic parameters of the camera (cf. fig. 2.5 P_{Camera}) and the tracked points (cf. fig. 2.5 $P_{Pointer}$), which gives us a new 3-dimensional translation vector. This corresponds to the positions of the pointer expressed in the camera coordinate system.

   (b) The transformed positions are now transformed from the camera coordinate system to our image plane.

(c) The camera is not fixed and is moved by the operator by hand as soon as the recording is started (cf. fig. 2.6). Approximately three frames per second are captured, one for the left and one for the right camera. The number of captured images per second is limited by the speed of our algorithm, while it also does not make sense to capture too many images per second as the positions change only minimally during the camera movement, and thus unnecessarily, many images would be generated which have the same characteristics. At the same time, two text files are generated containing the positions of the K-wires in normalized coordinates, one for the left and one for the right camera. To match the image and position pairs, they are named with a timestamp as the filename. This way, each image-file-pair can be assigned to a corresponding position-file-pair. The position file contains the information of the saved positions which have been picked up with the pointer.

(d) The capture and transformation of these coordinates are then repeated for each incoming frame.

6. This process can be supervised on a screen where the recordings are streamed, and the projected start and end points of the K-wires are visualized after projecting them onto the image plane.

FIGURE 2.6: A picture taken during the data aquisition.

Chapter 3

K-wire detection

This chapter deals with the prediction part of the K-wires. It contains the K-wire detection and the triangulation of the predictions. The pipeline's inputs are the RGB stereo images taken in the previous chapter and the corresponding ground truth labels. These image pairs are passed to a stereo neural network, forming prediction pairs for each K-wire. The resulting predictions are then triangulated so that each start and end point of the K-wires can be assigned a coordinate in the 3-dimensional space. These coordinates can then be registered to 3-dimensional ground truth to evaluate the prediction performance.

3.1 Stereo Neural Network

The model used in this work corresponds to a stereo neural network. It does not use bounding boxes (Li et al., 2019). An approach using bounding boxes would not generate any added value for this work, since bounding boxes cannot represent the trajectories of the individual K-wires. The model represents the wires to be predicted by a center, a normalized 2D direction vector, and a length represented by a YOLO-like output (Redmon & Farhadi, 2018).

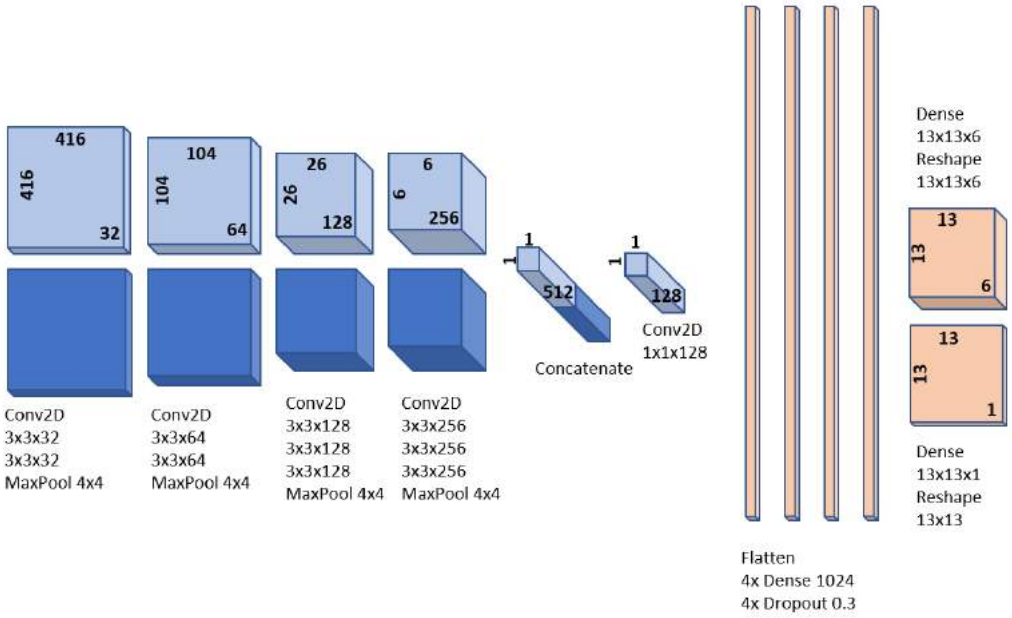


FIGURE 3.1: Our proposed network architecture with two identical branches for the left and right input images.

3.1.1 Model Architecture

The stereo model consists of two identical branches (cf. fig. 3.1). Two input layers initialize it with the shape of each image size (the images are resized to 416x416) for the left and right image and three channels for RGB input. The input layers are followed by a batch normalization layer and two convolutional layers with 32 filters, a kernel size of 3x3, and a ReLu activation function. A max-pooling layer then processes these feature maps with a pool size of 4x4. The next convolutional block is identical to the one described, except it doubles the number of filters in the convolutional layers from 32 to 64. The subsequent convolutional block is initialized by a batch normalization layer followed by three consecutive convolutional layers with 128 filters, a 3x3 kernel, and a ReLu activation function which is then reduced by a max-pooling layer with a pool size of 4x4. The next convolutional block is identical to the previous, except it doubles the number of filters in the convolutional layers from 128 to 256. The last convolutional block is initialized by a batch normalization layer followed by three consecutive convolutional layers with 256 filters, a 3x3 kernel, and a ReLu activation function which is then reduced by a max-pooling layer with a pool size of 4x4. The convolutional part of the model is built so that both branches share their weights and enable the model to generate consistent feature maps for the left and right input images. Then the resulting feature maps from both branches are concatenated and compressed by a convolutional layer with 128 filters and a 1x1 kernel with a subsequent flatten layer. The one-dimensional part of the model consists of 4 consecutive dense layers with 1024 units each, a leaky ReLu activation function with a negative slope coefficient of 0.1 and a dropout layer with a rate of 0.3.

The output of our model consists of two reshaped tensors (cf. section 3.1.2). The first tensor has a shape of 13x13x1. It is responsible for the detection and is regressed with a sigmoid function. The second tensor has a shape of 13x13x6 and has a linear activation function. It is responsible for the parameters that describe the properties of the K-wires on the right and left input images.

