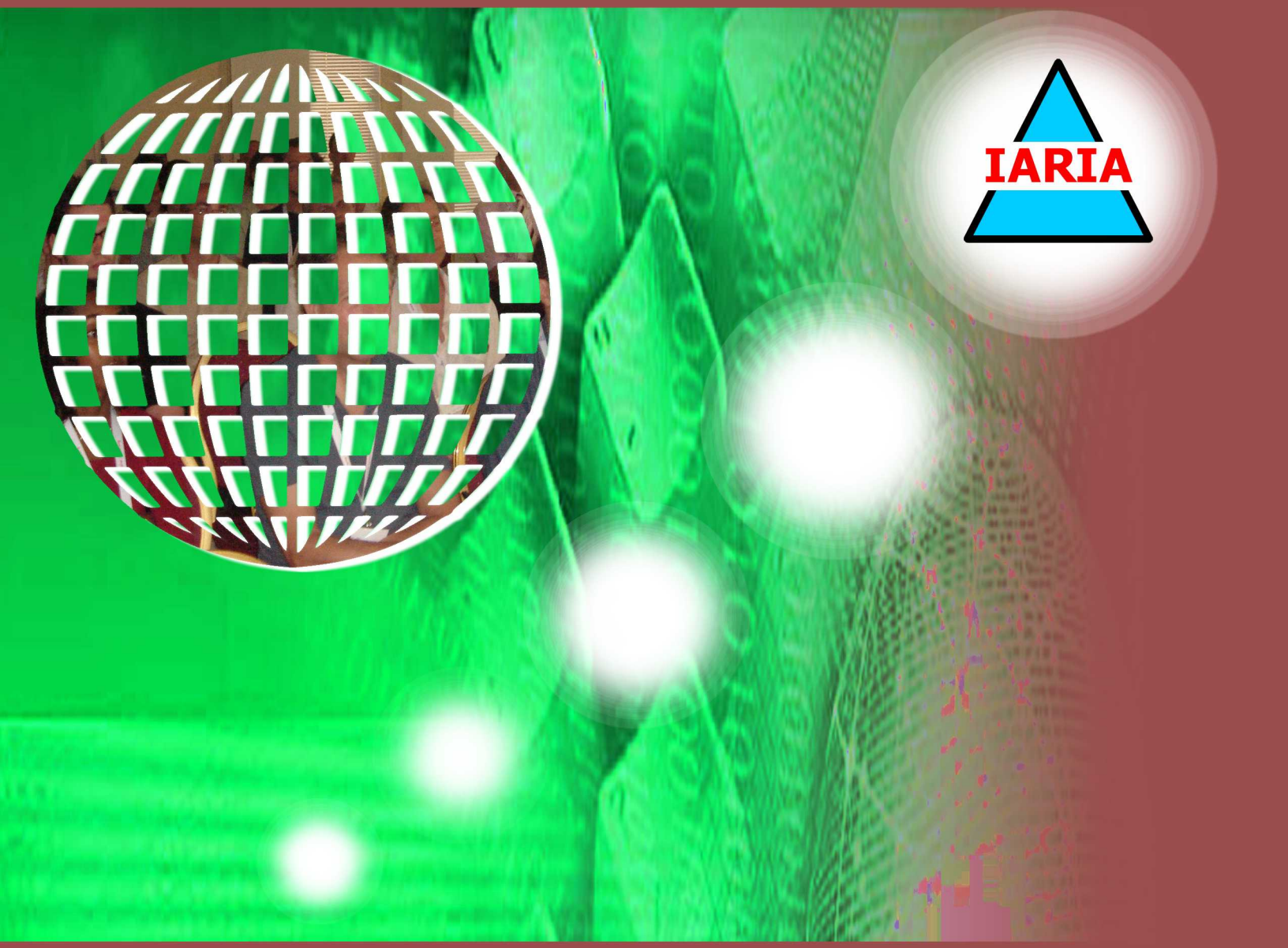
International Journal on Advances in Life Sciences



IARIA

2023 vol. 15 nr. 3&4

The *International Journal on Advances in Life Sciences* is published by IARIA.
ISSN: 1942-2660
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Life Sciences
Volume 15, Number 3 & 4, 2023

Dirk Malzahn, OrgaTech GmbH / Hamburg Open University, Germany
Salah H. Mandil, eStrategies & eHealth for WHO and ITU - Geneva, Switzerland
Herwig Mannaert, University of Antwerp, Belgium
Agostino Marengo, University of Bari, Italy
Igor V. Maslov, EvoCo, Inc., Japan
Ali Masoudi-Nejad, University of Tehran , Iran
Cezary Mazurek, Poznan Supercomputing and Networking Center, Poland
Teresa Meneu, Univ. Politécnica de Valencia, Spain
Kalogiannakis Michail, University of Crete, Greece
José Manuel Molina López, Universidad Carlos III de Madrid, Spain
Karsten Morisse, University of Applied Sciences Osnabrück, Germany
Ali Mostafaeipour, Industrial engineering Department, Yazd University, Yazd, Iran
Katarzyna Musial, King's College London, UK
Hasan Ogul, Baskent University - Ankara, Turkey
José Luis Oliveira, University of Aveiro, Portugal
Hans C. Ossebaard, National Institute for Public Health and the Environment - Bilthoven, The Netherlands
Carlos-Andrés Peña, University of Applied Sciences of Western Switzerland, Switzerland
Tamara Powell, Kennesaw State University, USA
Cédric Pruski, CR SANTEC - Centre de Recherche Public Henri Tudor, Luxembourg
Andry Rakotonirainy, Queensland University of Technology, Australia
Robert Reynolds, Wayne State University, USA
Joel Rodrigues, Institute of Telecommunications / University of Beira Interior, Portugal
Alejandro Rodríguez González, University Carlos III of Madrid, Spain
Nicla Rossini, Université du Luxembourg / Università del Piemonte Orientale / Università di Pavia, Italy
Addisson Salazar, Universidad Politecnica de Valencia, Spain
Abdel-Badeeh Salem, Ain Shams University, Egypt
Matthieu-P. Schapranow, Hasso Plattner Institute, Germany
Åsa Smedberg, Stockholm University, Sweden
Chitsutha Soomlek, University of Regina, Canada
Monika Steinberg, University of Applied Sciences and Arts Hanover, Germany
Les Sztandera, Thomas Jefferson University, USA
Jacqui Taylor, Bournemouth University, UK
Andrea Valente, University of Southern Denmark, Denmark
Jan Martijn van der Werf, Utrecht University, The Netherlands
Liezl van Dyk, Stellenbosch University, South Africa
Sofie Van Hoecke, Ghent University, Belgium
Iraklis Varlamis, Harokopio University of Athens, Greece
Genny Villa, Université de Montréal, Canada
Stephen White, University of Huddersfield, UK
Levent Yilmaz, Auburn University, USA
Eiko Yoneki, University of Cambridge, UK

International Journal on Advances in Life Sciences
Volume 15, Numbers 3 & 4, 2023

CONTENTS

Study of Generalization Performance on Non-Contact Estimation of Lumbar Load Using Webcam Image by Deep Learning for Stationary Standing Forward Bending Posture

Riku Nishimoto
Kochi University of Technology
Miyanokuchi 185, Tosayamada, Kami, Kochi, Japan
Kochi, Japan
email: nishimoto.riku19990917@gmail.com

Kyoko Shibata
Kochi University of Technology
Miyanokuchi 185, Tosayamada, Kami, Kochi, Japan
Kochi, Japan
email: shibata.kyoko@kochi-tech.ac.jp

Abstract - **To prevent lumbago, it is effective to have a system that enables people to improve their habitual bad posture. Therefore, we will develop a method of estimating body load without user burden for constant observation of posture. Hence, this study proposes the use of a web camera, which everyone has and can acquire images on a daily basis without any burden, as a non-contact sensing method, and the use of deep learning as a means of estimating body load from web images. Deep learning models are created by deriving body load values using musculoskeletal analysis based on skeletal position coordinates extracted from posture images and labeling the images with these as true values. Thus, if a pre-trained deep learning model is created in advance, body load can be estimated from images alone, without the use of specialized software or cloud communication. If it is possible to easily visualize one's own body load in daily life, the system can be developed to provide feedback on posture evaluation and improvement plans based on the estimated body load. We consider that this will further increase the users' awareness of improvement and lead to the maintenance and promotion of health. In this paper, as the first step, a deep learning model is created for a stationary standing forward bending posture, and the accuracy of the lumbar load estimation by the deep learning model is evaluated. The results of individual learning using untrained data allowed us to estimate the lumbar load with high accuracy. Hence, the possibility of applying the proposed method to certain individuals is indicated. The other is, the results of ensemble learning confirmed models with high and low accuracy. Hence, the deep learning models that estimated untrained participants showed large variations in accuracy and insufficient generalization performance. Discussion of the results confirms that data bias is a contributing factor to the accuracy loss and indicates the possibility of obtaining generalization performance by improving data bias.**

Keywords- Deep learning; Single camera; Estimation; Musculoskeletal model simulation; Lumbar load.

I. INTRODUCTION

Many people of all ages and genders experience lumbago, and lumbago has become a social problem because of its potential impact on daily life. One of the causes of lumbago is the habit of a broken posture, which is very demanding on the body. From this, to prevent lumbago, it is useful to constantly observe posture in daily life. In cases where posture is out of balance, it is effective to have a system that allows people to improve their posture by themselves. To achieve this, this research group has been considering the quantitative estimation of the load on the lower back in order to determine whether the posture is good or bad. In the past, the lumbar region has been measured using optical motion capture, wearable inertial sensors, and bending sensors to non-invasively estimate lumbar load using biomechanics and statistics [2] [3] [4] [5] [6]. These estimation results showed qualitatively similar trends to the measured lumbar load ratios of Nachemson et al. [7] and Wilke et al. [8] and confirmed the usefulness of the estimation method. However, since specialized equipment and analysis are required, and users are burdened during measurement and estimation, it is difficult to apply this method to the observation, estimation, and evaluation of posture in daily life.

Therefore, in this study, as a way to reduce the burden on the user during measurement, consider using an easily accessible, non-wearing sensing device. Muto et al. [9] evaluated the posture of an elderly person using Kinect v2 for Windows (Microsoft), a depth camera, as a non-contact sensing device. However, the depth cameras essential to this research are not widely available to the public. On the other hand, several systems have been commercialized to evaluate posture based on the skeletal position that is detected by AI from 2D images that lack depth information (e.g., Posen [10]). Although joint angles and other factors are visualized in these systems, however, the loads applied to the body are not quantified. Hence, this study proposes a method for estimating body load using AI from a single camera image, which is a readily available device, as a method of constantly observing posture by self and quantitatively estimating body load [1]. If the proposed method can be realized, by creating a deep learning model in advance using specialized software, it will then be possible to visualize one's own body load in daily life simply by inputting posture images, without going through the cloud. In addition, it can be developed into a system that provides feedback on posture evaluation and improvement plans and evaluations based on this, it will

enable posture condition to be evaluated without burden and lead to the prevention of lumbago.

In this paper, as the first step in creating the proposed system, the deep learning model is created with only the lumbar load as the body load and the posture as a static standing forward-bending posture. After that, the proposed method will be evaluated by the accuracy of the lumbar load estimation using the deep learning model created. In the previous paper [1], a deep learning model was created using multiple experimental participants as training data and estimation using untrained data from participants used for training data. The results were shown to be useful, as high correlations and small errors were identified. However, scope of application remains unclear since the estimation of training participants by a deep learning model was created using several participants. Therefore, this paper verifies scope of application of the proposed method. First, it is verified that the proposed method is applicable to specific individual. A deep learning model is created using specific participant as training data, and its accuracy is confirmed by estimation using untrained data from the same participant. Furthermore, to verify the generalization performance of the proposed method. A deep learning model is created using multiple participants as training data, and the accuracy is confirmed by performing estimation on untrained participants.

The rest of this paper is organized as follows. Section II describes the methods of the lumbar load estimation system proposed in this paper. Section III determines the criteria for evaluating the accuracy of the deep learning model. Section IV provides the methods and conditions for creating the lumbar load estimation system proposed in this paper, and discusses the experimental results based on the evaluation criteria identified in Section III. Section V discusses the results of Section IV. Finally, the conclusions close the paper.

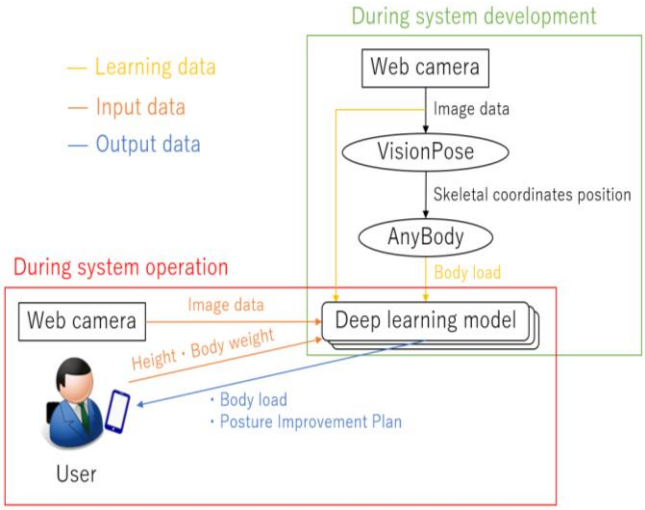


Figure 1. Overview of the proposed body load estimation system. [1]

II. LUMBAR LOAD ESTIMATION SYSTEM

This section describes the system proposed in this study. Figure 1 shows an overall view of the system to be developed in this study, as proposed in the previous report [1]. During system operation, user inputs an image of his/her posture, along with his/her height and weight, into the system, which estimates the body load and outputs improvement plans based on this. To achieve this, a deep learning model is created in advance during development.

A method for deriving the body load to be learned by the deep learning model is described. Tagawa et al. [11] proposed a device to visualize the dynamic load of various body parts from video alone using skeletal detection software. In this system, the body load is calculated using the Newton-Euler method based on the coordinates of the detected skeletal position. However, it is not suitable for estimating static posture, which is the participant of this study, because no acceleration occurs. In addition, it is difficult to obtain an accurate body load from an estimation based on skeletal position alone, because muscle activity and other factors cannot be considered. Therefore, in this study, body load is derived using AnyBody [12]. AnyBody is a musculoskeletal analysis software that can derive various human body information by creating a virtual human body model from skeletal positions. Also, AnyBody can be obtained the account the amount of muscle activity and other factors to determine the force, moment (torque), and muscle tension applied to a region. In the field of healthcare, much research has been conducted that make effective use of AnyBody. Previous research used AnyBody to analyze the effects of age and height on the lumbar region during manual material handling [13] and the effects of lumbar disc herniation on spin loading characteristics [14]. However, the input data for AnyBody are the skeletal coordinate positions of the human body. An optical motion capture camera is generally used, although this device cannot obtain information from the images. Therefore, AI skeletal detection software is used to detect skeletal coordinates from images. The skeletal coordinates detected using such software are used as input data to AnyBody. Thus, in this study, the skeletal position coordinates are detected from images using VisionPose [15]. VisionPose is one of the AI skeleton detection software, a highly accurate AI posture estimation engine that can detect skeletons from 2D camera images without using markers or depth sensors. VisionPose detects a total of 30 skeletal positions, including the hip and shoulder joints shown in Figure 2. Hence, in this study, the load applied to the body using AnyBody is derived from the skeletal position coordinates detected using VisionPose from the images. Out of the 30 locations detected by VisionPose shown on the right in Figure 2, the 15 locations in deficit are used to derive the load by AnyBody. After that, a deep learning model is created by labeling this as the true value with the image. As described above, during system development, specialized software such as AnyBody and VisionPose is used to create deep

learning models. Then, based on this, when the system is used by the user, a structure will be built to enable estimation using only AI applications, without the need for several specialized software. These are the primary characteristics of the proposed system. It is a novel approach to combine AnyBody with AI.

In this paper, since we focus on lumbago, we use the lumbar load as the body load. In this process, a deep learning model is created for each body load that is appropriate for the posture to be observed because AnyBody can derive the load that occurs in any region from single image data. That is, many body loads can be estimated and visualized by the proposed system from single image data.

Next, the lumbar loading used in this paper is described. According to previous research [16], positive agreement was observed between in vivo measurements of disc compression forces between L4L5 and the values derived by AnyBody, demonstrating the suitability of the AnyBody model. Based on these results, this paper uses the compression force of the intervertebral disc between L4L5 derived from AnyBody as the lumbar load. In a previous report [17], as a preliminary step in creating a deep learning model, we evaluated the compression force of intervertebral disc between L4L5 derived by AnyBody using the skeletal position coordinates detected by VisionPose from web images of standing forward bending posture. This result showed an increasing trend of disc compression force with forward bending of the upper body, as measured by Nachemson et al. [7]. In response to this result, as one of the training data for the compression force of the deep learning model, the intervertebral disc between L4L5 derived by AnyBody is used as the true value in this paper.

III. Establishment of Criteria for Evaluating the Accuracy of Deep Learning Models

In deriving the true value of the disc compression force for the proposed method, the relationship between the error in skeletal detection by VisionPose and the anterior tilt angle and the disc compression force will be clarified. This determines the evaluation criteria for the accuracy of the deep learning model to be created.

A. Experiment

Three male participants (age 21 ± 1.00, height 1.70 ± 0.02 [m], weight 67.0 ± 1.70 [kg]) agreed to participate in the experiment in advance after obtaining approval from the University Ethics Committee and explaining the experimental details to the participants. A standing static posture image is acquired for them to obtain the anterior tilt angle and lumbar load by skeletal detection with VisionPose. One webcam (StreamCam: logicool) is used to get video. The camera is placed at the distance of 3 [m] from the center of the participant's body and at a height of 0.85 [m] from the floor. The movies are shot at 1080p/30fps. Three pictures are taken in each of the following conditions using the webcam: upright posture (0 degrees), 10 degrees, 20 degrees, and 30 degrees of forward tilt angle of the upper body. The angle ofc forward bend is determined by pressing the board against the lower back and measuring with a digital angle meter.