3.1.2 K-wire Encoding

The output layer consists of 2 tensors which can be combined into a 13x13x7 tensor. With this shape, all K-wire detections in both images shall be reconstructed. The output tensor segments each image into a 13x13 grid of 169 cells (cf. fig. 3.2). The grid size and model architecture are inspired by von Atzigen et al., 2022. Each cell of this grid is responsible for detection y_i of a K-wire, where the center of a K-wire (dx_i, dy_i) must lie within the boundaries of a cell at (cx_i, cx_i). Within these cells, there are seven parameters to be regressed, which can be divided into three groups:

$$y_i = \left[\underbrace{tp_i}_{\text{presence}}, \underbrace{tx_i, ty_i, tl_i, tdirx_i, tdiry_i}_{\text{position, length, direction}}, \underbrace{tst_i}_{\text{stereo correction}} \right] \tag{3.1}$$

The first parameter, tp_i, is binary. It indicates whether a K-wire is detected in a specific cell at (cx_i, cy_i) or not. The threshold for a positive outcome is 0.5. The next two parameters tx_i, ty_i define the position of the center of a K-wire (cf. fig. 3.2b). Each cell in our 13x13 grid has a unit width and height, where the upper left corner is represented by the values (cx_i, cy_i). In this scenario, we will not regress the position of our wires in global pixel coordinates, but relative to the cell (cx_i, cy_i) where a detection takes place. The position of the center of the K-wire is then obtained by

$$dx_i = \sigma(tx_i) + cx_i \tag{3.2}$$

$$dy_i = \sigma(ty_i) + cy_i \tag{3.3}$$

where dx_i and dy_i define the exact location of a K-wire center within a grid cell and σ denote the sigmoid function. The sigmoid function is used here to keep the regressed values in a range of [0,1] in order to assure that the predicted center remains within the predicted cell. The parameter tl_i is used to calculate the length of a K-wire. In order to include prior knowledge about the length of the K-wire, an anchor value αl is applied. This value is derived from the average value of all training data. For a detailed explanation of the anchor concept, see (Redmon & Farhadi, 2016). This gives us the equation

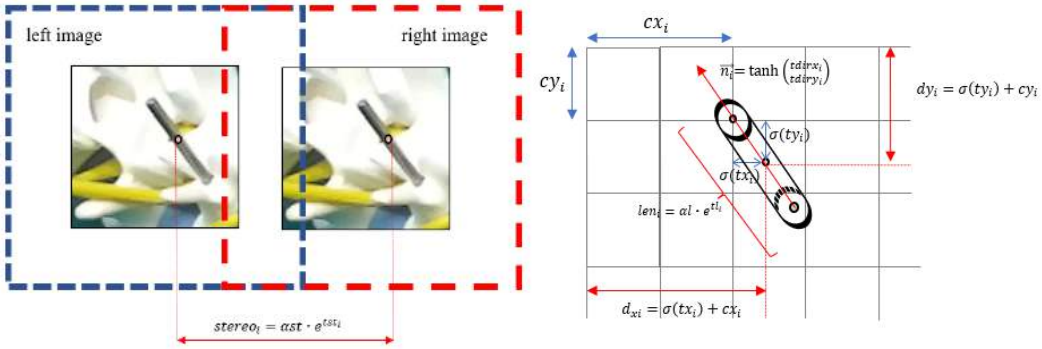
$$len_i = \alpha l \cdot e^{tl_i} \tag{3.4}$$

where len describes the length of a K-wire in pixel space. An anchor αst equally handles the stereo correction tst_i. This lets us state the equation

$$stereo_i = \alpha st \cdot e^{tst_i} \tag{3.5}$$

where $stereo$ denotes the translational disparity from the K-wire center in the left frame compared to the right frame. The parameters are only regressed for the left frame and then transformed by the horizontal offset tst_i to get the parameters of the right frame. The direction $tdirx_i, tdiry_i$ in which a detected K-wire points is indicated by a normalized direction vector encoded by the hyperbolic tangent resulting with the statement:

$$\overrightarrow{n_i} = tanh\begin{pmatrix} tdirx_i \\ tdiry_i \end{pmatrix} \tag{3.6}$$



(A) (B)

FIGURE 3.2: (A) The blue box shows a section of the left image, and the red box shows a section of the right image. The red dots show the center of a K-wire. The distance between the centers of the K-wires defines the stereo disparity parameter $stereo_i$. (B) The values dx_i, dy_i, len_i and $\overrightarrow{n_i}$ and encode the K-wire. All values are defined relative to the top left corner (cx_i, cx_i) of the respective cell. The sigmoid function σ normalizes the activations into the range [0,1].

3.1.3 Training

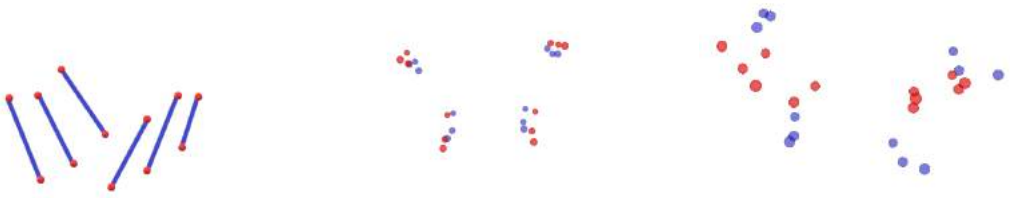
Our network was implemented in TensorFlow within the Keras API. All images were scaled to a resolution of 416x416 pixels and normalized. Weights were randomly initialized and trained from scratch using 5-fold cross-validation with 150 epochs each and a batch size of 64. A binary cross-entropy loss was used for the detection tensor, while a mean square error loss was used for the rest of the parameter estimation, optimized by an Adam optimizer with a learning rate of 0.001. For better data generalization, image augmentation was performed by flipping the image pairs with one-third probability horizontally and one-third probability vertically. The training took approximately 14 hours for each cross-validation fold and was performed on an NVIDIA RTX A6000.

3.1.4 Experiments

Prior to the implementation of the stereo model, models based on mono view were first implemented. Subsequently, the mono view models were extended to the stereo view models to take the 3-dimensionality into account. Instead of the presented encoding, alternative encoding were applied, such as the representation by an angle of the K-wires instead of a normalized direction vector. Furthermore, the K-wire was not detected as a whole, but the start and end points of the K-wires were detected independently of each other. Different loss and activation functions were also tried to account for the circularity of an angle, such as a Taylor series expansion of the cosine function and a hyperbolic tangent activation function. However, the experiments did not lead to better results.

3.2 Evaluation

The automated image labeling results in stereo image pairs with corresponding normalized 2D coordinates. These can be scaled back to the pixel space by multiplying the image width and height. Highlighting these coordinates on the image shows us the start and end points of the K-wires on the images (cf. fig. 4.2). In this setup, the performance of our system cannot be evaluated quantitatively unless we compute the deviation in pixel space. However, this would give little insight into the accuracy as the scale of an error is unknown since a deviation of one pixel is far less in a close-up image compared to an image taken from a larger distance. To address this problem, we triangulate the coordinates of the start and end points of the K-wires for the stereo image data. Using the built-in OpenCV function `triangulatePoints()`, the corresponding 2D points for the left and right camera, and our intrinsic parameters, we obtain the 3D coordinates for the start and end points of the K-wires. As the "real" ground truth of our spine phantom, the start and end points of the K-wires are captured in the world coordinate system with fusionTrack 500 multiple times and averaged. These 3D coordinates are then built into a 3D model in a Computer Assisted Surgery Planning Application (CASPA is a planning software designed and implemented by the ROCS research group) (cf. fig. 3.3a). Subsequently, the 3D coordinates from our triangulated image labeling data and our triangulated predictions are merged into a 3D model in CASPA (cf. fig. 3.3b, 3.3c). To compare the models, they are now registered using the Iterative Closest Point Algorithm (ICP) (Arun et al., 1987).



(A) (B) (C)

FIGURE 3.3: (A) Visualization of the start and endpoints in our ground truth 3D model. (B) Triangulated labels (red) compared to our ground truth 3D Model (blue) from an axial perspective. (C) Triangulated predictions (red) compared to our ground truth 3D Model (blue) from an axial perspective. In this example, it is visible that the triangulated predictions show a higher discrepancy to the ground truth 3D model than the triangulated labels.

The coordinate systems of the ground truth and the coordinate system of the predictions are now aligned. We have matched the point clouds as well as possible in the least squares sense by ICP. Now we calculate the average distance of the individual points and the average angle deviation of the corresponding K-wires.

The 3D angular error deg was calculated using the directional vector of our ground truth model $\vec{GT}$ and the directional vector $\vec{L}$ obtained by our triangulated predictions.

$$deg = \arccos\left(\frac{\vec{GT}\cdot\vec{L}}{\left\|\vec{GT}\right\|\left\|\vec{L}\right\|}\right)\cdot\frac{180}{\pi} \tag{3.7}$$

The distance $dist$ from a start and endpoint of our ground truth model $gt_{xi}, gt_{yi}, gt_{zi}$ to a start or endpoint of our triangulated predictions l_{xi}, l_{yi}, l_{zi}, which represents the translational error, was calculated by:

$$dist_i = \sqrt{(gt_{xi}-l_{xi})^2+(gt_{yi}-l_{yi})^2+(l_{zi}-l_{zi})^2)} \tag{3.8}$$

where $i = 1, 2$ and denotes either a start or an endpoint of the K-wire.

Chapter 4

Results

This chapter is divided into three parts: first, the performance of the ground truth generation is evaluated, then the predictions of the K-wire detection are evaluated, and finally we evaluate detections based on CGI.

4.1 Ground Truth Generation

Figure 4.2 shows some examples that visualize our ground truth labels and predictions. On the left-hand side are the images of the left camera, and on the right-hand side are the image of the right camera. The start and end points of the K-wires of the ground truth labels are visualized with red points connected by a red line. The predictions of our network are visualized in blue. Figure 4.2 shows that the ground truth labels qualitatively represent the K-wires very well from different angles. However, it is also visible that the ground truth label representation on the right images is mostly worse. This deviation could be related to poor camera calibration, for example (cf. fig. 4.2e).

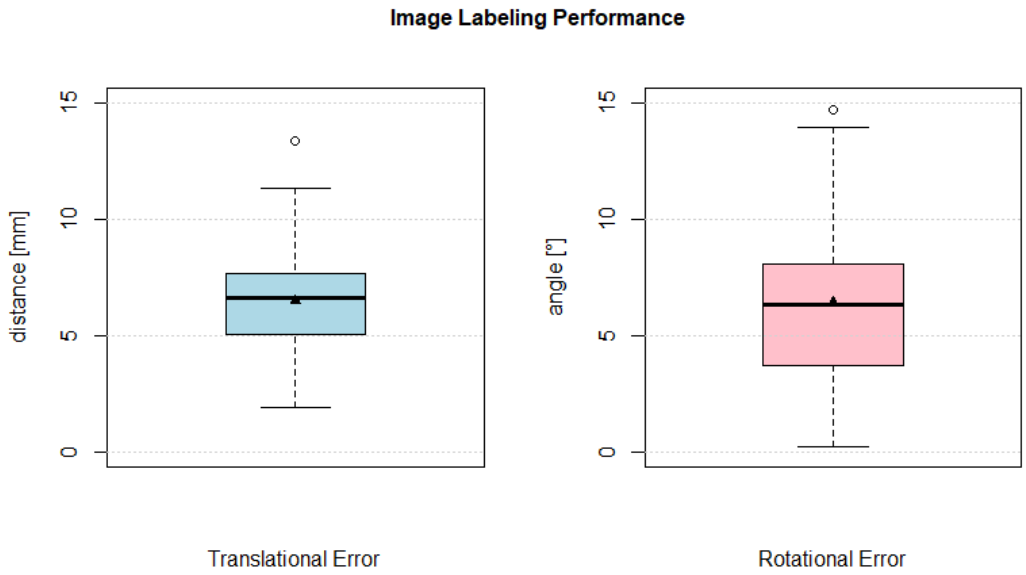


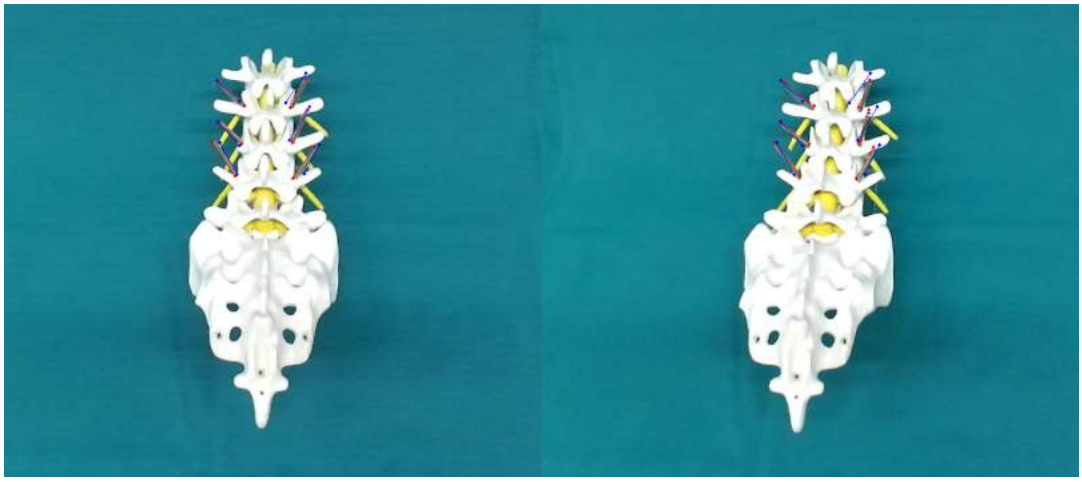
FIGURE 4.1: The translational and rotational error of the automated image labeling system compared to a ground truth 3D model. The solid line in the box represents the median, whereas the triangle represents the mean.

Since our evaluation procedure, particularly the ICP registration, cannot be automated (as we do not have the required camera poses), the entire process is calculated manually. Five random samples of the labeled images were evaluated, each with 12 3D coordinate points or six K-wires (n = 60). The average deviation of the distance between a K-wire start or end point was 6.60 mm ± 2.3 mm. The median translational deviation was 6.50 mm. The average angle deviation between 2 associated K-wires was 6.35° ± 3.69°. The median rotational deviation was 6.40° (cf. fig. 4.1).