Based on the obtained images, one is, the skeletal position coordinates indicating the body center shoulder and hip positions detected by VisionPose are used to calculate the anterior tilt angle using a trigonometric function. After that, the error is calculated from the results of the measured and calculated values. The other is, based on the images, the skeletal position coordinates are detected from each image using VisionPose, and the skeletal position coordinates are input to AnyBody to derive the compression force of intervertebral disc between L4L5. In this process, the height and weight in the human body model in AnyBody are standardized to the participant's average in order to eliminate differences in the participant's physique in the derived values.

B. Estimation Results

Table I shows the error between the calculated and measured values of each forward tilt angle of the upper body, and the compression force of the intervertebral disc between L4L5 derived by AnyBody. Table I shows that the average absolute error of the forward tilt angle of the upper body detected by VisionPose is 3.20 [°]. Furthermore, the mean of standard deviation of the derived disc compression force between L4L5 was 13.1 [N], which is approximately 2.00 [%] of the mean body weight, indicating a high accuracy with little variation between participants of the data. In addition, from the derivation results shown in Figure 3, it can be read that the L4/L5 intervertebral disc

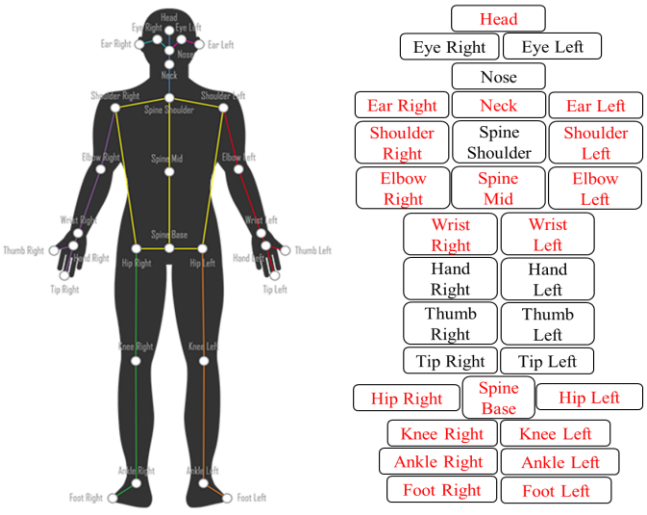


Figure 2. Skeletal position coordinates to be detected by VisionPose. Skeletal positions to be used for AnyBody are shown in red. (Source [15] on the left of the image)

TABLE I. MEAN ABSOLUTE ERROR BETWEEN CALCULATED AND MEASURED VALUES AND COMPRESSION FORCE OF INTERVERTEBRAL DISC BETWEEN L4L5 AT EACH UPPER BODY FORWARD TILT ANGLE

Upper body forward tilt angle [°]	0	10	20	30	Mean value
Mean absolute error of angle [°]	2.17 ± 0.3	2.58 ± 1.1	4.58 ± 0.9	3.46 ± 4.3	3.20 ± 0.924
L4/L5 intervertebral disc load [N]	365 ± 16.4	584 ± 6.50	798 ± 10.3	998 ± 19.2	

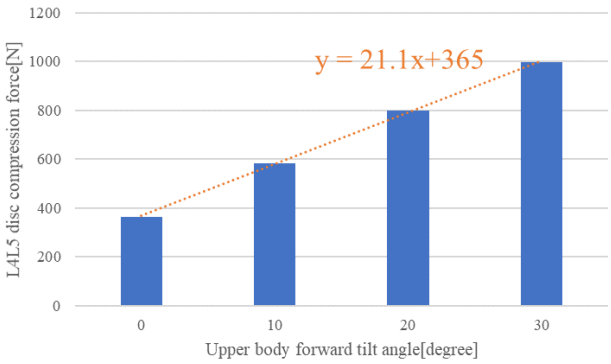


Figure 3. Compression force of intervertebral disc between L4L5 derived using AnyBody for each angle of forward tilt of upper body.

load increases almost linearly in 10-degree increments. The estimation results captured the trend of increased lumbar loading due to forward tilt of the upper body, as described in orthopedic clinical practice. Therefore, the lumbar load derived by AnyBody from the skeletal position coordinates detected by VisionPose can be used as the true value of the training data for the deep learning model. Following the results, the slope of the linear function of the approximate line was calculated to obtain an average change in compressive force per unit angle of 21.1 [N]. These results will be used to evaluate the deep learning models to be created in subsequent sections.

IV. ESTIMATION OF LUMBAR LOAD USING DEEP LEARNING MODEL

This section describes the experimental methods used to collect training and validation data, the preprocessing applied to the measured data, and the training conditions for CNN. Furthermore, the created deep learning model is used to estimate the lumbar load and confirm its accuracy.

A. Experiment

Three male participants (age 23.2 ± 0.748, height 1.73 ± 3.49 [m], weight 67.2 ± 4.35 [kg]) agreed to participate in the experiment in advance after obtaining approval from the University Ethics Committee and explaining the experimental details to the participants. To efficiently acquire posture images for use in the creation and accuracy validation of the deep learning model, video is captured for the forward bending motion of standing posture. The equipment used and camera locations are the same as in the experiment in Section III. The body gradually bends from an upright standing posture to about 30 degrees in 2 seconds, then the body gradually raises in 2 seconds to an upright standing posture. This is taken as one trial, and 5 trials are obtained. A total of three videos are obtained for each participant.

B. Estimation Methods

In this estimation, it is desirable to obtain the posture load at a specific point in time, so frame-by-frame images should be used for learning and estimation, rather than processing with video that includes time information. Hence, the video obtained by the experiment for a total of 15 trials for 5 using the Python module OpenCV (image processing library). The video of each participant is converted to an image at each frame rate, generating 1800 images per participant for a total of 9000 images. In addition, the lumbar load to be used as the true value is obtained by using VisionPose to detect the skeletal position coordinates from the videos of 15 participants in the trials. The first through fourth images of each trial as training data and the fifth image as validation data. The training data for each model are 1440 images and the of 5 participants in the same experiment. The derived disc compression force is normalized by dividing it by height and weight to eliminate differences due to body size. The normalized values are labeled as the true values for each frame of training data to create a deep learning model. After that, using the deep learning model created, estimation of lumbar load is performed on the validation data, and the normalized values are converted to disc compression force [N] by multiplying by height and weight.

TABLE II. CNN LEARNING CONDITIONS

		Set value
Batch size		64
Classes		100
Epochs		200
Dropout		0.2
Convolution layer	Filter size1	32
	Filter size2	64
	Stride	1
Pooling layer	Size	(2, 2)
Fully connected layer		64

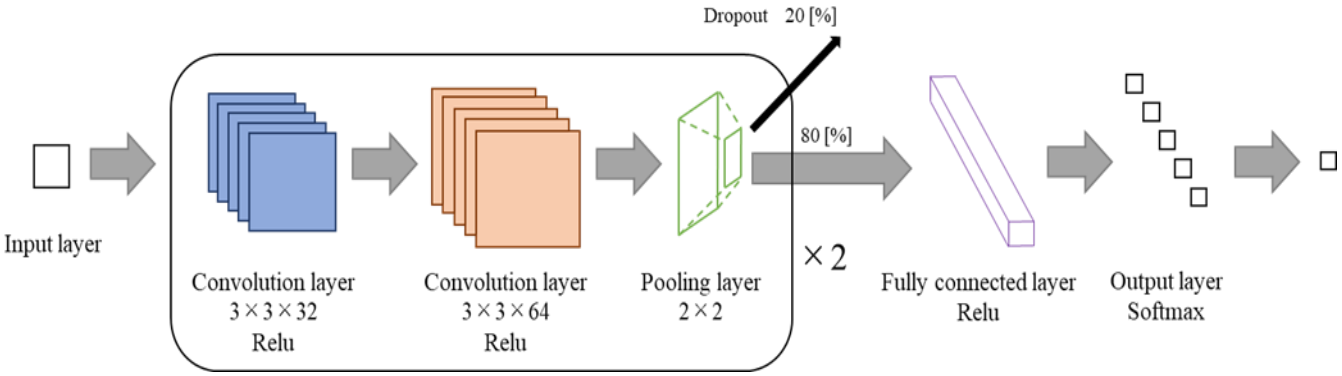


Figure 4. The structure of the CNN learning conditions.

In this paper, we use Convolutional Neural Network (CNN) for deep learning to estimate lumbar load. Numerical estimation is based on images, we consider that the same mechanism can be used for estimation as in the classification problem. The values of each parameter are shown in Table II. Figure 4 shows the structure of the CNN learning conditions used in this report. The CNN structure in the deep learning model consists of an input layer, followed by two convolutional layers, one pooling layer, two convolutional layers with dropout to prevent overfitting, one pooling layer, smoothing to prevent dropout to prevent overfitting, and output to an output layer after passing through all coupled layers one layer. The Relu function is used as the activation function in the convolution layer, the Relu function is used in the all-coupling layer, the Softmax function is used in the output layer, and Adam [18] is used for optimization. Keras Documentation [19] was used to create the above structure in Python. Keras.Callbacks.EarlyStopping is used as the termination condition, with the training error used as the monitor and auto as the mode.

C. *Individual Learning*

In this section, a deep learning model created for an individual confirms the applicability of the proposed method to a specific individual. A total of five deep learning models (Models A, B, C, D, and E) are created for each validation data, which are 360 images. The accuracy of the lumbar load estimated by the deep learning model is evaluated from each of the 360 images of the validation data. Figure 5 plots the estimates for each angle of forward tilt. The anterior tilt angle is calculated using a trigonometric function with the skeletal position coordinates indicating the body-centered shoulder and hip positions detected by VisionPose, based on the upright posture as 0 [°]. All model estimation results captured the trend of increased lumbar load due to upper body forward tilt as described in orthopedic clinical practice. Then, the lumbar load derived from the same verification data using AnyBody is compared to the estimated value as the true value. Figure 6 plots the estimated values from the deep learning model and the true values derived by AnyBody. Table III shows the Pearson's correlation coefficient and mean absolute error for each deep learning model. Pearson's correlation coefficients were 0.993 at maximum, 0.978 at minimum, and 0.987 ± 0.00770 at mean, indicating a high correlation in all models. In addition, the mean absolute error between the deep learning estimates and the true values derived by AnyBody was a maximum of 28.8 [N], a minimum of 22.5 [N], and an average of 26.3 ± 5.22 [N]. This is approximately 3.91 [%] of the average weight. Further, based on the results of the experiment described in Section III-B, the compression force changes by 21.1 [N] per 1 [°] of forward tilt angle. The average absolute error of the results of this experiment is equivalent to an error of 1.25 [°] of forward tilt angle, which is smaller than the average detection error of 3.20 [°] for the forward tilt angle in VisionPose. Therefore, the error is small.

In response to this result, the deep learning model for the standing forward bending posture that was created was able to estimate the lumbar load of the participant with high accuracy, and the proposed method is applicable as a lumbar load estimation method for the individuals used in the training. Thus, the user can check the posture change by own self by creating a deep learning model specialized for own self in advance.

TABLE III. PEARSON'S CORRELATION COEFFICIENT AND MEAN ABSOLUTE ERROR FOR EACH DEEP LEARNING MODEL CREATED FOR AN INDIVIDUAL

	Model A	Model B	Model C	Model D	Model E
Pearson's correlation coefficient	0.983 ± 0.0101	0.994 ± 0.00172	0.993 ± 0.00271	0.978 ± 0.0178	0.990 ± 0.00617
Mean absolute error [N]	28.4 ± 4.05	25.1 ± 3.79	26.8 ± 5.91	28.8 ± 7.39	22.5 ± 4.94

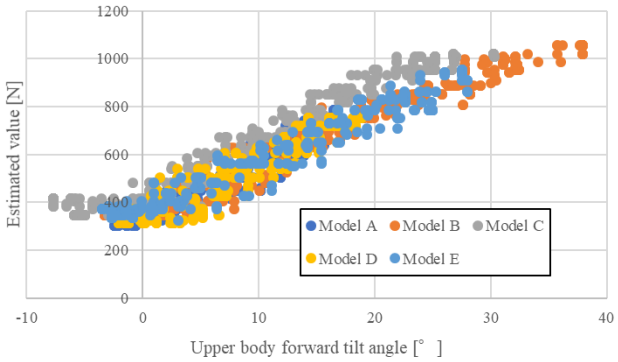


Figure 5. Intervertebral disc compression forces estimated for untrained data by a deep learning model created for an individual at each anterior tilt angle. For example, Model A plots the results of a deep learning model created for participant a using the first through fourth images of each trial as training data, and the fifth image of each trial is estimated as validation data.

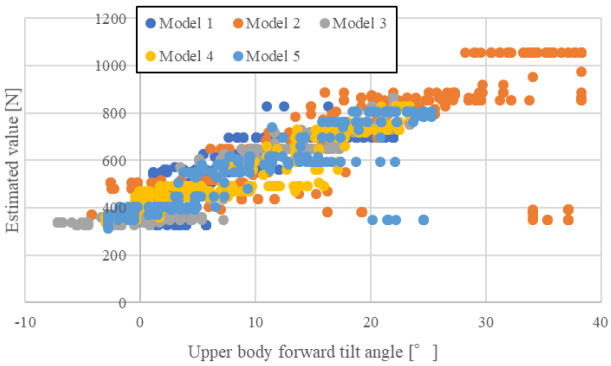


Figure 7. Intervertebral disc compression forces estimated for untrained persons by deep learning models trained on multiple people for each anterior tilt angle. For example, Model 1 plots the results of a deep learning model created with participants B, C, D, and E as training data, with participant A estimated as validation data.

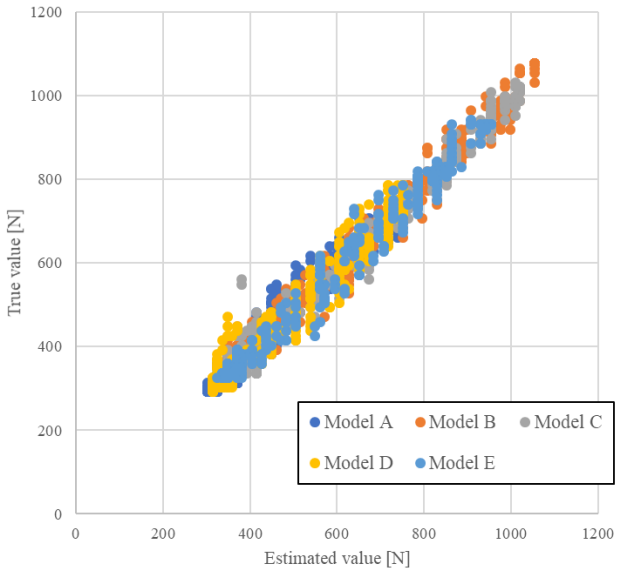


Figure 6. Comparison of disc compression forces estimated for untrained data by a deep learning model trained on an individual with the true values derived by AnyBody.

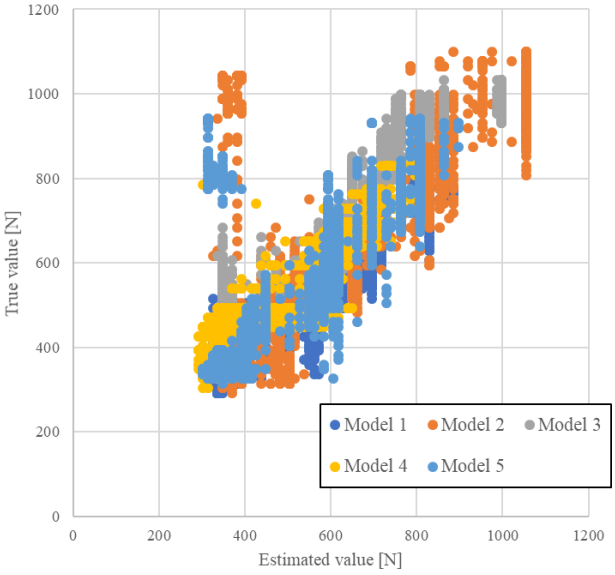


Figure 8. Comparison of disc compression forces estimated for untrained participants by a deep learning model trained on multiple people with the true values derived by AnyBody.

D. *Ensemble Learning*

In this section, a deep learning model created for several participants is used to verify the generalization performance of the proposed method. A total of five deep learning models (Models 1, 2, 3, 4, and 5) are created by cross-validating four of the five participants as training data and one as validation data. The training data for each model are 7200 images and the validation data are 1800 images. Evaluate the accuracy of the lumbar load estimated by the deep learning model from each of the 1800 images of the validation data. Figure 7 plots the estimated values for each angle of forward tilt. Although there were some outliers, all models captured the trend of increased lumbar load due to forward tilt of the upper body, as described in orthopedic clinical practice. Then, the lumbar load derived from the same verification data using AnyBody is compared to the estimated value as the true value. Figure 8 plots the estimated values from the deep learning model and the true values derived by AnyBody. Table IV shows the Pearson's correlation coefficient and mean absolute error for each deep learning model. Pearson's correlation coefficient was 0.966 at maximum, 0.800 at minimum, and 0.888 ± 0.0274 at the mean, indicating a high correlation, although inferior to individual learning. However, Figure 8 shows that

TABLE IV. PEARSON'S CORRELATION COEFFICIENT AND MEAN ABSOLUTE ERROR FOR EACH DEEP LEARNING MODEL WITH MULTIPLE PEOPLE TRAINED

	Model 1	Model 2	Model 3	Model 4	Model 5
Pearson's correlation coefficient	0.947 ± 0.0226	0.801 ± 0.0357	0.966 ± 0.00509	0.921 ± 0.0237	0.803 ± 0.0498
Mean absolute error [N]	54.2 ± 10.4	89.4 ± 9.44	126 ± 9.49	57.0 ± 6.13	69.5 ± 13.8

outliers were observed when the estimated values were between 300 and 400 [N]. In addition, the mean absolute error between the deep learning estimates and the true values derived by AnyBody was 126 [N] at maximum, 54.2 [N] at minimum, and 79.2 ± 9.84 [N] on average. This is approximately 11.9 [%] of the average weight. Based on the results of the experiment described in Section III-B, the compression force changes by 21.1 [N] per 1 [°] of forward tilt angle. The average absolute error of the results of this experiment is 3.75 [°] of forward tilt angle, which is a higher value than the average detection error of 3.20 [°] for the forward tilt angle in VisionPose. Therefore, the error is large.