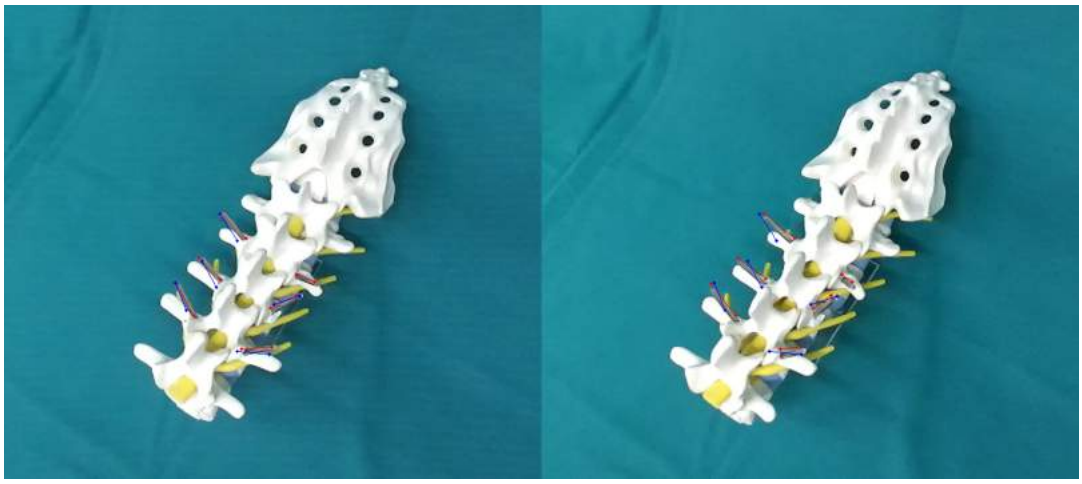
4.2 K-wire Detection

The labels represented as red dots connected by a red line and the predictions, represented as blue dots connected by a line in figure 4.2, are of different qualities. Figure 4.2a shows that all six K-wires are detected. The accuracy is better on the left image than on the right image. This observation can be seen by the fact that on the left image, the blue lines representing the predictions overlap more with the red lines representing our labels than on the right image. Furthermore, it is visible that in both pictures, the position and angle of a K-wire (top right) deviates particularly. In figure 4.2b, it can be seen that the angles of the K-wires are already not well represented on the left frame, which leads to an even more significant error on the right frame. Furthermore, it can be observed here that out of six, only five K-wires were detected. Figure 4.2c and 4.2d show that 2-3 K-wires are tracked accurately, while the remaining ones are only tracked poorly. In figure 4.2c, the prediction on the left frame is relatively good except for one K-wire, while on the right frame, the predictions look worse, which could also be due to inferior ground truth labels in this example. The images shown in figure 4.2 were randomly selected and then manually filtered so that the detections have different characteristics.

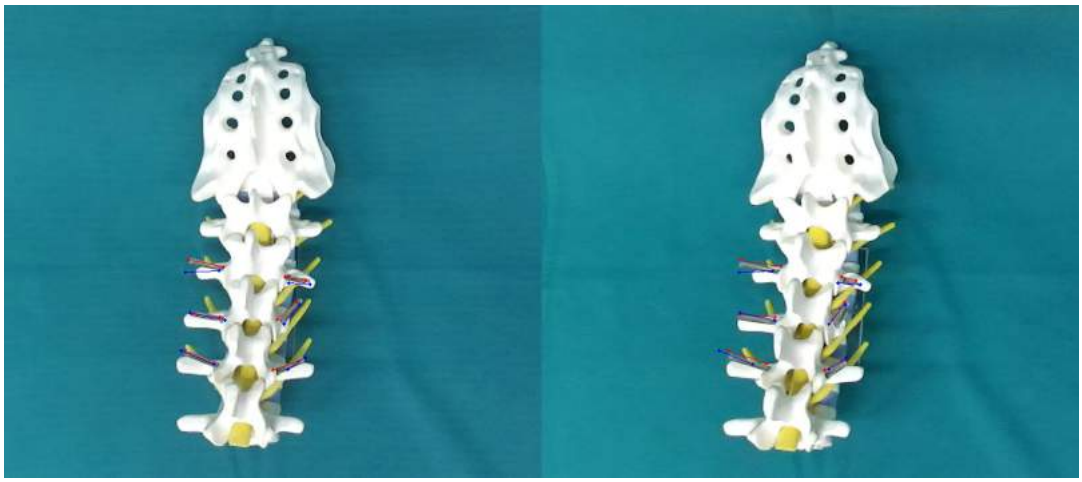
After the triangulation of the predictions, we noticed that the 3D representation of the K-wires is too flat in most samples (cf. fig. 3.3c). The assumption here is that the stereo parameter encoding is insufficient. With the implementation of a stereo parameter for the center of a K-wire, it is assumed that the start and end points of the K-wire are equidistant from the camera. Consequently, this leads to the same disparity for start and end points and the same depth. This assumption is incorrect and leads to the phenomenon mentioned above, expressed by the flat pressed triangulation. Thus, an evaluation of the triangulated model predictions does not adequately describe performance of our model. This problem is therefore addressed and partially solved in the next section (cf. 4.3).



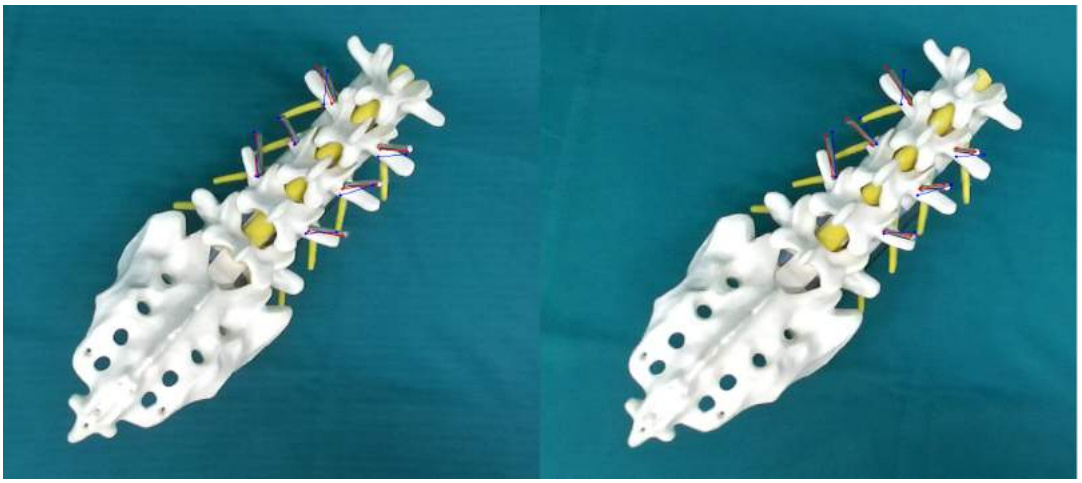
(A)



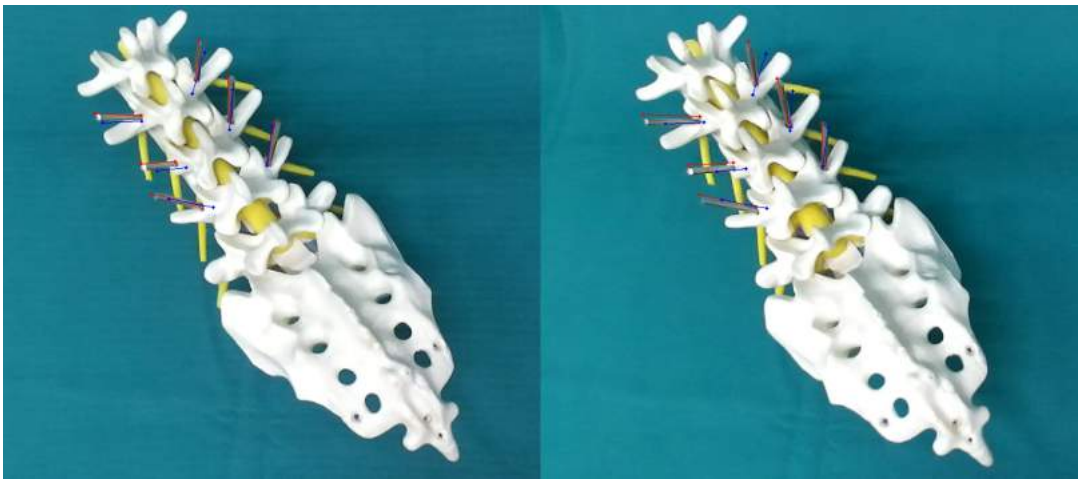
(B)



(C)



(D)



(E)

FIGURE 4.2: Left and right image of our automated labeling and prediction system. The red lines mark the ground truth K-wires tracked by our system. The blue lines mark the predicted detections of our stereo neural network.

4.3 CGI-based K-wire Detection

Due to the limitations that arose during the evaluation of K-wire detection, a parallel experiment was conducted in this work using computer-generated images (CGI) made with Blender. The ground truth labels have perfect accuracy. Figure 4.3 shows random CGI with their ground truth labels in red and the predictions in blue. It can be seen that the model very well represents most K-wires. In figure 4.3c it can be seen how K-wires are well detected, but the direction in which they are represented is not correct for some of them. From a visual point of view, most of the predictions of the stereo neural network on the CGI are very good. This assumption is also reflected in the evaluated performance.

Due to the constant working distance from which the images were rendered and a given field of view and sensor width of the CGI's, the error in pixel space can be estimated as an error in mm. For this, the ratio between pixel and millimeter is calculated by applying the law of sines and solving the following equation (cf. fig.4.4) for ratio:

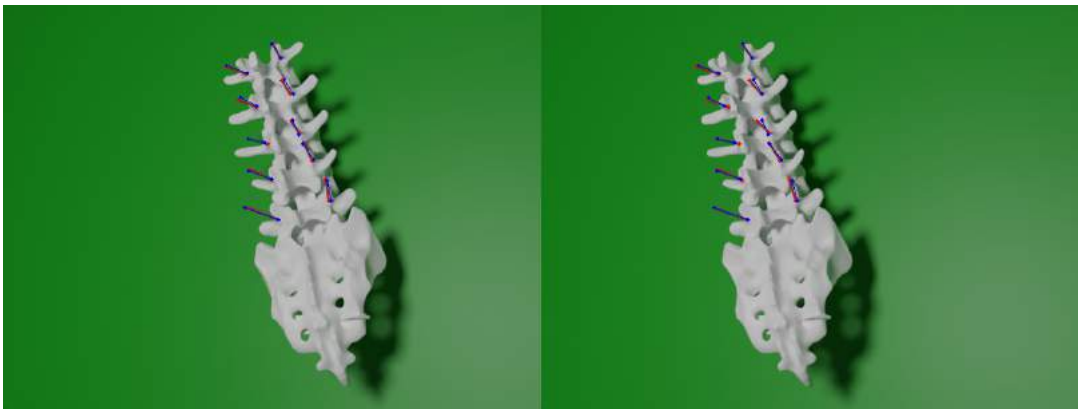
$$ratio = \frac{\frac{sensorwidth\ [px]}{2}}{\tan\left(\frac{FOV}{2}\right) \cdot f[mm]} \tag{4.1}$$

which gives us the ratio

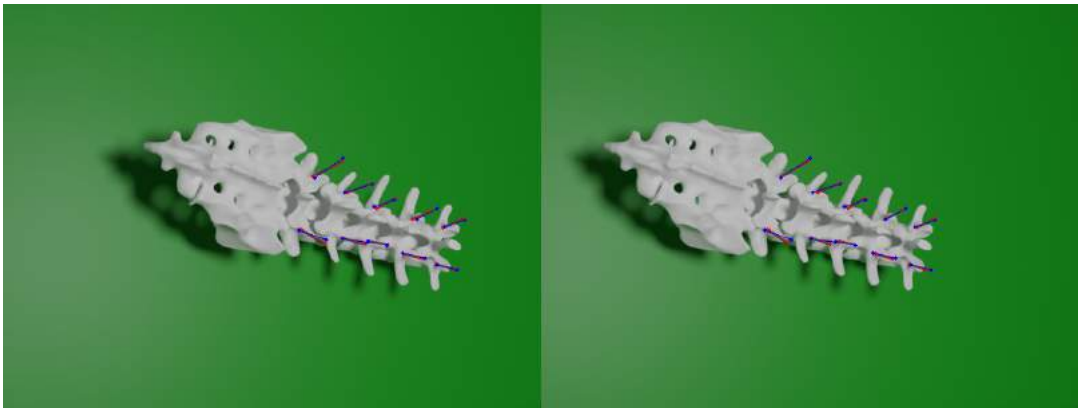
$$17.78px = 1mm \equiv 1px = 0.05625mm \tag{4.2}$$

We can now use this ratio to obtain an approximate estimate in mm. For this, we compute the euclidean distance (cf. eq. 3.8 without a z-coordinate) in pixel between the label and the prediction of a start or end point of a K-wire and scale it by the obtained ratio in equation 4.2.

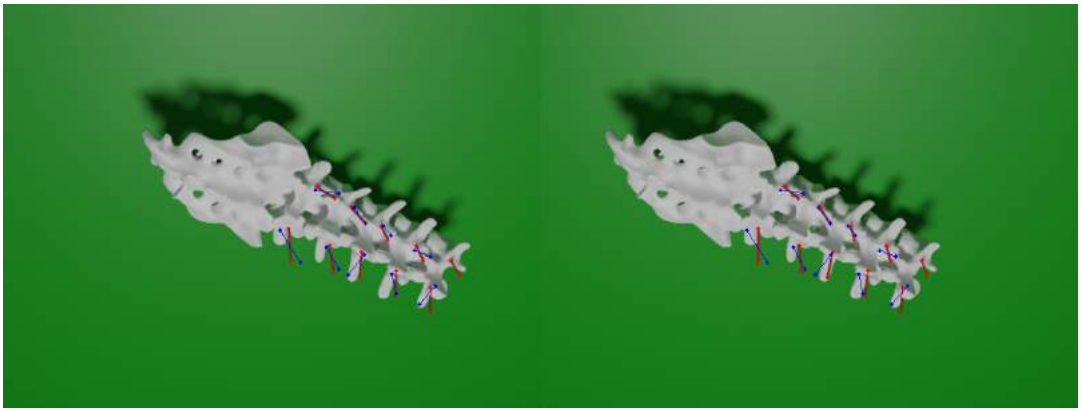
For the evaluation, 382 stereo predictions from the test set are evaluated, each containing 10 K-wires (n = 3820). The average distance of a K-wire start point was 6.19 mm ± 2.06 mm. The median distance was 6.29 mm. The average distance of a K-wire end point was 6.11 mm ± 1.33 mm. The median distance was 6.28 mm (cf. fig. 4.5). Averaged over start and end values, we get an error with a distance of 6.15 mm ± 1.73 mm. The results of the predictions for the left and right images are equivalent.