However, Model 1 with the smallest average absolute error has an error equivalent to a forward tilt angle of 2.57 [°], which is smaller than the average detection error of 3.20 [°] for the VisionPose's forward tilt angle, and thus can be estimated with a small error.

The deep learning models created for the standing forward bending posture target did not show sufficient generalization performance in estimation for untrained participants, due to variations in accuracy caused by some models satisfying the evaluation criteria and others not.

V. CONSIDERATION

The deep learning model created in Section IV-3 was used to estimate the lumbar load of untrained participants in several participants. The results showed that the estimation accuracy varied and the proposed method did not demonstrate sufficient generalization performance. This section considers the causes of this result and offers prospects for improving the accuracy of deep learning models.

First, Figures 9 and 10 show the error rate and accuracy of Model 1. Model 1 has a relatively good Pearson's correlation coefficient and mean absolute error, with few outliers, among all deep learning models. Figure 9 shows the error rate per epoch during training for the Model 1 deep learning model. In the training data, the loss function decreases as the number of epochs increases. However, in the validation data, the loss function increases after a certain point. Furthermore, Figure 10 shows the accuracy per epoch during training for the Model 1 deep learning model. In the training data, the percentage of the accuracy increases as the number of epochs increases. However, in the validation data, there is no change in the accuracy value at a certain point in time. These figures suggest that overfitting has occurred. Citation [20] has been validated using MNIST and states that one of the causes of overfitting is lack of data. The 7200 training data for the deep learning model created in this paper are extremely small compared to the 50000 training data for CIFAR-100, a data set with the same number of classifications. This indicates insufficient training data. Hence, the deep learning model is expected to be improved by increasing the training data. However, collecting huge amounts of data through experiments is costly and labor intensive. Therefore, one idea is to artificially edit image data through data expansion, as in previous research [21], to increase the training data without involving any actual experiments. Thus, it is expected to lead to an improvement in the accuracy of deep learning models.

Second, Figure 11 shows the distribution of the image data acquired from the experiment for per normalized disc compression force. Figure 12 shows the distribution of

Figure 9. Error rates for training and validation data for Model 1.

Figure 10. Accuracy for training and validation data for Model 1.

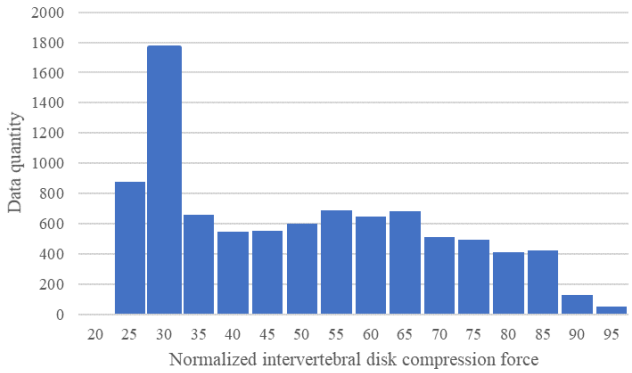


Figure 11. Distribution of the number of image data acquired in the experiment per normalized disc compression force.

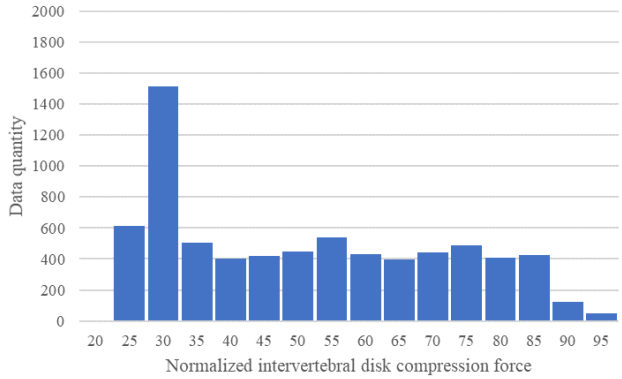


Figure 12. Distribution of the number of images used as training data for Model 1 per normalized disc compression force.

image data per normalized disc compression force used to train Model 1. Figure 12 shows that the amount of image data in the range of 30 to 35 normalized compressions used to train Model 1 was more than the other compressions. Further, a similar trend can be seen in all other models. Hence, the occurrence of outliers due to bias in the number of training data is another factor thought to reduce accuracy. Therefore, to verify the occurrence of outliers due to the bias in the number of training data, the number of image data is randomly deleted so that the number of image data between 30 to 35 becomes 800, the same level as the other range. A deep learning model was created using training data that had been adjusted to reduce bias in a simplified manner by this process, and the accuracy of the model was verified. Figure 13 plots the estimated values for each angle of forward tilt. All results of model estimation captured the trend of increased lumbar load due to forward tilt of the upper body as described in orthopedic clinical practice. Figure 14 plots the estimated values by the deep learning model using the training data after bias adjustment and the true values derived by AnyBody. Table V shows the Pearson's correlation coefficient and mean absolute error for each deep learning model created using the training data after bias adjustment. Pearson's correlation coefficients were 0.971 at maximum, 0.843 at minimum, and 0.936 ± 0.0361 at the mean, reducing the outliers seen in Figure 8 when the estimates were 300~400 [N], and showing a higher correlation than the results in Section IV-3, in which the training data were biased. In addition, the mean absolute error between the deep learning estimates and the true values derived by AnyBody was 126 [N] at maximum, 44.7 [N] at minimum, and 80.1 ± 12.37 [N] on average. Next, the bias adjustment is made to confirm what changes occur in each of the deep learning models. For Model 2 and Model 5,

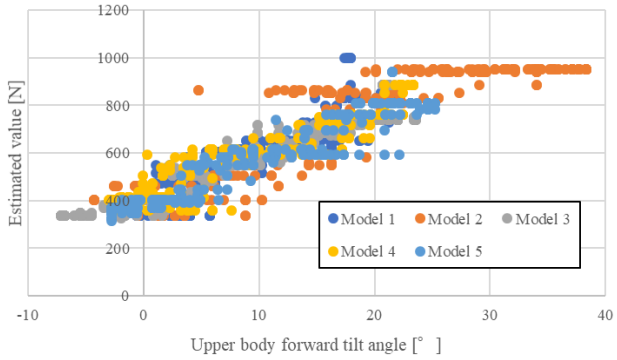


Figure 13. Intervertebral disc compression forces estimated for untrained persons by deep learning models trained on multiple people using bias-adjusted training data for each anterior tilt angle.

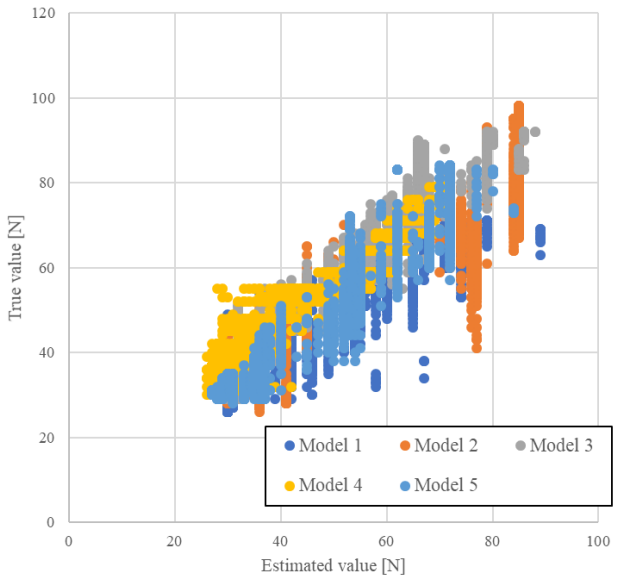


Figure 14. Comparison of disc compression forces estimated for untrained participants by a deep learning model trained on multiple people using bias-adjusted training data with the true values derived by AnyBody.

TABLE V. PEARSON'S CORRELATION COEFFICIENT AND MEAN ABSOLUTE ERROR FOR EACH DEEP LEARNING MODEL WITH MULTIPLE PEOPLE TRAINED USING BIAS-ADJUSTED TRAINING DATA

	Model 1	Model 2	Model 3	Model 4	Model 5
Pearson's correlation coefficient	0.843 ± 0.159	0.938 ± 0.00524	0.972 ± 0.00439	0.960 ± 0.00674	0.968 ± 0.00590
Mean absolute error [N]	80.0 ± 35.2	73.8 ± 3.18	126 ± 9.47	80.3 ± 6.65	44.7 ± 7.32

where many outliers can be seen in Figures 7 and 8, the bias adjustment significantly improved the outliers. Accompanying this improvement was an increase in accuracy in both Pearson's correlation coefficient and absolute mean error. However, Model 1 after bias adjustment was less accurate than Model 1 before bias adjustment for both Pearson's correlation coefficient and the absolute value of the mean error. Otherwise, Model 4 after bias adjustment improved the correlation coefficient and worsened the mean absolute error. Model 3 confirmed no change in accuracy due to bias adjustment.

In the two indices used to evaluate the accuracy of this paper, the accuracy by average of all models was lower than the results in Section IV-3, in which the training data were biased. In some of these cases, the correlation coefficient improved and the mean absolute error worsened, while in other cases both Pearson's correlation coefficient and mean absolute error worsened. Although, the outliers are eliminated in all models. Figures 15 and 16 show the error rate and the accuracy per epoch during training for the Model 1 deep learning model after bias adjustment. Although a minute change, the results of the validation data track the results of the training data, indicating that overfitting can be prevented by correcting the bias in the data. Hence, the possibility of improving the accuracy of the deep learning model was observed by homogenizing the training data. Furthermore, in this study, the training data was acquired through continuous repetition of forward bending movements, thus there is probably room for improvement with respect to this approach.

If these problems can be improved and applied to untrained users, it will be possible to estimate lumbar load with a deep learning model prepared in advance, without having created a deep learning model specific to the individual in advance. That is, the scope of application can be expanded to include untrained user for general use. As a result, the system will not only improve the posture of users who habitually have bad posture, but also enable healthy users to easily use the system as a preventive measure.

VI. CONCLUSIONS

To prevent lumbago, it is effective to constantly observe the posture of daily life. Therefore, we will develop a method to quantitatively estimate and visualize the load applied to the user's own body without any burden on the user. To achieve this, this study proposes a body load estimation method used on a deep learning model that uses web images and body load derived by AnyBody as training data. In this paper, as a preliminary step, we created a deep learning model using only lumbar load as the body load and assuming a stationary standing forward bending posture. After that, the accuracy of estimating lumbar load from web images using the created deep learning model was evaluated.

In the individual learning model, a high correlation was obtained between the estimates by the deep learning model and the true values derived by AnyBody, indicating that the errors were small. Therefore, it is possible to create a deep learning model in advance specifically for a specific user by using specialized software to create a deep learning model to be applied to that user. Thereafter the user to estimate the lumbar load in the target posture simply by inputting images, and can check the posture transition by own self. Hence, improvement of posture suited to individual will be possible without repeated visits to the hospital.

Figure 15. Error rates for training and validation data for Model 1 after bias adjustment.

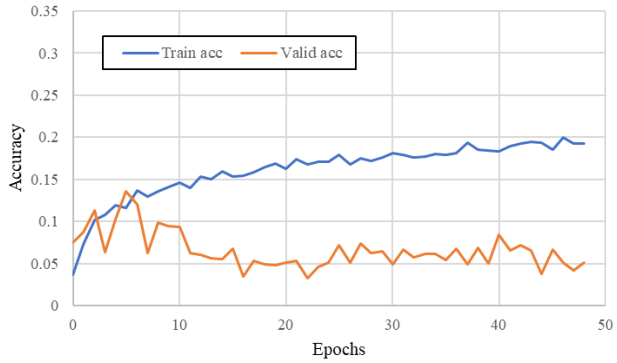


Figure 16. Accuracy of training and validation data for Model 1 after bias adjustment.

The other is, ensemble learning models showed high correlations, but models with high and low accuracy were identified, with a large variation in accuracy, and they did not show sufficient generalization performance. However, it was confirmed that data bias was one of the contributing factors to the lower accuracy. Therefore, if the data bias is improved, the proposed method has the potential to be applicable as a lumbar load estimation method to untrained persons.

The deep learning model created in this paper only covers the estimation of lumbar load in the forward bending posture of the upper body. However, the AnyBody used to derive the lumbar load in this paper can derive various body loads on the body from single measured data. Therefore, various body loads are obtained from a single image data, and a new deep learning model is created using this as training data. That is, it is possible to estimate selected body loads in each posture by acquiring various loads applied to each body part from images of postures that are considered to have a large load on the lumbar, such as hunching back and warped back, including the upper body forward bending posture targeted in this paper, and learning them together with the images. In addition, the accuracy of the deep learning model is improved by optimizing the program through filtering and attention mechanisms.

If these methods can be used to estimate the body load of any posture with a high degree of accuracy, the system can be developed into a system that quantifies and presents the load based on the proposed methods, allowing users to observe their own posture without burden. Thus, raise awareness of improvement, prevent lumbago, and ultimately, maintain and promote health.

REFERENCES

[1] R. Nishimoto and K. Shibata, “Estimation of Lumbar Load from Webcam Images Using Convolutional Neural Network for Standing Forward Bending Stationary Posture,” GLOBAL HEALTH 2022, Lecture No. 70027, 2022.

[2] K. Shibata, Y. Inoue, Y. Iwata, J. Katagawa, and R. Fujii, “Study on Noninvasive Estimate Method for Intervertebral Disk Load at Lumbar Vertebrae,” Transactions of the JSME, vol. 78, No. 791, 2012, pp. 130-141. (in Japanese)

[3] Y. Tsuyoshi, K. Shibata, M. Sonobe, and Y. Inoue, “A Method to Estimate Lumbar Intervertebral Disk Load Using Inertial Sensors for Development of Postures Improvement Support System,” The Japan Society of Mechanical Engineers Chugoku-Shikoku Branch, Student Organization, The 56th Graduation Research Presentation by Students, Lecture No. 112, 2018. (in Japanese)

[4] T. Iituka, K. Shibata, and Y. Inoue, "Estimation of Radius of Curvature of Lumbar Spine Using Bending Sensor for Low Back Pain Prevention," Human-Computer Interaction – INTERACT 2015, vol. 9299 of the series, Lecture Notes in Computer Science, pp. 533-536, 2015.

[5] Y. Suzuki, K. Shibata, M. Sonobe, Y. Inoue, and H. Satoh, "Noninvasive estimation of lumber disk load during motion to improve the posture," AHFE 2017 International Conference on Safety Management and Human Factors, Advances in Intelligent Systems and Computing, vol. 604, pp. 578-588, Code 193719, 2017.

[6] H. Himeda, K. Shibata, and H. Satoh, "Estimation of Load on Lumbar Spine While Walking by Using Multiple Regression Analysis," Advances in Intelligent Systems and Computing, vol. 1205 AISC, pp. 282-288, 2020.

[7] B. J. G. Andersson, R. Örtengren, A. Nachemson, and G. Elfström, “Lumbar Disc Pressure and Myoelectric Back Muscle Activity during Sitting,” I. Studies on an Experimental Chair, Scand J Rehab Med 6, pp. 104-114, 1974.

[8] H.J. Wilke, P. Neef, M. Caimi, T. Hoogland, and L.E. Claes, “New In Vivo Measurements of Pressures in the Intervertebral Disc in Daily Life,” SPINE, vol. 24, No. 8, pp. 755-762, 1999.

[9] Y. Muto, M. Sugou, H. Ito, K. Tsumurai, Y. Hosono, and T. Muto, “Application Method of Kinect for Evoluation of Physical Distortion of Elderly Adult,” Human Interface Society, vol. 19, No. 3, 2017. (in Japanese)

[10] Posen, Inc., Available from: https://posen.ai/, September 2023.

[11] K. Tagawa, T. Kawan, and E. Matsuo, “Development of AI camera for visualizing physical workload,” The 63rd Conference of the Japan Human Factors and Ergonomics Society, No. 2D2-06, 2022. (in Japanese)

[12] AnyBody Technology A/S, Available from: https://www.anybodytech.com/, December, 2023.

[13] T. Chihara, K. Iwahara, and J. Sakamoto, “Effect of age-related muscle weakness and height on L5/S1 compression force during manual material handling,” Transactions of the JSME, vol. 85, No. 876, 2019, p. 19-00125. (in Japanese)

[14] S. Kuai, W. Liu, R. Ji, and W. Zhou, “The Effect of Lumbar Disc Herniation on Spine Loading Characteristics during Trunk Flexion and Two Types of Picking Up Activities,” Journal of Healthcare Engineering vol. 2017, doi: 10.1155/2017/6294503.

[15] NEXT-SYSTEM Co., Ltd., Available from: https://www.next-system.com/visionpose/, December, 2023.

[16] T. Bassani, E. Stucoviz, Z. Qian, M. Briguglio and F. Glbusera, “Validation of the AnyBody full body musculoskeletal model in computing lumbar spine loads at L4L5 level,” Journal of Biomechanics, vol. 58, pp. 89-96, 2017.

[17] R. Nishimoto and K. Shibata, “Estimation of Lumbar Load from Webcam Images Using Musculoskeletal Model Simulation,” The Japan Society of Mechanical Engineers Chugoku-Shikoku Branch, Student Organization, The 50th Graduation Research Presentation by Students, Lecture No. 05b2, 2022. (in Japanese)

[18] J.L. Ba and D.P. Kingma, “Adam: A method for stochastic optimization,” In Proceedings of International Conference on Learning Representations, 2015.

[19] Keras Documentation, Available from: https://keras.io/, August, 2023.