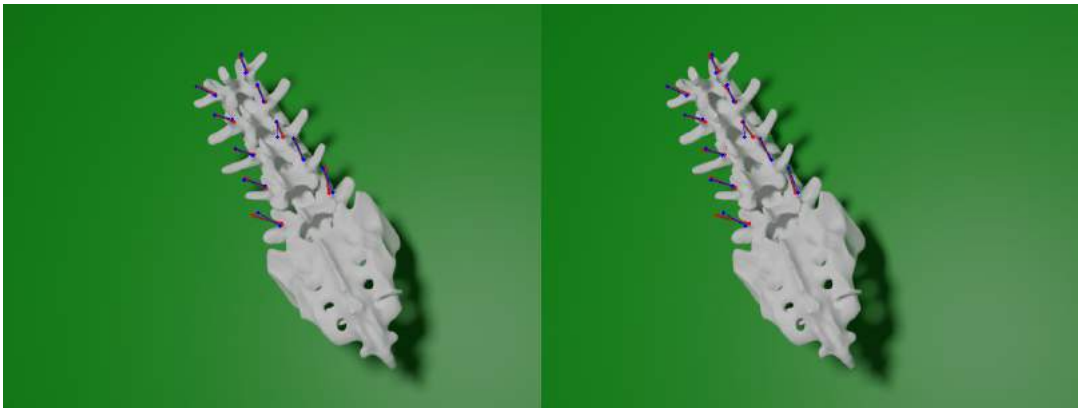
(A)



(B)



(C)



(D)

FIGURE 4.3: Left and right CGI. The red lines mark the ground truth K-wires. The blue lines mark the predicted detections of our stereo neural network.

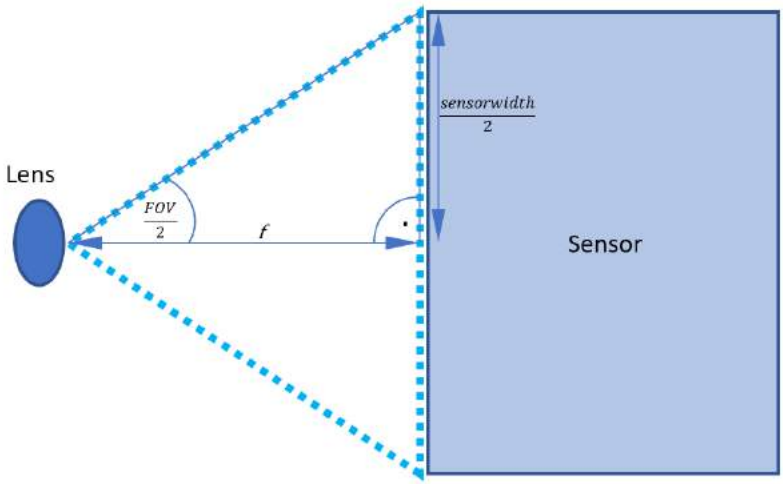


FIGURE 4.4: The ratio can be determined using the law of sines. The three angles required for this correspond to half of the FOV, a right angle, and the resulting third angle to obtain the angular sum of 180°. The focal length f corresponds to 50 mm.

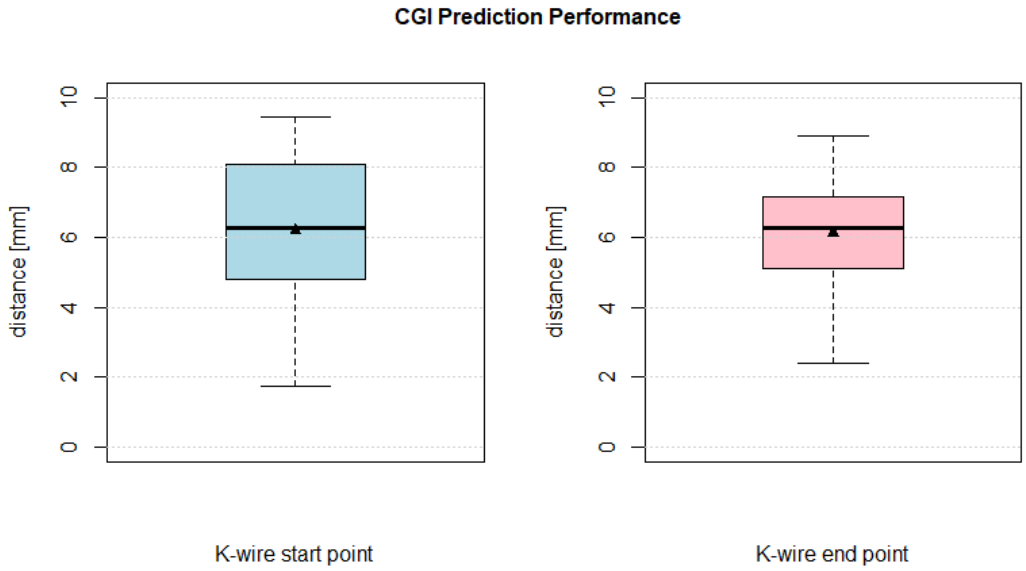


FIGURE 4.5: The distance of the predictions compared to our labels for a start and endpoint of a K-wire. The solid line in the box represents the median, whereas the triangle represents the mean.

Chapter 5

Discussion and Outlook

The results have shown that it is possible to label data automatically with acceptable accuracy. Our system makes it possible to use data-hungry methods such as deep learning, which would otherwise not be possible since the data or the labels are often unavailable in medical applications for machine learning. Our work makes it possible to acquire several thousand images within a few hours, in which various positions can be tracked. Nevertheless, tracking is limited by restrictions such as the visibility of fiducials and the working volume of fusionTrack 500.

Due to the good results in detecting K-wires using our stereo neural network on the computer generated images, it can be said that our method meets the requirements to contribute as a marker-free method. In work preceding this thesis, pedicle screws have been predicted with a marker-less approach with a positional error of 5.43 mm (von Atzigen et al., 2022). This performance is comparable to the positional error of our automated labeling method, which achieved a positional error of 6.6 mm. Although we could not evaluate the K-wire detection model on the recorded images because of the faulty stereo encoding, we have shown that our model architecture works for CGI. With an average distance error of 6.15 mm, these values are in a good range for markerless detection. However, marker-based methods have shown better results, with an average positional error of 2.3 mm (Kriechling et al., 2020). Regardless, we believe markerless methods are the future of computer-assisted surgical navigation due to their various advantages described in the introduction.

During the development, different approaches were evaluated to achieve better detection results on the recorded images. Different loss functions were applied to account for an angle's circularity to represent the angles better. On the other hand, we tried to encode the angle as a radian instead of the normal vector encoding shown here. Furthermore, we tried to encode the K-wire not by center length and angle, but by the x and y coordinates of the start and end point. We also tried to work with different model architectures using smaller and larger models and fine-tune pre-trained models such as Res-Net (He et al., 2016). We also tried to use thicker K-wires and to consider different camera perspectives. However, all efforts to significantly improve the detection of the K-wires did not result in a better performance. In retrospect, it would have been essential to encode the K-wires using two stereo parameters to achieve meaningful results in triangulation.

A further consideration is also that detecting an object using a bounding box is easier than rendering lines, where even minor angle deviations lead to high discrepancies between the label and the original. However, in the paper of von Atzigen et al., 2022, two additional parameters are needed to represent a bounding box, which would argue against this statement. It is unclear whether the number of parameters to be regressed significantly changes the difficulty of solving the problem. We also

assume that it is more challenging to detect delicate texture-free objects like K-wires than larger surgical tools like drills.

Nevertheless, our results provide new insight into the detection of narrow objects. We could train networks with abundant data thanks to our automated ground truth generation. However, it became apparent that data quality is more important than quantity, especially in our case. Poorly labeled data, which came from systems which were not calibrated, achieved poor results in both ground truth generation and predictions. Therefore, before training deep learning models, great attention should be paid to the data quality.

Although computer-assisted surgical navigation could positively impact the quality of treatment in orthopedic surgeries, these systems are only used in 5% of orthopedic surgeries (Joskowicz & Hazan, 2016). This lack of acceptance could also be because the data needed to train machine learning methods are often unavailable or have to be laboriously labeled by hand. Our approach shows that it is possible to address these problems and thus contribute positively to advancing this research area of computer-assisted surgical navigation. Our image labeling system is not limited to tracking K-wires alone. Theoretically, it can also track other static objects in the surgical field or other key points like anatomical landmarks. Our system is generally not limited to surgical/medical applications, as it can track any unmoved object on a moving camera.

Future research dealing with markerless detection of K-wires should consider alternative encoding methods, which account for depth in an image by applying two or more stereo parameters. Also, it was not possible to consider alternative augmentation methods like contrast or brightness changes of the images to simulate changing light conditions in a operating room caused by surgeons' headlights or the bright spotlights in the operating room. Also, our spine phantom was very simple and did not cover the actual conditions in the operating room, where K-wires can have different lengths and diameters or other characteristics. The K-wires in our phantom were always clean and not covered by blood or other human tissue, as possible in surgical procedures. Whether this could positively or negatively influence the detection should further be investigated.

Chapter 6

Conclusion

To conclude, it can be said that the goals of this work were fulfilled. It could be shown that with the help of our method, large amounts of labeled data can be generated in a brief time. This automated data generation system can advance computer-assisted surgical navigation, as it can partly or entirely substitute the data generation aspect in various medical image analysis applications.

Although we could not evaluate the detection of K-wires with the recorded data, we have shown with the help of computer-generated images that our stereo neural network performs well in marker-free detection. The results obtained in this work should pave the way for future research in marker-free detection, and thus contribute positively to further advances in computer-assisted surgical navigation.

Appendix A

Schematic Illustrations

Pinhole Camera Model

The pinhole camera model has a box-shaped body that allows light to enter through a small opening. The object is thereby projected upside down onto the back of the body. This process is similar to the principle of the Camera obscura. In the context of computer vision, the image is projected through an infinitesimally small hole onto an imaginary image plane.

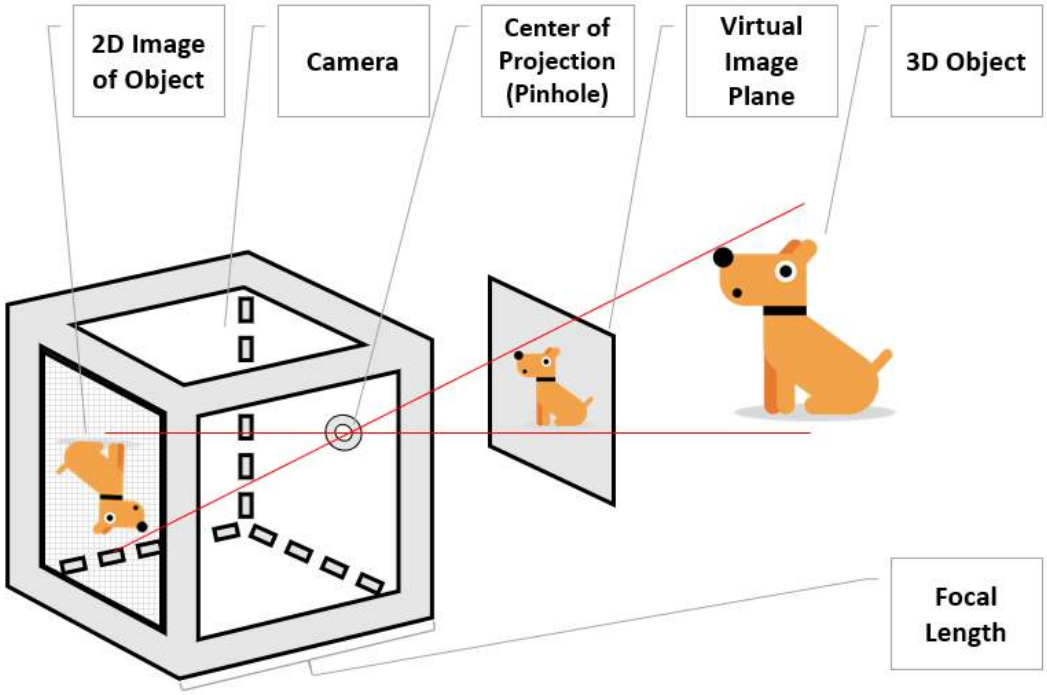


FIGURE A.1: A schematic illustration of the pinhole camera model

Distortion

We have a pattern on the left with an orthogonal chessboard pattern, a radial distortion in the center, and a tangential distortion on the right.

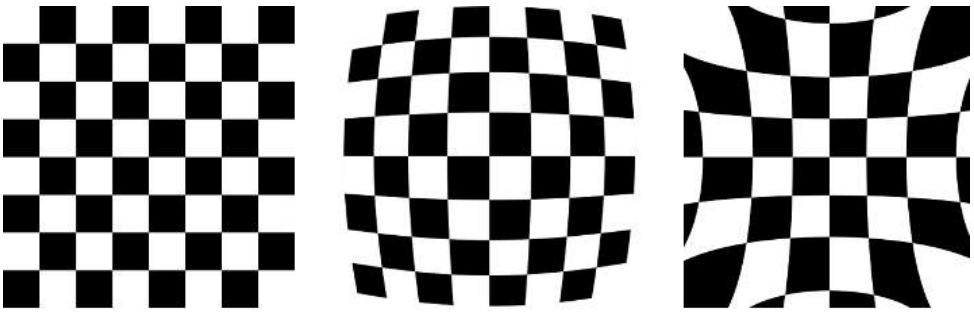


FIGURE A.2: Examples for two different distortion effects. (Image Ref. (Opencv, 2022)).