[20] K. Saito “Deep Learning from the Basic,” p. 190, O’Reilly Japan, 2016. (in Japanese)

[21] L. Taylor and G. Nitschke, "Improving Deep Learning with Generic Data Augmentation," 2018 IEEE Symposium Series on Computational Intelligence (SSCI), Bangalore, India, pp. 1542-1547, 2018.

Analyzing and Reporting Wearable Sensor Data Quality in Digital Biomarker Research

Hui Zhang[1], Regan Giesting[1], Guangchen Ruan[1], Leah Miller[1], Neel Patel[1], Chakib Battioui[2], Ju Ji[2], Ming Zhong[1], Andrew David Kaczorek [1], Tianran Zhang[1,3], Yi Lin Yang[1,4]

[1]*Digital Health Office, Eli Lilly & Company, Indianapolis, Indiana, USA*

[2]*Advanced Analytics and Data Science, Eli Lilly & Company, Indianapolis, Indiana, USA*

[3]*Department of Computer Science, Brown University, Rhode Island, USA*

[4]*Department of Computer Science, Purdue University, Indiana, USA*

email: {zhang_hui, rgiesting, ruan_guangchen, miller_leah, patel_neel_k, battioui_chakib, ji_ju, zhong_ming, kaczorek_andrew_david}@lilly.com

tianran_zhang@brown.edu

yang2501@purdue.edu

***Abstract*—Digital Health Technologies (DHT) utilize a combination of computing platforms, connectivity, software, and sensors for healthcare-related uses. Today, these technologies collect complex digital data from participants in clinical investigations, including a large amount of wearable sensor signals. These collected data are used to develop digital biomarkers (dBMs), which can act as indicators for health outcomes for monitoring life quality and measuring drug efficacy. One essential step towards realizing the full potential of these complex digital data is to define the fundamental principles and methods to demonstrate sufficient data quality and fidelity needed for the research. This paper aims to develop a digital data quality assessment framework across the complete data life cycle in dBM research, including data quality metrics and methods to analyze and report digital data quality. Aggregating and reporting digital data quality is often challenging and error-prone. We developed Magnol.Ai, a data platform equipped with data quality assessment and reporting tools that allow us to define data compliance criteria and view data quality reports at different levels in a consumable fashion.**

***Keywords*—*digital health technology*; *connected clinical trial*; *sensor data*; *data quality assessment*; *data visualization*; *digital biomarker*.**

I. Introduction

Digital biomarkers (dBMs) are patient-generated physiological and behavioral measures collected through connected digital devices. The collected data are then used to explain, influence, or predict an individual's health-related outcomes (see [1], [2]). While the development of dBMs invests heavily in advanced analytics, effective results depend on trusted and understood data collected from digital devices. An established data quality assessment framework is thus needed to define the expectation of data, monitor the data for conformance to expectations throughout trials, and report various measures to assess the data quality (see, *e.g.*, [3]). Establishing a meaningful data quality function will help reduce risk throughout the dBM research activities, ultimately ensuring that the criteria for success are met.

Today, we use DHT (see, *e.g.*, [4]) to collect some of the most complex digital data from patients for dBM research. There has been an overall need for better understanding of data, as well as easier access to both data quality and trusted digital data to support operational and analytical activities in the research. Establishing a data quality assessment framework and building software tools to facilitate the assessment is an emerging industry capability. Some unique challenges for this class of data quality strategy include:

- **Complexity of digital data** — We collect some of the most complex digital data in the dBM context, including sensor signals from wearables, patient-reported outcomes from hand-held devices, and labels and annotations processed and used as ground truth information for algorithm development. Handling the wearable sensor data can be challenging. For example, with a sampling frequency of 50Hz, over 4 *million* 3-axial data points are collected from an accelerometer sensor for a single day to understand a patient's daily activities. Similar sensor data streams include, *e.g.*, continuously collected photoplethysmography (PPG) and electrocardiogram (ECG) signals from trial participants.
- **Full-spectrum quality expectations** — Defining quality expectations for digital data and monitoring their conformance to expectations are full-spectrum in the data life cycle. For example, given that data can be collected in a free living environment, scanning the invalid values and noises in wearable sensor signals is often the first profiling step. Identifying the wearable sensor signal's useable (wear-compliant) portions is also a leading data quality function. The ultimate answer to the digital data quality question is the extent that our digital data satisfies the specific requirements needed for dBM analysis.
- **Aggregation and reporting** — Generating various measures to assess digital data quality is not trivial. For example, aggregating compliance information from signal level to the number of analyzable digital measures at the visit and study levels can often be tedious and error-prone. More challenging is reporting data quality in an efficient and effective means across the data life-cycle, and

with more difficulty, at individual participant level, which requires tools to extract and report quantified compliance information, including patterns.

In this paper, we build upon our prior work [1] in the field of data quality assessment. Our contributions can be summarized into two significant parts:

- **Introduction of Magnol.Ai Data Platform**: We introduce Magnol.Ai, a comprehensive data platform tailored to support digital dBM research. At its core, Magnol.Ai incorporates an enhanced data quality assessment framework as an integral component. We begin by offering an overview of the primary digital data categories central to our research focus. Subsequently, we delve into the various metrics employed for profiling digital data quality and their application at multiple aggregation levels.
- **Data Quality Reporting Dashboard**: This paper also presents our innovative Data Quality Reporting Dashboard integrated within Magnol.Ai. The dashboard unifies and streamlines all data quality assessment functions into a cohesive system. It empowers data stewards and quality analysts by allowing them to work seamlessly with specific study data. Users can run processes through interactive workflows and effortlessly generate consumable data quality reports within a centralized, cloud-based ecosystem.

This paper is organized as follows. Section II presents the related work. We present our digital data quality assessment framework and the visual interfaces and tools we developed in Section III. In Section IV, we showcase the data quality reporting portal and the underlying data infrastructure of Magnol.Ai and finally, we conclude the paper in Section V.

II. Related Work

Developing dBMs requires conducting studies in a lab or free-living settings to collect raw sensor data, often with appropriate labels and annotations (*e.g.*, reported patient outcomes). Collection and analysis of wearable sensor data, together with other digital data sets, has thus become an emerging capability needed in dBM development.

Industry players have begun exploring cost-effective and purpose-built solutions in the past few years. For example, the Medidata sensor cloud [5] is used to manage wearable sensor and DHT data for clinical trials. The Koneksa platform [6] provides support to improve compliance monitoring and patient engagement, and other representative efforts to store and deliver raw or processed data from devices in trials, including Evidation [7] and DHDP [8].

Meanwhile, good data is more important than big data in dBM development. Given that wearable sensor data can be collected from participants in a free-living environment, noises, missingness, and invalid values in wearable sensor signals are inherent. To extract and leverage useful and meaningful sensor data, we need to monitor the quality and eventually standardize and process them to support dBM research, as digital data quality is of fundamental importance to developing algorithms for new dBMs (see, *e.g.*, [9] [10] [11]).

In this paper, we are mainly concerned with digital data sets that fall into four general categories:

- *Raw Sensor Signals*. A device typically collects data from multiple sensor signals at varied pre-configured sampling frequencies to minimize study participants' burden under free living conditions. In most cases, the sensor signals are collected in a nonstop $24 * 7$ fashion throughout the entire study, which generally runs between weeks to months. Therefore, assessing potential issues, such as sensor malfunctioning, or wear non-compliance due to participants' behaviors, is critical to ensure data quality can satisfy the downstream analytics needs. Meanwhile, the quality and coverage of sensor data directly correlate to the dBMs derivation, which will be discussed in the later sections of this paper.
- *Scored Data, or Digital Biomarkers*. In addition to raw sensor signals, device companies usually have their proprietary algorithms to analyze sensor data and derive dBMs from it. For example, heart rate and blood volume pulse can be derived from the raw photoplethysmography (PPG) sensor signal. Derived dBMs are at a much lower resolution than the sensor signal, often at the minute or half-minute level.
- *Labels/Annotations*. As algorithms and machine learning models used in developing dBMs become more complex, requirements for large annotated data sets grow. Annotating data for machine learning applications is especially challenging in the biomedical domain as it requires the domain expertise of highly trained specialists to perform the annotations. Annotations can come as interval-based events, with precise timestamps to label the onset and offsets of disease events.
- *Clinical Records*. Apart from raw sensor data and derived dBMs, one yet important piece of data is clinical records that provision key mappings, *e.g.*, device ID to participant ID, participant ID to the treatment cohort, visit dates to treatment phases, *etc*.

Unique challenges arise from these digital data and have made a case for us to develop a data quality assessment framework to define the expectation of these digital data (*e.g.*, completeness, uniqueness, validity, integrity), to monitor the data for conformance to expectations throughout the dBMs trials, and, finally, a user interface as the front-end of Magnol.Ai to display the findings to support operational and analytical activities.

III. Digital Data Quality Assessment Framework

The key functions in our data quality assessment framework should now be clear in Figure 1. The logical series of modeling steps, the problems they induce, and the ultimate resolution of the problems are in the rest of this section as follows.

A. Signal Data Quality Metrics

In the pre-study phase, we establish the Data Transfer Agreement (DTA), to clearly define data quality metrics regarding signal data, including raw sensor signals and dBMs.

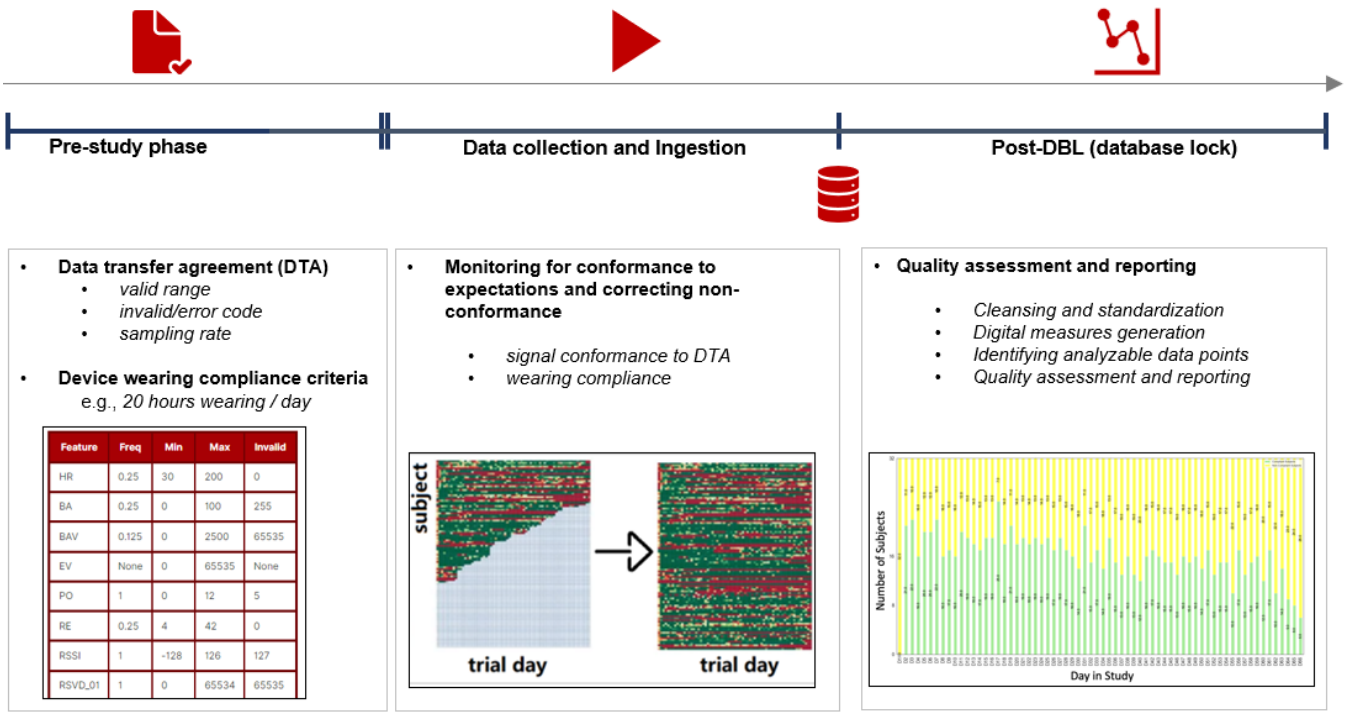


Figure 1: The overall data quality assessment scenario — from establishing the data transaction report (DTA) in the pre-study phase, to compliance monitoring in the live phase, and finally to the quality assessment and reporting in the post-Database Lock (DBL) phase.

TABLE I: EXAMPLE OF A SIGNAL DATA QUALITY METRICS TABLE FOUND IN A TYPICAL DTA DOCUMENT.

Channel	Description	Units	Min Value	Max Value	Invalid Value	Sampling Frequency (Hz)
$accel_x$	Accelerometer X Vector	gravity/1024	-32768	32767	None	50
$accel_y$	Accelerometer Y Vector	gravity/1024	-32768	32767	None	50
$accel_z$	Accelerometer Z Vector	gravity/1024	-32768	32767	None	50
ec	ECG signal	μV	-10000	10000	32767	125
st	Step count	Steps	0	65535	None	1
hr	Heart rate	beats/min	30	200	0	0.25
re	Respiration rate	beats/min	4	42	0	0.25
po	Posture • Laying Down = 0 • Standing = 2 • Walking = 3 • Running = 4 • Unknown = 5 • Leaning = 11	Enum	0	11	5	1

Below we list the typical quality metrics, and Table I gives an example of the data quality metrics table we find in the DTA, where $acce_x$, $accel_y$, $accel_z$ and ec are raw sensor signals, st, po (categorical) are derived dBMs (or, scored data) from accelerometry data, and hr and re are the scored ones from ec.

- **Sampling Frequency** — For raw sensor signals, the sampling frequency is the preconfigured average number of samples obtained in one second. For derived dBMs, it is the resolution of resultant features from analyzing raw sensor data.
- **Valid Range** — For numerical variables (*i.e.*, sensor signals and dBMs), the valid range is indicated by minimum and maximum values that can be measured. For enumerated variables, the valid range is a list of predefined categorical values. One example is the rest classification biomarker, which has the following classes: "awake", "sleep", "toss and turn" and "interrupted".
- **Invalid Value/Error Code** — In addition to the valid range, devices often provision specific invalid values or error codes to indicate different statuses of malfunctioning, which help pinpoint the underlying issue.

B. Signal Data Quality Assessment

Connected clinical trials for dBM research often are conducted under a free living condition, *i.e.*, participants wear sensor devices on a best effort basis using instructions communicated during study enrollment. Inevitably, the free living conditions, potential for device failure or malfunction, and device wearing compliance introduce data issues such as

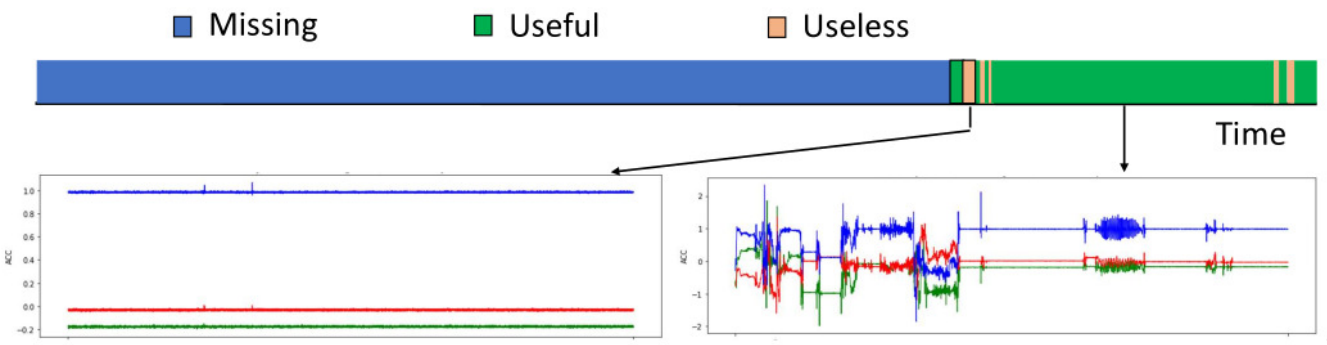


Figure 2: Illustration of sensor signal data issue. Visualized sensor data show different patterns when worn correctly versus incorrectly.

missing data or invalid data collected when participants do not wear or incorrectly wear the devices. Figure 2 illustrates how valid signals (*i.e.*, correctly worn signals) can mix with invalid signals (*i.e.*, incorrectly or not worn signals) in the data collection and how they differ when plotted. Therefore, a *qualitative* means is needed to tell whether a device was operating normally and worn correctly (*i.e.*, **data usefulness**).

To fulfil this goal, the quality assessment is performed in two stages, as discussed in the following.

- **Validity Check**. Data validity checks leverage signal data metrics, as discussed in Section III-A. We immediately know how many valid data points we expect to receive for a sensor signal or dBM using its pre-configured sampling frequency. We can filter out invalid values with a valid value range to get valid data coverage, *i.e.*, coverage of valid data points.
  Since raw sensor signal directly correlates with derived dBMs, we can perform a validity check against the two independently and then align their valid data coverage to check the consistency. We may further overlay device incident events to understand the root cause of observed issues better.
- **Non-wear Detection**. After dropping out invalid data through the validity checking process, the subsequent task is to detect moments when the devices were not correctly worn. The non-wear detection can be challenging as data from such moments can be entirely valid in terms of falling within its valid data range. Instead of reinventing the wheel, we rely on Biobank [12] [13], an accelerometer data processing pipeline whose non-wear detection module is widely adopted as a standard. Below are two key concepts in non-wear detection.
  - *Epoch* — Although data points are collected initially at a high resolution, *e.g.*, 50Hz sampling frequency, the processing is conducted on aggregated values (*e.g.*, 1 or 5 second *short epochs* or 15 minutes *long epochs*) due to the following reasons: (1) collapsing data to epoch summary measures helps to standardize differences in sample frequency across studies; (2) there is little evidence that raw data is an accurate representation of body acceleration, and all scientific evidence so far has been based on epoch averages; (3) collapsing data to epoch summary measures also helps to average out different noise levels making results more comparable across sensor brands.
  - *Non-wear Detection* — Accelerometer non-wear time is estimated based on the standard deviation and the value range of the raw data from *each* accelerometer axis. Classification is done per 30-second epochs based on the characteristics of a larger window centered at these 30-second epochs. Specifically, Biobank identifies stationary periods in 10-second windows where all three axes have a standard deviation of less than $13.0mg$ ($1mg = 0.0098\,m \cdot s^{-2}$). These stationary periods are then used to define whether a window is stationary or not.