Bibliography

University Hospital, B. (2022). *Future-oriented surgical training*. Retrieved July 14, 2022, from https://www.balgrist.ch/fileadmin/user_upload/Aktuelles/Aktuelles/2022/220517_MM_EN_Proficiency_Innosuisse_final.pdf

Koskinen, J., Torkamani-Azar, M., Hussein, A., Huotarinen, A., & Bednarik, R. (2022). Automated tool detection with deep learning for monitoring kinematics and eye-hand coordination in microsurgery. *Computers in Biology and Medicine*, *141*, 105121. https://doi.org/https://doi.org/10.1016/j.compbiomed.2021.105121

von Atzigen, M., Liebmann, F., Hoch, A., Spirig, J. M., Farshad, M., Snedeker, J., & Fürnstahl, P. (2022). Marker-free surgical navigation of rod bending using a stereo neural network and augmented reality in spinal fusion. *Medical Image Analysis*, *77*. https://doi.org/https://doi.org/10.1016/j.media.2022.102365

Liebmann, F., Roner, S., von Atzigen, M., Scaramuzza, D., Sutter, R., Farshad, J. S. M., & Fürnstahl, P. (2019). Pedicle screw navigation using surface digitization on the microsoft hololens. *International Journal of Computer Assisted Radiology and Surgery*, *14*, 1157–1165. https://doi.org/https://doi.org/10.1007/s11548-019-01973-7

Hofstetter, S., Slomczykowski, & Nolte. (1999). Fluoroscopy as an imaging means for computer-assisted surgical navigation. *Computer Aided Surgery*, *4:2*, 65–76. https://doi.org/https://doi.org/10.3109/10929089909148161

Siston, R. A., Giori, N. J., Goodman, S. B., & Delp, S. L. (2007). Surgical navigation for total knee arthroplasty: A perspective. *Journal of Biomechanics*, *40*(4), 728–735. https://doi.org/https://doi.org/10.1016/j.jbiomech.2007.01.006

Balgrist, U. (2022). *Swiss, surgical workflow in spine surgery*. Retrieved July 6, 2022, from https://rocs.balgrist.ch/forschung/forschungsprojekte/swiss-192/

Kang, D.-H., Jung, D.-W., Kim, Y.-H., Kim, T.-G., Lee, J., & Chung, K. J. (2015). Kirschner wire fixation for the treatment of comminuted zygomatic fractures. *Archives of craniofacial surgery*, *16*(3), 119–124. https://doi.org/https://doi.org/10.7181/acfs.2015.16.3.119

Kriechling, P., Roner, S., Liebmann, F., Casari, F., Fürnstahl, P., & Wieser, K. (2020). Augmented reality for base plate component placement in reverse total shoulder arthroplasty: A feasibility study - archives of orthopaedic and trauma surgery. https://link.springer.com/article/10.1007/s00402-020-03542-z

Nasser, Yadla, Maltenfort, Harrop, Anderson, Vaccaro, Sharan, & Ratliff. (2010). Complications in spine surgery. *Journal of neurosurgery. Spine, 13(2)*, 144–157. https://doi.org/10.3171/2010.3.SPINE09369

Merloz, Tonetti, Cinquin, Lavallée, Troccaz, & Pittet. (1998). Computer-assisted surgery: Automated screw placement in the vertebra. *Memoires de l'Academie de chirurgie*, *123(5)*, 482–490. https://doi.org/https://doi.org/10.1016/s0001-4001(99)80077-4

Huber, W. (2008). Historical remarks on martin kirschner and the development of the kirschner (k)-wire. *Indian journal of plastic surgery*, *41*(1), 89–92. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2739562/

von Atzigen, M., Liebmann, F., Hoch, A., Bauer, D. E., Snedeker, J. G., Farshad, M., & Fürnstahl, P. (2021). Holoyolo: A proof-of-concept study for marker-less surgical navigation of spinal rod implants with augmented reality and on-device machine learning. *The international journal of medical robotics and computer assisted surgery : MRCAS*, *17(1)*, 1–10. https://doi.org/https://doi.org/10.1002/rcs.2184

Liu, H., & y Baena, F. R. (2020). Automatic markerless registration and tracking of the bone for computer-assisted orthopaedic surgery. *IEEE Access*, *8*, 42010–42020. https://doi.org/https://doi.org/10.1109/ACCESS.2020.2977072

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, 91–99.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector [To appear.]. http://arxiv.org/abs/1512.02325

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). You only look once: Unified, real-time object detection [cite arxiv:1506.02640]. http://arxiv.org/abs/1506.02640

Redmon, J., & Farhadi, A. (2016). Yolo9000: Better, faster, stronger. http://arxiv.org/abs/1612.08242

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. http://arxiv.org/abs/1804.02767

Petscharnig, S., & Schöffmann, K. (2018). Learning laparoscopic video shot classification for gynecological surgery. *Multimedia Tools Appl.*, *77*(7), 8061–8079. http://dblp.uni-trier.de/db/journals/mta/mta77.html#PetscharnigS18

Mourgues, F., Devemay, F., & Coste-Maniere, E. (2001). 3d reconstruction of the operating field for image overlay in 3d-endoscopic surgery. *Proceedings IEEE and ACM International Symposium on Augmented Reality*, 191–192. https://doi.org/10.1109/ISAR.2001.970537

Ye, M., Johns, E., Handa, A., Zhang, L., Pratt, P., & Yang, G.-Z. (2017). Self-supervised siamese learning on stereo image pairs for depth estimation in robotic surgery. *CoRR*, *abs/1705.08260*. http://dblp.uni-trier.de/db/journals/corr/corr1705.html#YeJHZPY17

Thormälen, T. (2021). Graphics programming cameras: Perspective projection. https://www.mathematik.uni-marburg.de/~thormae/lectures/graphics1/graphics_6_1_eng_web.html#1

Ortiz, L., Gonçalves, L., & Cabrera, E. (2017). A generic approach for error estimation of depth data from (stereo and rgb-d) 3d sensors. https://doi.org/10.20944/preprints201705.0170.v1

Bradski, & Kaehler. (2008). *Learning opencv*. O'Reilly Media, Inc.

Atracsys. (2022). *Fusiontrack 500*. Retrieved July 6, 2022, from https://www.atracsys-measurement.com/wp-content/uploads/2017/10/flyer_fusionTrack500_v3_atracsys_ksa.pdf

Li, P., Chen, X., & Shen, S. (2019). Stereo r-cnn based 3d object detection for autonomous driving. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7636–7644. https://doi.org/10.1109/CVPR.2019.00783

Arun, K. S., Huang, T. S., & Blostein, S. D. (1987). Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-9*(5), 698–700. https://doi.org/10.1109/TPAMI.1987.4767965

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Joskowicz, L., & Hazan, E. J. (2016). Computer aided orthopaedic surgery: Incremental shift or paradigm change? [20th anniversary of the Medical Image Analysis journal (MedIA)]. *Medical Image Analysis*, *33*, 84–90. https://doi.org/https://doi.org/10.1016/j.media.2016.06.036

Opencv. (2022). *Camera calibration and 3d reconstruction*. Retrieved July 6, 2022, from docs.opencv.org/4.x/d9/d0c/group__calib3d.html