C. Signal Data Quality By Granularity

In addition to *qualitative* assessment as discussed in Section III-B, *quantitative* measures that define how much usable data is in a specific period (*i.e.*, **data quality** at different levels) are required before statisticians can begin analysis.

The Data Quality Model. Based on Biobank's non-wear classification on 30-second epoch level, we can further generate data quality that can be used for analysis at different time resolutions. Each phase in our data quality derivation flow is illustrated in Table II to Table V and expanded upon below.

- **Epoch Level** — This table is generated from Biobank's 30-second epoch classification. It serves as the working basis for subsequent data quality tables. Note that we have one additional column, "Subject," to indicate participant ownership of an epoch.
- **Hourly Level** — From the epoch quality table, we can apply a filter to only keep correctly worn epochs and in turn infer hourly data coverage in terms of compliant minutes. This hourly data quality table is the source for data quality reporting at the finest granularity.
- **Daily and Intraday Window Level** — From the hourly data quality table we can summarize the total coverage for each day and produce daily level data quality tables. In addition, for analysis purposes, we are often interested in specific intraday windows from which digital endpoints

TABLE II: EPOCH LEVEL QUALITY.

Subject	Timestamp	Non-wear
1002	2021-09-15 19:15:00	false
...	...	...
1005	2021-10-18 09:45:30	true

TABLE III: HOURLY LEVEL QUALITY.

Subject	Date	Hr	Coverage $(minute)$
1002	2021-09-15	19	45
...	...	...	...
1005	2021-10-18	09	60

TABLE IV: DAILY AND INTRADAY LEVEL QUALITY.

Subject	Date	Coverage $(minute)$	Window
1002	2021-09-15	1440	pa_daily
...	...	...	...
1005	2021-10-18	720	sleep_night

TABLE V: EXTENDED QUALITY WITH EXTERNAL MAPPINGS.

Site	Subject	Date	Trial Day Index	Visit	Coverage $(minute)$	Window
101	1002	2021-09-15	1	0 $(PreTreatment)$	1440	pa_daily
...	...	...	...	...	...	...
103	1005	2021-10-18	32	4	720	sleep_night

are derived — for instance, walking time or step count during the daytime (*i.e.*, daily **p**hysical **a**ctivity) and sleep hours during the nighttime. Thanks to the "Hour" column in the hourly quality table, intraday window coverage can be easily derived by applying filters.

- **Extended Quality with External Mappings** — We can further extend the data quality table with additional mappings when they become available as the study progresses, for instance, mapping between patients and sites/visits, as reported from the clinical operation site. These extra fields allow analysis-specific filtering and aggregation, *e.g.*, to find out which participants have sufficient data and set up individual baselines. We use this table to look for the patients with at least three valid days ($>= 20$ hours of data for a day to be qualified as a valid day) during a pre-treatment visit.

D. Representing Digital Data Quality

Fully understanding the quality of a large dataset, especially one that contains data from wearable device sensors, is not always a trivial undertaking. With numerous considerations to be cognizant of, as discussed in Section III-C, the most logical first step is to present the data with visualizations. Thoroughly understanding the data coverage and quality requires more than one visualization, simply because there is more than one aspect to check. This section presents a family of commonly used visualization examples in our data quality strategy.

- **Identifying Outliers and Missing Data.** Certain metrics must fall between threshold ranges depending on the study and associated data sources. One example is heart rate, which falls within a specified range of 30 to 200 beats/minute for one study. This range is outlined in the DTA for the study and must be applied to all heart rate data points collected. By plotting these signals against the specified thresholds, outliers can be immediately detected by viewing a plot. If outliers exist, further investigation will be completed for that participant's data to see if there are outliers for other metrics. Further, gaps in data can be identified within the same visualization, as demonstrated in Figure 3(a). Detailed data quality reports are generated in conjunction with the visualizations created for displaying outliers and missing data. For example, we convert the signal data from 3(a) to a sequence of colored blocks in Figure 3(b), with green blocks indicating valid sensor signal value in the corresponding period and red indicating missing or invalid signal value identified. In Figure 3(c), we compute the valid data ratio, and therefore can represent the data quality with a numeric value, or with a color from the color palette, keyed to the valid data ratio (see *e.g.*, Figure 3(d)).
- **Data Quality Map with Levels of Detail.** The quality of sensor signal data must be examined on various levels, each offering a specific level of detail. While certain levels are more useful for identifying distinct patterns, we will focus on the hourly, daily, and study levels on both a patient and population level:
  - *Minute-by-Minute Quality Map for a Day* — Examining signals on a minute level can help to identify the minutes where a device may have intermittent connectivity, or more minor issues can be identified and further inspected, as seen in Figure 4(a).
  - *Hour-by-Hour Quality Map for a Trial* — Zooming out, we can look at each hour across all days in the study. The hourly level aggregation mentioned in Section III-C is used to configure the day level plot, shown in Figure 4(b). This figure shows minutes of data coverage for each hour across all study days. This type of visualization allows us to look at compliance trends for a patient that may persist during certain hours of each day. Figure 4(b) shows an interesting device wearing pattern for the participant — taking off the wearable device to charge the battery for a couple of hours in the middle of each day of the trial has resulted in *missing data*, visualized as a sequence

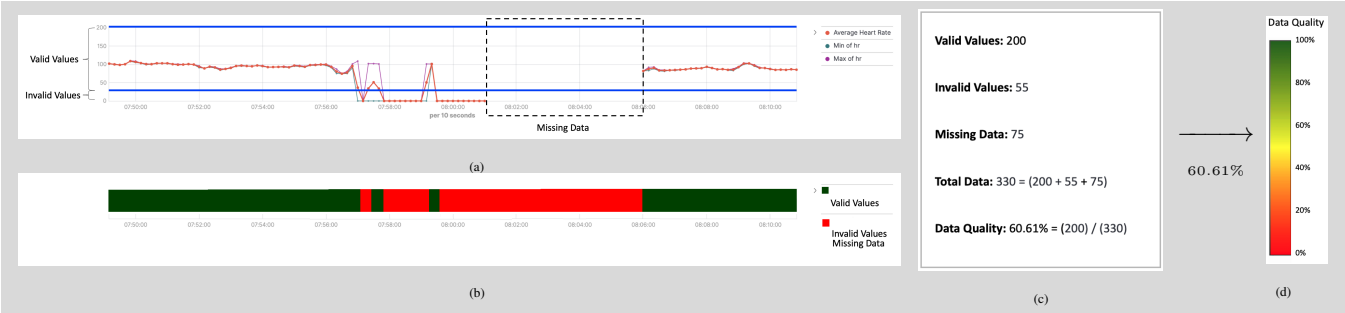


Figure 3: Visualization for sensor data quality. (a) Heart rate data (beats/minute) observed for one participant between 2021-02-15 07:49:00.000 and 2021-02-15 08:11:00.000. Valid range between 30 - 200 beats/minute, as denoted by threshold lines. Invalid data was observed multiple times. Missing data was observed between 2021-02-15 08:01:08.994 and 2021-02-15 08:06:09.000 with nearly 5 minutes of no data. (b) Use colored blocks to represent sensor signal data quality. (c) Deriving numeric representation of the data quality, *i.e.*, valid data ratio. (d) Interpreting data quality with color.

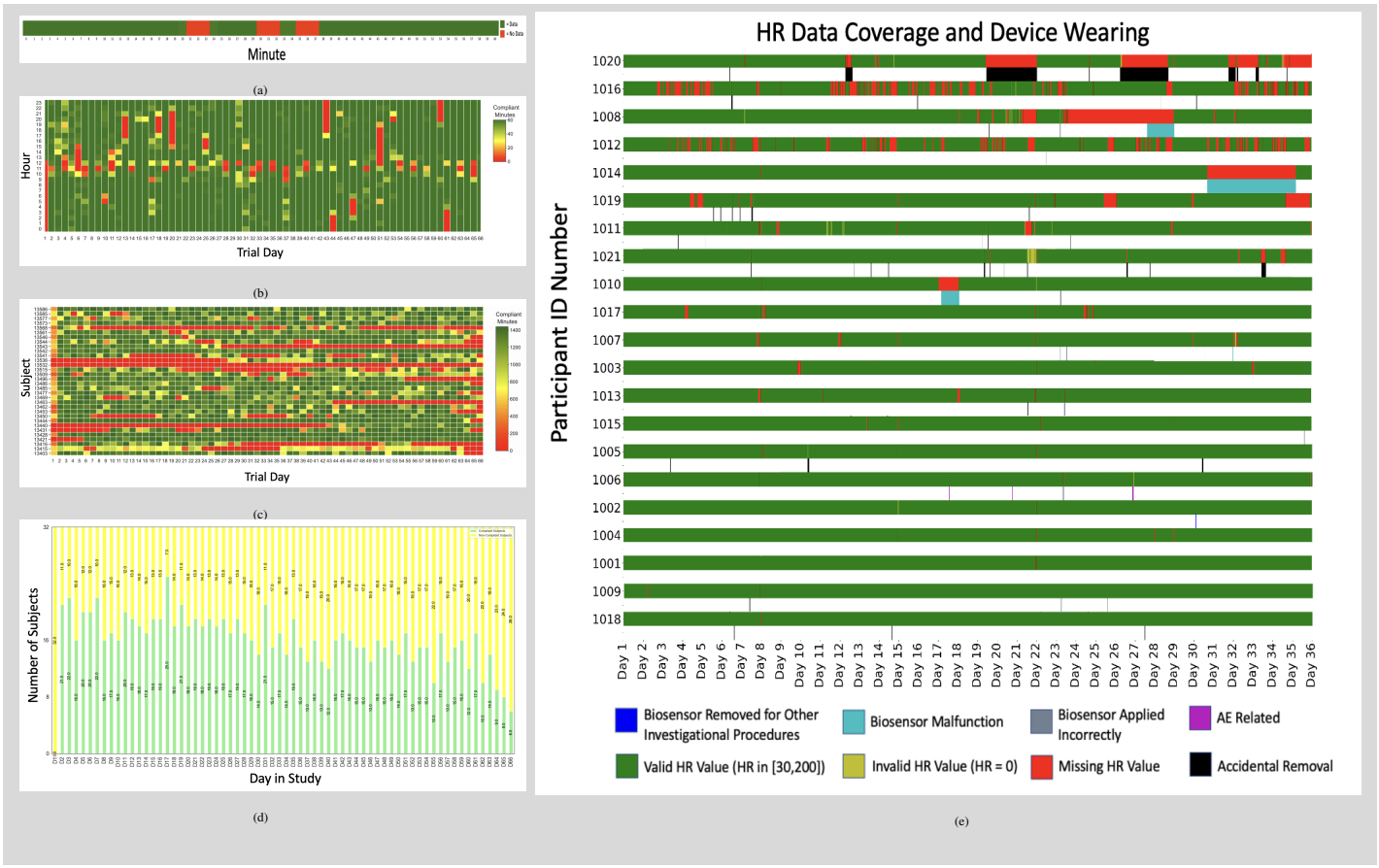


Figure 4: Plots showing (a) minute-level quality representation throughout a participant day, (b) hourly-level quality representation for a participant throughout an entire trial, (c) daily-level quality for a population throughout the entire trial, (d) number of compliant days across all days in a study and (e) data coverage and device wearing issues observed throughout a study.

of red blocks in the center area of the map.

– *Day-by-Day Population-level Quality Map for a Trial* — Plotting data quality for all hours, days, and participants in a study yields the observation of data quality patterns seen in Figure 4(c). This study-level visualization can help us gain insights into the overall data quality at the population level and the compliance trends at the participant level throughout the trials.

– *Compliant Days Throughout a Trial* — In addition to the number of hours per day, it is also useful to view the number of *compliant* In addition to the number of hours per day, it is also useful to view the number of *compliant* days throughout the study, with a definition of compliance dependent on a study's protocol. One

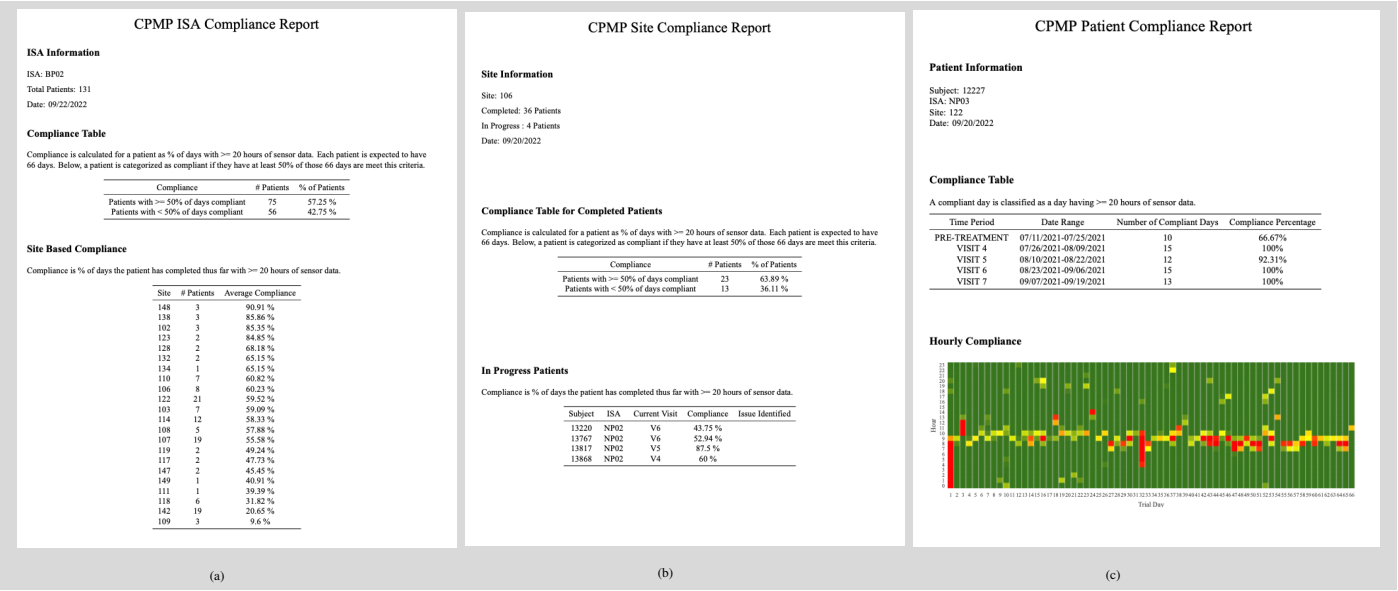


Figure 5: Putting together compliance reports for Intervention-Specific Appendices (ISAs) under Chronic Pain Master Protocol (CPMP). (a) Generated compliance reports on the patient level. (b) Compliance by visit. (c) Customizable compliance report at patient level.

can recognize device-wearing patterns by plotting the number of patients compliant daily in a given study. As seen in Figure 4(d), the number of compliant days in a study decreased due to reduced device wearing as the study progressed.

- **Identifying and Aligning Data Issues.** In many clinical trials, it is a requirement that patients visit a site periodically. Whether it be for receiving dosing of a drug, having their vitals checked, or obtaining a device, information is collected by the sites and stored in various reports. One type of report, device reports, are used during data processing and can help understand the device's overall performance, specifically if any device issues exist. Additionally, information derived from these reports can be used to populate visualizations such as Figure 4(e). By combining this visualization with the information received in site reports, patterns specific to potential device issues and wearing patterns can be derived.
  From the aforementioned data visualizations, various issues and patterns can be identified. When these are paired with actionable recommendations and delivered to the study team promptly, the study team can notify the corresponding site and participant to ensure the issue is rectified. This process leads to a quick turnaround time for potential improvements to data collection and can resolve the challenges that create low compliance in studies.

E. Generating Compliance Reports

Visualizing data is key to understanding data quality, as discussed in Section III-D. However, it is equally important to have a standardized reporting system for compliance to distribute quality and compliance information. Such systems generate reports that outline compliance on three levels: trial, site, and patient. In addition, automated generation allows systems to be configured at the start of a trial and run at set cadences to produce consistent quality assessment reports efficiently.

For each report, regardless of the level or contents, the thresholds used to configure and derive data metrics and visualizations are based on the expectations outlined in the study protocol. Each report aims to give insights into the population's compliance behavior:

- **Trial Summary**: A single comprehensive trial report can be generated and contains metadata regarding the number of patients, sites, and overall compliance percentages.
- **Study-Level Compliance**: A study-level report, such as Figure 5(a), will typically contain metrics displaying overall enrollment and compliance on a site level. These can allow a clinical trial team to gauge the progress of a specific study easily, *i.e.*, the number of patients who have completed their time in the study and the number of patients still in progress.
- **Site-Level Compliance**: Generating reports based on sites, as seen in Figure 5(b), allows clinical teams to efficiently identify which sites may be experiencing issues regarding low compliance across their assigned patients. Typically, site reports contain information for overall performance, with specifics for patients that may fall below a set compliance threshold. The patients with low compliance are labeled with a potential issue- such as low compliance during the nighttime. The potential issues are derived from the hourly compliance for that patient. From here, sites can identify which of their patients contribute most to low compliance and attempt to resolve the issues linked to the low compliance.
- **Patient-Level Compliance**: Reports on a patient level

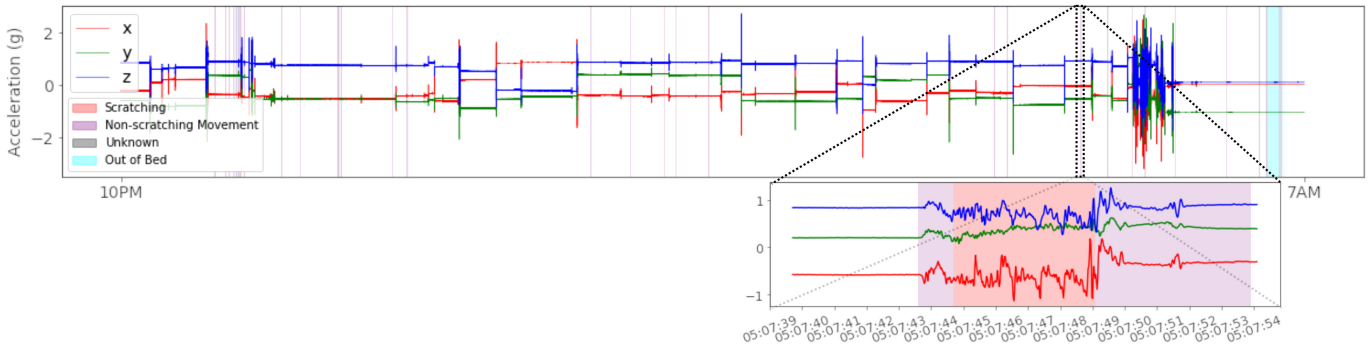


Figure 6: A plot of sensor signals overlaid with annotation labels is used to assess the data quality of annotations in conjunction with sensor signals.

can give insight into their specific patterns of device wearing. In these reports, as seen in Figure 5(c), the number of visits, compliant days within each visit, and compliance percentage per visit are displayed. In addition, an hourly compliance heatmap is visible, allowing for further understanding of when patients wear their devices across the study duration.

F. Data Quality in Novel Digital Endpoint Development

For novel digital endpoint development, raw sensor signals are collected along with annotations or labels, considered the ground truth. Annotations describe events explaining the status of the patient. As such, it is critical to assess the data quality of annotations and sensor signals to identify and address as many defects as possible.

Assessing Annotation Quality. Annotations are typically collected through patient reporting via a survey system or are labeled via software by trained clinicians who observe patient behavior. We first check for defects in the annotations. Defects may include improper data structure, invalid label categories, incomplete annotations, duplicates, and impossibly overlapping annotations. Defects could be caused by bugs in the annotation software or improper training on how to label.

Assessing Annotation Quality with Sensor Signals. Evaluating annotation quality in isolation is insufficient because digital endpoint development requires both annotations and raw sensor signals. So, we must also assess the data quality of annotations and raw sensor signals in conjunction. Therefore, we plot annotated time segments along with raw sensor signals (*e.g.*, Figure 6) to facilitate the data quality assessment.

Discrepancies in the alignment of annotations and raw sensor signals can vary considerably due to time tracking configurations and device properties in each step of the data collection process. Misalignment between annotation and raw sensor signals can be caused by improper device time configuration or the precision of the sensor device's initial time configuration. In addition, if the sensor device's time tracking is not periodically synced, the device's internal Real-time clock (RTC) will slowly drift over time. We measure drift using the sensor signal overlaid with annotation plots. Once the misalignment from the initial configuration time and RTC drift are measured, we align the raw sensor signals to the annotations.

After the annotations and sensor signals have been properly aligned, we observe the plots to identify possible defects in annotation quality. Defects could include improper labels, annotated events that are not apparent in the sensor signals, and time segments that appear to be missing annotations or sensor signals. Specific time segments of concern are selected and validated with the source to determine if further action is needed.

Lastly, depending on study-specific requirements, we may apply other methods to assess data quality. For example, output from movement detection algorithms can be compared to annotated time segments that describe the movement to check annotation validity and coverage. Using various methods to assess data quality from different approaches is essential to maintain the data quality needed for novel digital endpoint development.

Throughout a clinical trial, accessing data quality metrics is critical to upholding our outlined principles. Therefore, in addition to the compliance reports generated, an interactive data quality assessment tool is needed to monitor data quality throughout a trial. We are thus motivated to establish a data platform, *i.e.*, Magnol.Ai, that allows users to customize the plots to view digital data and the associated data quality reports through various lenses, utilizing filters and other user controls. For example, users may want to view the raw sensor data at the scale and resolution they desire, review derived compliance reports on a day, visit, or patient level. Figure 7(a) is such a typical screen image of the dashboard of Magnol.Ai where users can select the level and the metric for which the visualization will show accordingly. A user wants to view compliance for all patients in a study on the visit level, as seen in Figure 7(b). They define *compliance* as having at least 12 hours of data daily, with 3 days each visit comprising a compliant visit. By selecting the compliance type, which in this case is visit, and inputting the number of hours and days for defining compliance, the user can see the population's compliance report with these specific thresholds, as seen in Figure 7(a). Additionally, they can easily compare and contrast different levels and compliance thresholds within Magnol.Ai's dashboard.

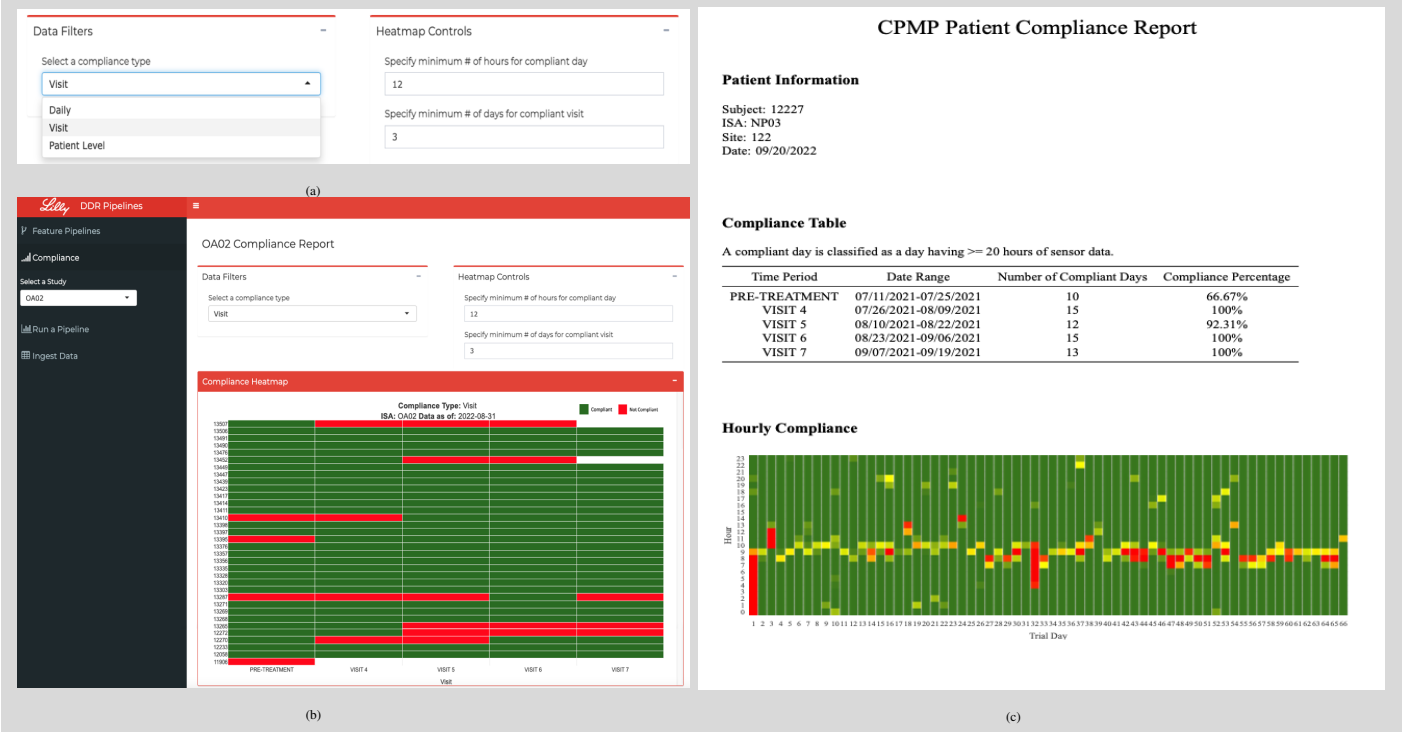


Figure 7: The platform features displaying (a) filters for customizable compliance reports, (b) compliance by visit, and (c) generated compliance reports on the patient level.

In addition to the compliance assessment, data quality visualizations, such as Figure 4, are created and customized within the platform. For example, as seen in Figure 7, a user can select a specific time range or time level to view the data. This zoom in and out can be used to identify and trace patterns of device wearing. The sensor data visualization, supplemented by the data quality assessment capability allows for customizable, real-time, informative visualizations that enable insights into patient compliance and device-wearing data patterns. The study team can process and act upon these key insights with these visualizations housed in a centralized, consistent, and efficient platform.

IV. Magnol.Ai — Putting Tools Together for Digital Biomarker Research

In this section, we focus on a few key building blocks of our data platform, *i.e.*, Magnol.Ai, which continuously ingests, visualizes, and profiles digital data for quality analysis and digital measure derivation. We introduce how we organize the various digital data sets collected from studies, and then how we present these digital data sets with interactive dashboards to help researchers navigate and explore these digital data, view the data quality reports and uncover data insights. Finally we focus on the technologies we leverage for cloudifing, versioning, and parallelizing Magnol.Ai's computing jobs for quality analysis and digital measure derivation.

A. Organizing Digital Data in Magnol.Ai

Organizing and presenting digital data from wearables is vital to the success of using digital technologies in a clinical study, and is a key consideration to regulators. Inside Magnol.Ai these wearable sensor data are stored and organized at study level for authorized visualization and access, with a few key attributes that can be leveraged to further subset or group the digital data in Magnol.Ai's data portal:

- **ISA**: There are cases where an overarching study consists of various unique studies, or a 'suite' of studies. In this case, each individual study is described as being an ISA. Using the aforementioned CPMP study suite, we see that there various ISAs. As seen in Figure 8, Magnol.Ai handles this case by treating each ISA as an individual study, under the umbrella of the overarching CPMP study.
- **Cohort**: A cohort is a grouping of participants in a given study that is specific to an activity or criteria as outlined in the study protocol. For example, in a drug trial, there can be a cohort for each specific drug dosage, as well as a placebo cohort. In an observational study, there can be cohorts that contain subjects who have varying levels of severity for a given disease state, as well as a healthy population cohort.
- **Participant**: Wearable sensor data are collected from participants enrolled in each clinical study. Displaying various sensor data collected from each participant as well as the derived features and data compliance is one typical way for one to explore sensor data using Magnol.Ai's

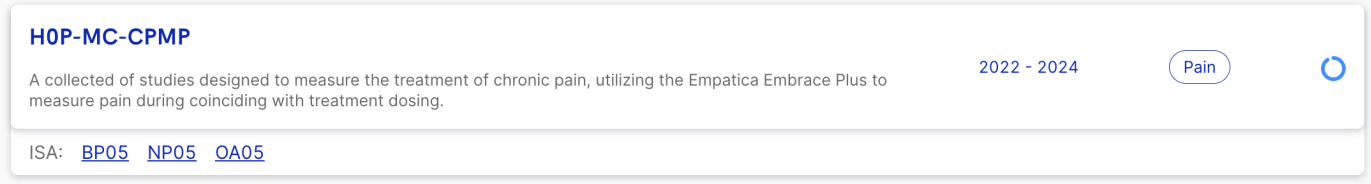


Figure 8: A typical "study suite" view in Magnol.Ai, where there are multiple ISAs, *i.e.*, individual studies, under this overarching "suite".

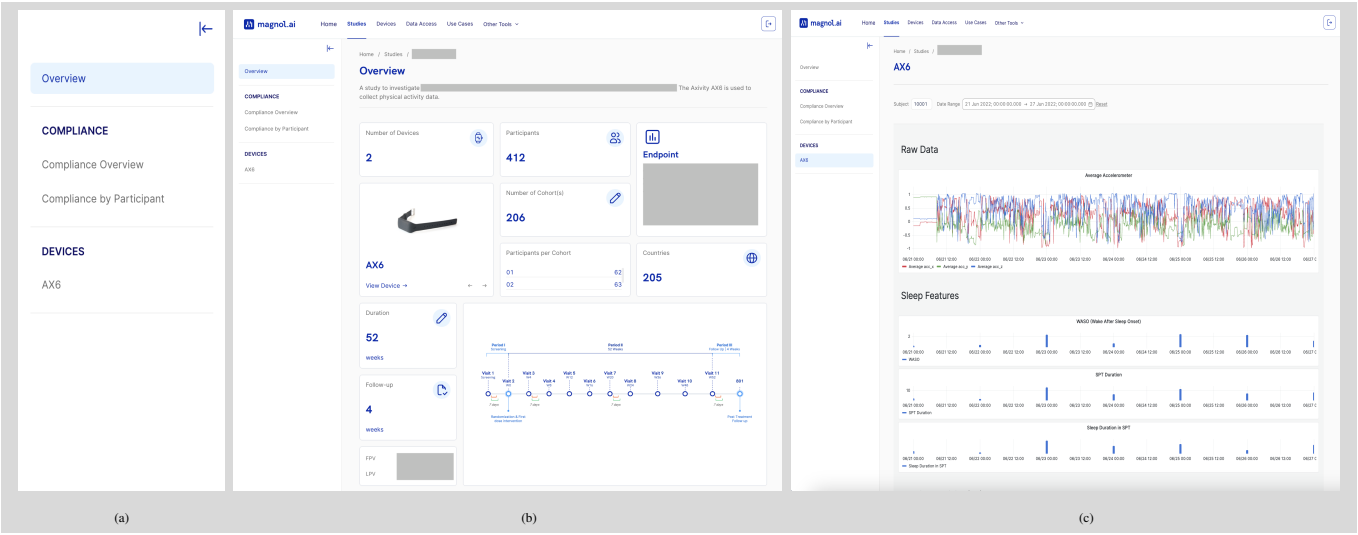


Figure 9: Navigating studies and exploring digital data sets with Magnol.Ai's dashboards. Key visuals: (a) side navigation panel, (b) study overview dashboard example, and (c) sensor data and derived features visualizations.

dashboard (see *e.g.*, Figure 9(c)).

- **Time Range**: Magnol.Ai allows one to view the full spectrum of all digital data collected in studies, or use controls to explore the data at the scale or resolution desired by the user.

By designing Magnol.Ai to work with various organizational hierarchies for study design, it can properly house and display the needed information, including an overview of a study, as seen in Figure 9(b). When beginning data exploration for a given study, users need to know the basic overview of study activities and study population. Magnol.Ai's custom design allows all study overview information to be viewed in one single dashboard. The study overview page contains high-level, universal metrics including the number of devices, endpoint descriptions, and study timeline.

B. Visualizing Wearable Sensor Data and Reporting Quality

In addition to the study overview page, each study contains various dashboards belonging to each type of data, including sensor data, ePRO data, and compliance data. Users can easily navigate to a given dashboard within a study directly from the study overview page. With each tab holding specific information, organizing the information from studies becomes as simple as making a new dashboard for each desired type of data. Taking compliance data as an example, a user would need to see an overview of compliance on a study level, as well as compliance on a participant level. By building dashboards representative of these two aggregation levels, these dashboards can be housed under 'Compliance'.

Compliance Overview The compliance overview dashboard shows aggregated levels of compliance at the study-level. These metrics account for all subjects, regardless of cohort, site, visit, or other grouping and often contain the following:

- *Number of Completed Participants* — The total number of participants who have completed the study.
- *Number of In-Progress Participants* — The total number of participants who are currently in the study.
- *Average Daily Compliance* — The average percentage of daily device wearing.
- *Average Daily Wearing Hours* — The average number of hours where a device was worn.

As shown in Figure 7, various visualizations showing compliance including daily, weekly, or visit heatmaps, distribution plots of average daily wearing compliances, and compliance metrics by site are often included.

Compliance at Participant Level The compliance analysis at participant level displays compliance metrics and detail plots corresponding to each participant with more granularity —- hour by hour throughout the entire trial. Additional filters can be applied including visit information or trial day information. While some kind metrics, such as average daily compliance percentage, are also used on the compliance overview dashboard, visualizations on the 'Compliance by Participant' dashboard

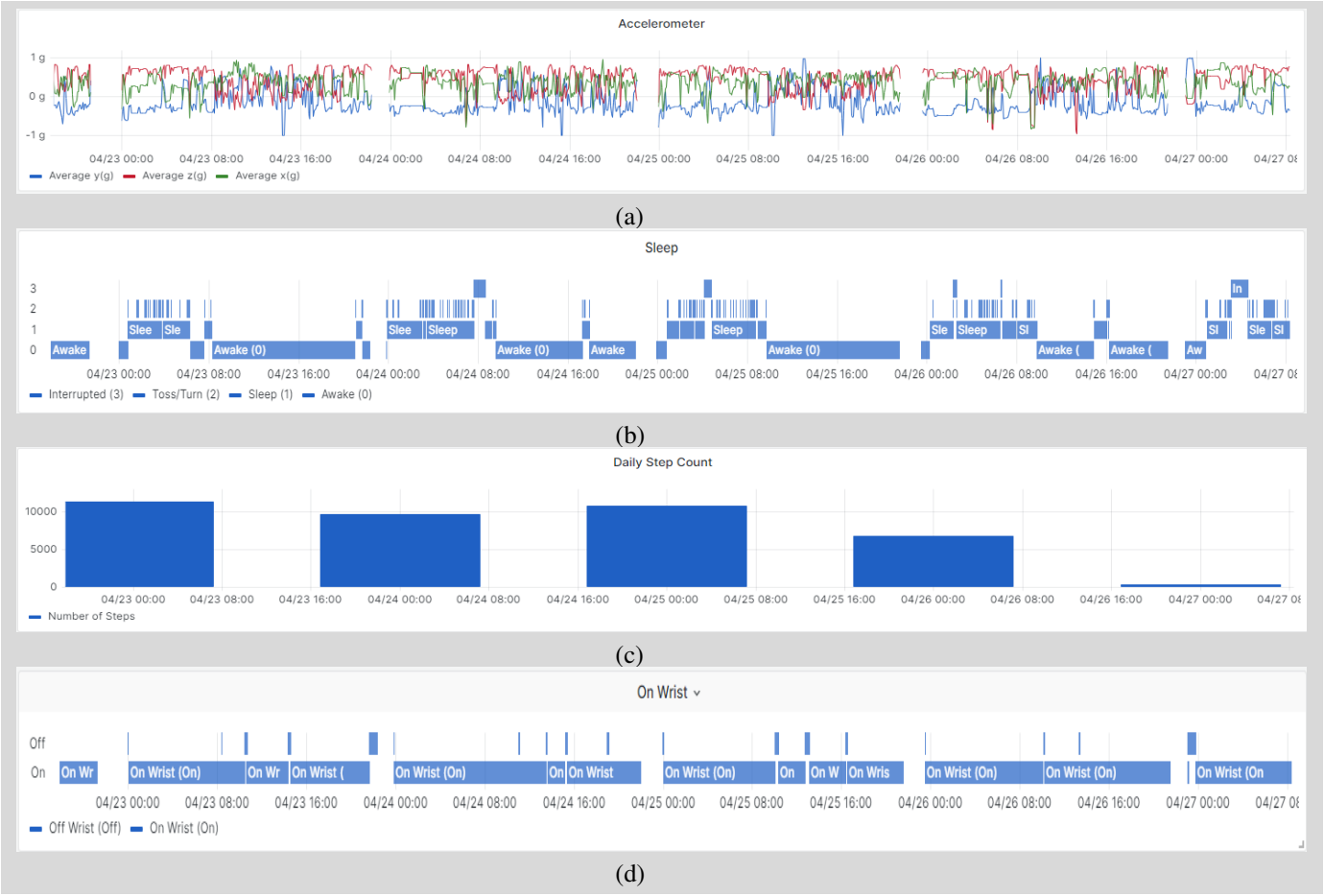


Figure 10: Magnol.Ai offers the power to visualize and directly compare raw sensor signals (a), derived features — sleep measure (b), step count (c), and daily wearing compliance (d) to fully understand the quality of the data.

can show more detailed information about a subject including:

- *Visit Compliance* — With the addition of a visit selection dropdown, users can select a visit and view compliance for each hour during a given visit in a study (Figure 7(b)).
- *Hourly Compliance* — A heatmap showing the number of wearing minutes for each hour across the duration of a study, offering more insights to wearing patterns of a given subject (Figure 7(c)).

Sensor Data Visualization Sensor data can also be organized and displayed similarly to compliance information (see Figure 9(c).) For each device in a study, there is a designated dashboard under the 'Devices' section in the side navigation panel, as seen in Figure 9(a). When navigating to a given dashboard, users can see visualizations of raw sensor data processed sensor data, and the derived features. Across types of visualizations, the data is synchronized to zoom / aggregate to the same level in the same view. This can be accomplished by 1) clicking and dragging to zoom into a specific time range (see *e.g.*, Figure 6), or 2) utilizing the date-picker at the top of the page to select a date on a calendar, or type in a specific date and time (see *e.g.*, Figure 9(c)).

Data Quality The true power of Magnol.Ai comes in the form of visualizing data quality. With the capability to view various types of data aligned to the same date and time range, users can easily identify the story that the data is telling. Examining Figure 10, we see the raw accelerometer data coming from a wrist-worn device. We can compare the raw sensor signals to the derived sleep minute features. Additionally, we can view the device wearing compliance for each day the device was worn, in form of a heatmap. With all three data visualization channels aligned to the same date and time range, users easily view the quality of the data and assess which segments of the data are most useful to analysis.

We have designed Magnol.Ai to view data in this way in order to directly compare data and fully understand the quality of the data. This capability is not limited to only viewing sensor data, but rather, we can apply the same methodology to comparing patient reported events to raw sensor data and derived features, as well as ePRO data, including pain ratings. With the capability to compare virtually any type of data, Magnol.Ai allows unlimited exploration, including data coverage, quality, and compliance.

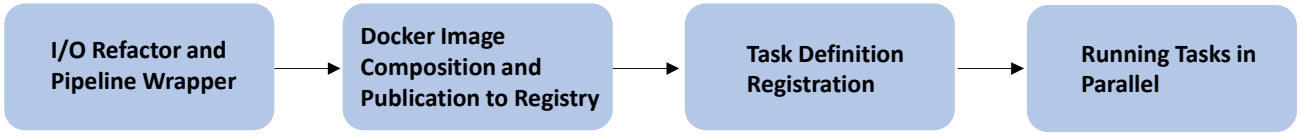


Figure 11: Flowchart of Pipeline Cloudifying Process

C. Cloudifying, Versioning, and Parallelizing Data Pipelines

Motivation Our data processing and digital measure derivation pipelines previously run on company's on-premises High Performance Computing (HPC) systems. As our computational needs grow significantly following limitations are identified:

- *Limited Scalability* — On-premises HPC systems often have fixed hardware resources, which can limit the scalability of data processing pipelines. As data sets grow in size or processing demands increase, it may become challenging to accommodate the additional workload efficiently. This can lead to performance bottlenecks and longer processing times.
- *Dependency Management* — Running pipelines on dedicated on-premises systems might involve managing various dependencies, libraries, and runtime environments manually. Ensuring consistent environments across different systems can be challenging and may lead to compatibility issues and version conflicts.
- *Lack of Elasticity* — On-premises HPC systems have fixed capacity, which means they cannot easily adapt to varying workloads. During periods of low activity, resources may be underutilized, wasting valuable computing power. Conversely, during periods of high demand, the system may struggle to handle the load efficiently, leading to delays in data processing.
- *Reproducibility Challenges* — Reproducing pipeline results can be challenging on on-premises HPC systems due to potential differences in hardware, configurations, and software versions. This lack of standardization can hinder the ability to validate and replicate research findings reliably.
- *High Maintenance Costs* — Maintaining and upgrading on-premises HPC infrastructure can be expensive, requiring significant capital investments and ongoing operational costs. Additionally, specialized personnel are needed to manage and support the infrastructure, which can add to the overall expenditure.

In contrast, containerizing pipelines and migrating to cloud-based environments like AWS using Docker addresses many of these drawbacks. Containerization is a lightweight and portable approach to software development and deployment. It is a method of packaging an application along with all its dependencies, libraries, and configurations into a single unit called a container. This container acts as a self-contained execution environment, allowing the application to run consistently and predictably on any platform that supports the containerization technology. Unlike traditional virtual machines, which require a full operating system for each application, containers share the host OS kernel, making them much more efficient and lightweight. This efficiency not only reduces resource overhead but also facilitates rapid deployment and scaling, making containerization an ideal solution for modern cloud-based architectures.

Cloud-based containerization offers elasticity, scalability, and cost-effectiveness, allowing researchers to efficiently process large datasets, optimize resource utilization, and adapt to varying workloads. Docker's containerization approach ensures consistent environments, simplifies dependency management, and enhances reproducibility, making it easier to validate research findings and collaborate with others seamlessly. Additionally, cloud providers handle infrastructure maintenance, disaster recovery, and offer a pay-as-you-go model, reducing the need for extensive upfront investments and ongoing maintenance costs. In below we detail the advantages of containerization.

Cloudifying Public cloud providers, such as AWS, Google Cloud Platform (GCP), and Microsoft Azure, offer robust support for container-based parallel and distributed processing services. AWS provides Elastic Container Service (ECS) and Elastic Kubernetes Service (EKS), GCP offers Google Kubernetes Engine (GKE), and Azure has Azure Kubernetes Service (AKS) and Azure Container Apps. These cloud-native services enable seamless deployment and management of containerized applications at scale, aligning perfectly with our needs for parallel data processing.

Migrating from on-premises high-performance computing to the cloud aligns with our company's strategy to leverage cloud-based storage and computation. By adopting container-based processing in the cloud, we can take advantage of the cloud's elasticity, scalability, and cost-effectiveness. Cloud infrastructure allows us to dynamically allocate resources based on demand, optimizing utilization and reducing operational costs. Additionally, it reduces the burden of managing and maintaining on-premises hardware, providing us with more flexibility and agility to adapt to changing research requirements and workloads.

Versioning Ensuring the reproducibility of results is crucial in scientific research and data processing. Dockerizing our pipelines allows us to version both the algorithm source code and the entire environment in which the processing takes place. This means that we can track changes to the pipeline code over time, allowing us to roll back to previous versions if needed. Moreover, capturing the entire environment, including

libraries, configurations, and runtimes, guarantees that the data processing pipeline will produce consistent and replicable results, regardless of the underlying infrastructure or platform.

Furthermore, provenance tracking, which involves recording the origin and history of data and processes, is essential for maintaining data integrity and traceability. Docker containers act as self-contained units that encapsulate all the dependencies and configurations required for data processing. By versioning the container images, we can precisely track the exact environment in which data processing occurred, ensuring that any future analysis or audits can be confidently performed based on the same conditions. This level of versioning and provenance tracking enhances the credibility of our research and allows other researchers to reproduce our findings with ease.

Parallelization The data processing pattern for digital measures, such as step count and sleep duration, often follows an embarrassingly parallel computing paradigm. This means that the processing of data for each subject day can be performed independently, without any need for inter-process communication or coordination.

By dockerizing these pipelines and running them in a public cloud environment like AWS, we can easily take advantage of the cloud's ability to handle data processing in parallel and at scale, enabling faster and more efficient analysis of large datasets. To be more specific, docker containerization provides a straightforward way to manage parallel computing tasks efficiently. With the ability to spawn multiple containers simultaneously, we can distribute the data processing workload across a cluster of containers, enabling concurrent execution of tasks. This scalability not only speeds up the overall data processing but also ensures that we can handle large volumes of data without overwhelming the system. Furthermore, container orchestration tools like Kubernetes make it easy to manage the deployment, scaling, and monitoring of containers, simplifying the management of parallel processing in the cloud environment.

Additionally, the elasticity of cloud resources ensures that we can dynamically adjust the number of container instances based on the workload, optimizing resource utilization and cost efficiency.

Pipeline Cloudifying Steps

Figure 11 summarizes the steps of cloudifying a traditional pipeline and below we detail these steps.

- *I/O Refactor and Pipeline Wrapper* — During the process of containerizing the pipeline, a significant aspect that required attention was the handling of file input/output, which originally relied on local storage. To make the pipeline cloud-ready and to ensure seamless data processing in a distributed environment, a crucial refactoring step was undertaken to implement an I/O layer that interacts with cloud storage, specifically an S3 bucket.

  During the I/O refactoring phase, a thoughtful design approach was adopted to seamlessly integrate cloud-based file input/output functionalities without modifying the original pipeline code. The design centered around the creation of a wrapper that serves as an intermediary layer between the pipeline and the cloud I/O services, specifically the S3 bucket. This wrapper encapsulates the necessary code to interact with the cloud storage, enabling the pipeline to leverage the benefits of cloud-based file management while maintaining the integrity of its core functionality. By isolating the cloud I/O layer in the wrapper, the underlying pipeline code remains untouched and agnostic to the storage medium. This design preserves the pipeline's portability and allows it to be run with minimal modifications in diverse computing environments.

  By leveraging Amazon S3's object storage service, the pipeline can now read input data from and write output data to the S3 bucket, providing a scalable and durable storage solution. The refactored I/O layer ensures that the pipeline can effectively handle cloud storage, making it well-suited for deployment in public cloud environments like AWS.

- *Docker Image Building* —

  The Dockerfile presented in Listing 1 serves as a blueprint for building a containerized environment tailored to process sensor data to derive digital measures using GGIR. It starts by utilizing the base image of R, upon which subsequent instructions are layered to set up the required configuration. For instance, setting up GGIR pipeline in R (line 5), installing dependent python libraries required by our driver (line 14). Lastly, the Dockerfile defines the command to run the application (driver 'ggir.py' in our case) upon container launch (line 20). Through this Dockerfile, researchers can create a self-contained environment with all necessary components, enabling seamless and consistent data processing across various platforms and environments. Once Docker image being fully validated, we deposit it to AWS Elastic Container Registry (ECR).

```
1  # Base image for R
2  FROM rocker/r-ver:4.3.1
3
4  # Install GGIR version 2.8-6
5  RUN R -e
       "devtools::install_github('wadpac/GGIR@2.8-6')"
6
7  # Install Python 3.9
8  RUN apt-get -y install python3.9
9
10 WORKDIR /app
11
12 # Install dependencies through requirements.txt
13 COPY requirements.txt .
14 RUN pip install --no-cache-dir -r requirements.txt
15
16 # Copy Python driver program and other files
17 COPY . .
18
19 # Set the entry point as the Python driver
20 ENTRYPOINT ["python", "ggir.py"]
```

Listing 1: Sample GGIR Dockerfile.

- *Task Definition Registration* —

```
family="pipeline-ggir"
task_role_arn="ecs-task"
execution_role_arn="dh-tti-ecs-exec"
# CPU units, 1,024 CPU units per vCPU
cpu=2048
# in MiB
memory=8192
container_name="ggir"
image_uri="dhrd-pipeline-ggir:2.8.6"

response=$(aws ecs register-task-definition \
  --family "${family}" \
  --task-role-arn "${task_role_arn}" \
  --execution-role-arn "${execution_role_arn}" \
  --network-mode awsvpc \
  --requires-compatibilities FARGATE \
  --cpu ${cpu} \
  --memory ${memory} \
  --container-definitions "[{
      "name": "${container_name}",
      "image": "${image_uri}",
      "essential": true,
      "cpu": ${cpu},
      "memory": ${memory} }]" \
  )
```

Listing 2: Sample script to register GGIR task definition.

In the provided sample shell script as shown in Listing 2, the task definition registration is a crucial step in deploying containerized applications on AWS. ECS task definitions define the configuration for individual tasks, specifying essential parameters such as the Docker container image, CPU and memory requirements, network settings, and task roles. By registering a task definition using the AWS CLI (Command Line Interface), we effectively create a blueprint for the containerized application that ECS can use to launch and manage instances of the task. The registration process associates the task definition with a specific family name, such as ”pipeline-ggir” in this case, making it easily identifiable and reusable across ECS services.

In addition, task definitions support the concept of revisions. A revision represents a specific version or iteration of a task definition. Whenever a task definition is updated, either to change container configurations, environment variables, or other parameters, a new revision is created. This approach allows ECS to maintain a historical record of changes to the task definition over time. This helps to maintain a history of configuration changes, which is useful for auditing, rollback purposes, and understanding the evolution of our pipelines over time.

- *Running Tasks in Parallel* — Following the successful registration of the ECS task definition, the next crucial step in our containerized data processing pipeline is to launch ECS tasks for each input/output file pair, as shown in Listing 3. The script employs a for loop to iterate over the lists of input and output file paths. For each pair, a separate ECS task is initiated using the AWS CLI's aws ecs run-task command. The run-task API call submits tasks in an asynchronous fashion, enabling ECS to efficiently run tasks in parallel based on the available resources in the designated cluster, in this case, “dbm-pipeline”. The tasks are executed using the specified Fargate launch type, which manages the underlying infrastructure, allowing us to focus solely on defining the task requirements in the task definition. The ‘–overrides’ flag within the script allows us to pass dynamic input and output directories to the Docker container as command-line arguments. This enables the containerized application to process the corresponding input data from S3 and store the results in the specified output location. By launching tasks in parallel, ECS leverages the underlying cloud infrastructure's scalability, making it a well-suited solution for processing large-scale datasets in an efficient and resource-effective manner.

```
# List of input file paths
input_folders=("s3://...", ...)

# List of output file paths
output_folders=("s3://...", ...)

cluster_name="dbm-pipeline"
task_definition_name="pipeline-ggir"

# Iterate over the input/output lists
# and run the Docker container for each pair
for i in "${!input_folders[@]}"; do
  input="${input_folders[i]}"
  output="${output_folders[i]}"

  # Run the Docker container as a separate ECS task
  aws ecs run-task \
    --cluster $cluster_name \
    --launch-type FARGATE \
    --task-definition $task_definition_name \
    --overrides "{"containerOverrides":
        [{
        "name": "${container_name}",
        "command": [
        "--inputdir", "${input}",
        "--outputdir", "${output}" ]}]}"
done
```

Listing 3: Sample script to run parallel GGIR tasks.

V. Conclusion and Future Work

As DHT continue to evolve and collect more complex digital data in clinical trials, the need for a digital data quality assessment platform is increasing. By defining and implementing the fundamentals of data quality into the digital data quality framework and platform, we can generate automated compliance reports, customizable visualizations, and real-time quality metrics. In addition, the methods for facilitating dBMs research have been simplified with the centralized digital data quality assessment platform. As dBMs research continues, so will the use of the digital data quality assessment platform. Future directions include the use of visual mining and data mining technologies to help identify data quality in a novel way to facilitate data quality assessment.

References

[1] H. Zhang, R. Giesting, L. Miller, G. Ruan, N. Patel, J. Ji, T. Zhang, and Y.-L. Yang, “A framework for digital data quality assessment in digital biomarker research,” in *The Ninth International Conference on Big Data, Small Data, Linked Data and Open Data (ALLDATA 2023)*, 2023.

[2] J. Wright, O. Regele, L. Kourtis, S. Pszenny, R. Sirkar, and C. Kovalchick, “Evolution of the digital biomarker ecosystem,” *Digital Medicine*, vol. 3, no. 4, pp. 154–163, 2017.

[3] R. Y. Wang, V. C. Storey, and C. P. Firth, “A framework for analysis of data quality research,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 7, no. 4, pp. 623–640, 1995.

[4] A. Sharma, R. A. Harrington, M. B. McClellan, M. P. Turakhia, Z. J. Eapen *et al.*, "Using digital health technology to better generate evidence and deliver evidence-based care," *Journal of the American College of Cardiology*, vol. 71, no. 23, pp. 2680–2690, 2018.

[5] R. Lyons, G. R. Low, C. B. Congdon, M. Ceruolo, M. Ballesteros, S. Cambria, and P. DePetrillo, "Towards an extensible ontology for streaming sensor data for clinical trials," in *Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*, 2021, pp. 1–6.

[6] C. M. Rey, "Wearable data revolution: Digital biomarkers are transforming research, promising a revolution in healthcare," *Clinical OMICs*, vol. 6, no. 2, pp. 10–13, 2019.

[7] I. Clay, "The future of digital health," *Digital Biomarkers*, vol. 4, no. 1, pp. 1–2, 2020.

[8] M. Chen and M. Decary, "Artificial intelligence in healthcare: An essential guide for health leaders," in *Healthcare management forum*, vol. 33, no. 1. SAGE Publications Sage CA: Los Angeles, CA, 2020, pp. 10–18.

[9] S. M. Hossain, T. Hnat, N. Saleheen, N. J. Nasrin, J. Noor, B.-J. Ho, T. Condie, M. Srivastava, and S. Kumar, "Mcerebrum: A mobile sensing software platform for development and validation of digital biomarkers and interventions," ser. SenSys '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 1–14.

[10] A. Dillenseger, M. L. Weidemann, K. Trentzsch, H. Inojosa, R. Haase, D. Schriefer, I. Voigt, M. Scholz, K. Akgün, and T. Ziemssen, "Digital biomarkers in multiple sclerosis," *Brain Sciences*, vol. 11, no. 11, pp. 1519–1544, 2021.

[11] M. M. Rahman, V. Nathan, E. Nemati, K. Vatanparvar, M. Ahmed, and J. Kuang, "Towards reliable data collection and annotation to extract pulmonary digital biomarkers using mobile sensors," in *Proceedings of the 13th EAI International Conference on Pervasive Computing Technologies for Healthcare*, 2019, pp. 179–188.

[12] A. Doherty, D. Jackson, N. Hammerla, T. Plötz, P. Olivier, M. H. Granat, T. White, V. T. Van Hees, M. I. Trenell, C. G. Owen *et al.*, "Large scale population assessment of physical activity using wrist worn accelerometers: the uk biobank study," *PloS one*, vol. 12, no. 2, p. e0169649, 2017.

[13] C. Sudlow, J. Gallacher, N. Allen, V. Beral, P. Burton, J. Danesh, P. Downey, P. Elliott, J. Green, M. Landray *et al.*, "Uk biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age," *PLoS medicine*, vol. 12, no. 3, p. e1001779, 2015.

A Framework for Developing and Evaluating Modular Mobility Aids for People with Visual Impairment: An Indoor Navigation Use Case

Florian von Zabiensky, Grigory Fridman, Sebastian Reuter, Oguz Özdemir, Michael Kreutzer, Diethelm Bienhaus
Institute of Technology and Computer Science
Technische Hochschule Mittelhessen
University of Applied Sciences
Gießen, Germany
{florian.von.zabiensky, grigory.fridman, oguz.oezdemir, sebastian.reuter, michael.kreutzer, diethelm.bienhaus}@mni.thm.de

***Abstract*—Electronic Travel Aids are devices that help people with visual impairments navigate and orient themselves. The development of such devices is often associated with a loss of time in repetitive work, resulting in slow progress in this field. A collaborative community that shares its expertise could accelerate this progress and lead to truly useful and market-ready products for visually impaired people. To make such an exchange efficient, a standardized, component-based ecosystem is required. So far, such an approach for Electronic Travel Aids has not been pursued in the literature and is therefore addressed in this paper. To this end, a model for identifying component boundaries is presented and illustrated by a project in the form of an ultra-wideband indoor navigation system. The advantages of such a component-based development in general are described. In particular, the use of the Robot Operating System 2 (ROS 2) for the implementation is highlighted and its suitability for such an ecosystem is discussed based on practical experience with it. The evaluation of such an ETA with the use of the ROS 2 ecosystem and a component-based ETA are also highlighted. The contribution of this work is a framework that reduces the effort for the development and evaluation of electronic travel aids and allows an early involvement of users in the development process.**

***Index Terms*—ETA; electronic travel aid; mobility aid; ROS 2; ROS; robot operating system; component-based development; user-centered design**

I. Introduction

This paper was originally presented at the SMART ACCESSABILITY 2023 conference with a focus on developing electronic travel aids (ETAs) [1]. For the journal article it was extended by a user evaluation of the described ETA to show the possibilities of the presented framework beyond the development and the advantages it offers for user-centered development.

In the field of ETA an active research takes place. ETAs are devices that help people who are visual impaired or blind in travelling tasks like orientation, navigation, and obstacle avoidance. Several overview papers summarize relevant and representative research [2]–[5]. It is important to continually expand the possibilities of these aids and thus increase the mobility of those people.

However, when looking at the systematic literature review of Khan et al., one can find tables containing groups of papers classified by the technology or hardware components used [2]. If the projects behind those papers rely on the same technology or hardware, the efficiency of their development could be increased by sharing common source code or libraries. This would increase the time that can be spent, e.g., on designing user interfaces. For example, if one researches a novel approach to the acoustic representation of obstacles, he can focus on his research, knowing there is a pool of hardware components, simulation methods and algorithms for locating obstacles in the environment. In this way, all human resources can be concentrated on the novel representation to achieve faster and better results.

This example is only possible, if the development of ETAs is based on common principles. For this purpose, we divide ETAs into interchangeable components and propose a framework to develop ETA components that can be shared. Such a development is presented with an indoor navigation system as an example.

The experiences made during the development of this system, as well as the development itself, are finally discussed to openly present the positive and negative sides and to justify a recommendation for such a development.

This paper is structured as follows. First, it presents several literature reviews that summarize and classify numerous concrete projects from ETA research in section II. Section III discusses the problems identified in the research. As a solution proposal, section IV describes a specialized Human Machine Interface (HMI) for ETAs to divide ETAs into their components, as well as the Robot Operating System 2 (ROS 2) as a software development kit. To discuss this solution, section V applies it to an indoor navigation system as an example for development and evaluation. Section VI discusses the practical experience of the development, with its advantages and disadvantages. Section VII concludes the results and gives an outlook to further work needed to be done.

II. Related Work

With a focus on ETAs, there are several development projects in the field of navigation systems for blind and partially sighted people. These projects use different approaches to help these individuals navigate safely in both indoor and outdoor environments. With the digital transformation of healthcare, Internet of Things devices can enhance the capabilities that can be achieved in this area. Khan et al. [2] conducted a systematic literature review to analyse the challenges and opportunities of such 'smart navigation devices' that have been researched and developed over the last decade. Using structured selection criteria, the review identified 191 relevant articles published between 2011 and 2020 in six different peer-reviewed digital libraries.

Khan et al. [2] categorized various approaches to navigation systems for blind and visually impaired individuals into three parts. The study provides a comprehensive list of commonly used systems, tools, and hardware components as examples.

1) Approaches reported for navigation system development e.g.:
   - Indoor navigation system
   - Mobile application
   - Wearable navigation systems with e.g., smart watches
2) Technologies/tools proposed for navigation assistant development e.g.:
   - Raspberry Pi microcomputer
   - Android-based applications
   - Microcontroller
3) Hardware components proposed for obstacle avoidance e.g.:
   - Bluetooth beacons
   - Haptic devices
   - Ultrasonic sensors
   - Global Positioning System (GPS)

It is evident that various projects developed in these fields share similarities in terms of system level, technology, and hardware components used, indicating that multiple development efforts can result in similar or identical solutions. This indication is present in most of the recent survey papers in the field of ETAs [2]–[5]. A similar situation in the field of robotics was part of the driver for the Robot Operating System [8].

III. Problem Statement

In 2007, then PhD students Eric Berger and Keenan Wyrobek discovered a fundamental problem in robotics research. A pattern was emerging, in which researchers wanted to build on a proof-of-concept presented in a paper to implement their own idea. Either they lack details of the software used, or it is unusable for whatever reason, so they are often forced to spend 90 percent of their time rewriting other people's code and developing their own prototype test-bed. This leaves only the remaining 10 percent to develop their own innovation, which then lacks quality but enables the intended publication. This creates a cycle of reinventing the wheel and wasting a huge amount of time. This led to the idea of creating a kind of Linux for robotics with the Robot Operating System (ROS), containing a common set of software and developer tools that would allow roboticists to build innovative ideas on the successes of others [9].

Looking at the numerous projects that have already emerged in the field of ETAs [2], one discovers this problem pattern again in many respects. In particular, the description of the selection process of the literature to be evaluated shows that many projects are similar and only a few add value to the state of the art. Additionally, these findings are rarely translated into products that benefit the end user. One reason for this might be that the projects are usually developed from scratch and thus valuable resources are lost to be put into the actual core of the work. For example, the categorization in [2] of some of the ETA prototypes known from research according to their hardware components makes it easy to see that many projects use similar, if not the same, subsystems and devices. The same concepts and technologies are being used for similar, if not identical, tasks. This leaves little time for iterative improvements and testing with visually impaired people. To counteract this, this paper presents a component-based development that contributes to the exchange between working groups and thus to a faster and more efficient prototype cycle. To this end, we built upon ROS 2, the successor of the above-mentioned ROS, which also serves as a motivator. In the field of mobile robotics, ROS 2 has helped components to be exchanged and to communicate with each other in a uniform manner, so that individual working groups can work much more efficiently on their research problems. In this paper, the development of an indoor navigation system using a vibration vest as an output device is presented. This project is not put in focus because there are other projects with similar results. The focus of this paper is on how the development can be made more efficient, and this will be shown and evaluated using the indoor navigation example.

IV. Solution Approach

If we look at the model of a human-machine interface in a very abstract way, it can be broken down according to Kantowitz and Sorkin [10] into the subcomponents shown in Fig. 1. A person (left) has the ability to acquire information through the senses available to him or her. This information is processed in the brain to make decisions based on it, such as operating the machine (right). The control components provided by the machine for this purpose have an influence on the internal state of the machine, from which outputs are generated to present information to the person. The two transitions between the human side and the machine side are called *human-machine interfaces*.

This model can also be used in an extended form to describe a visually impaired person and his mobility aid, where the physical environment is added as a crucial component (see Fig. 2). The ETA itself takes on the role of the machine by sensing relevant information about the environment (e.g.,

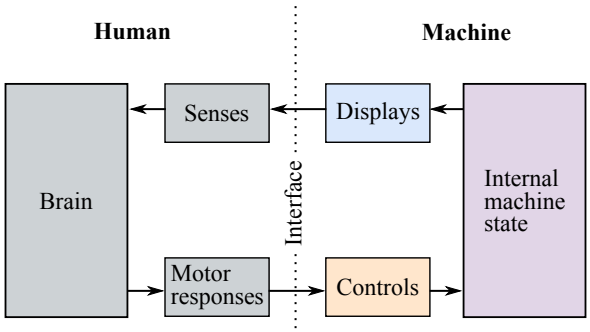


Fig. 1. Human-machine-interface model according to Kantowitz and Sorkin [10]

obstacles) and making it available to the person through an accessible information channel. The person's sensory system (the senses and perceptions available to him or her) takes in both this information and the information perceived directly from the environment (e.g., a car horn) and uses it to construct a *mental environment model* in the brain. The person can then influence the environment through their motor skills and, by interacting with the mobility aid, control its machine state and the *digital environment model* based on it. Other external information channels (e.g., online weather services) can also be used to enrich this digital model.

Looking into ETAs, the following system components, which are directly linked to the internal machine state, can be identified:

1) *Sensors*: Used to gather information from the immediate environment to build up an internal system state.
2) *Controls*: Used to directly control the assistive device without having to go through the environment.
3) *Additional Information Sources*: Sources of information not associated with the system itself, but which contribute to the construction of the internal state of the system.
4) *Displays*: Used to present information to the user, for visually impaired people to substitute the sense of sight, usually in acoustic or haptic form.

An overall system thus represents a composition of concrete instances of these components and a kind of business logic that receives information from *Sensors*, *Controls* and *Additional Information Sources*, converts it into a digital environment model thanks to certain algorithms, and provides a representation of it via *Displays*.

By defining good and consistent interfaces for individual component types, there are two advantages to such a component-based view. On the one hand, you can achieve easy *interchangeability* of individual components without having to adapt much to the overall system. Consider, for example, a navigation system that uses GPS to determine the current position of the user. Developing the same navigation system, but using RFID technology, would now require very little overall effort with a common interface, since only the sensors component would need to be changed. On the other hand, *reusability* increases with different overall systems that use the same subcomponents. As an example, consider an obstacle detection system and a navigation system, both of which use a vibration belt as a display component. The former uses it to signal obstacles in a particular direction, and the latter to indicate the direction of travel. If developed within a component-based framework, it would only be necessary to determine the obstacle or walking direction from the digital environment model, but not to redevelop the vibration belt as a component.

When developing mobility aids, avoiding collisions with obstacles, following certain navigation routes or, more generally, minimizing dangerous situations play a crucial role. However, the testing of such dangerous situations is essential for the evaluation of the developed prototypes, which is why a *simulation environment* has great advantages in the development of ETAs. On the one hand, it increases reproducibility by allowing test persons to be led through the same scenarios and their behaviour to be recorded and statistically evaluated. It also increases variability, as a simulation environment can be freely parameterized and configured to meet a wide range of system requirements. For example, weather conditions, which often strongly influence the behaviour of a sensor- or camera-based ETA, can be changed with little effort. It is also possible to generate custom obstacles, roads, traffic situations, etc. Such variability is difficult to achieve in the real world. In addition, the dangerous situations mentioned above can be mitigated, as real collisions are impossible or can be provoked for testing purposes in a controlled environment.

Considering that individual components are to be used in a simulation environment with little effort, it makes sense to embed this environment in the model shown in Fig. 2. In principle, any of the components on the ETA side can be simulated, the most obvious being the physical environment and the sensors. The former is a *virtual reality* in the simulation, which requires it to be sensed by *virtual sensors*. Since such sensors can provide perfect, noise-free environmental data, it is possible to test displays, controls and the algorithm used to build the internal state of the machine individually and in a controlled manner. In the indoor navigation system presented in the next chapters, this is demonstrated in more detail using an example.

Looking at past research projects on ETAs, one can see the presented component-based structure in many of these overall systems, mentioned by Khan et al. in their literature review [2]. Often the boundaries between the individual components become blurred because they are very closely related, but the basic structure remains the same. This suggests that, again, components could be easily exchanged and reused in similar systems if they were developed within a standardized, common ecosystem.

One such component-based ecosystem is ROS 2, which is a set of software libraries and tools for developing applications that originated in robotics (especially mobile robotics). It is open source and aims to support developers from different industries from research to prototyping, deployment and

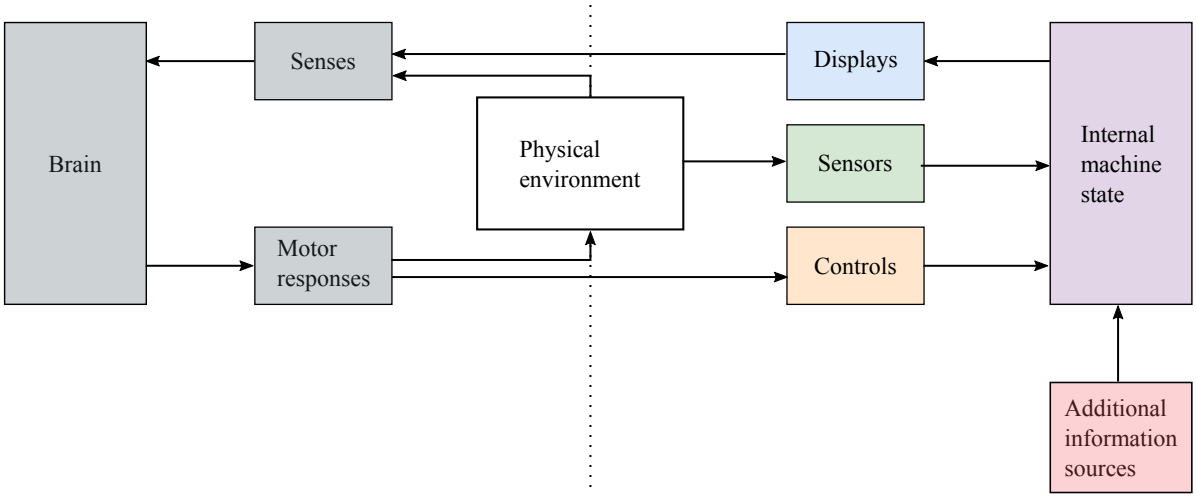


Fig. 2. Extended HMI model for ETAs

production using a standard software platform. The modular and flexible architecture allows easy integration of different hardware and software components, enabling the development of complex overall systems. A standardized real-time capable communication protocol enables efficient and reliable communication between different subcomponents of a system. It is not tied to a specific platform, nor is it domain or vendor specific. Because of its origins in mobile robotics, it provides many algorithms and sensor drivers to address problems of environmental perception, navigation and orientation, problems that are also common in the field of mobility aids. ROS 2 simplifies the development and testing of complex systems by providing debugging, visualization and, above all, simulation tools.

In ROS 2, development is strictly based on the "divide and conquer" principle by providing the following architectural components [6]:

1) *Nodes*: Independent processes that communicate with each other through different mechanisms.
2) *Topics*: Named event channels that allow nodes to communicate with each other. Nodes can publish messages to a topic, and other nodes can subscribe to that topic to receive those messages. Topics can have multiple publishers and subscribers, making it possible to build complex communication patterns between nodes.
3) *Services*: Remote procedure calls that allow nodes to request a specific task or information from another node in a synchronous way. Nodes that provide services and respond to requests are called servers, while nodes that request services are called clients.
4) *Parameters*: Parameters are used to store configuration data for nodes. Parameters can be set and retrieved by nodes, and they can be changed dynamically during runtime.
5) *Launch files*: Used to simplify the process of starting and configuring a ROS 2 system by specifying a collection of ROS 2 nodes, their parameters and other configuration details without having to start each node individually and configure it manually.
6) *Packages*: Collection of nodes, configuration and launch files and documentation, representing a subcomponent of a ROS 2 system. They provide a modular and extensible way to organize and distribute code, making it easier for developers to share and reuse code across different projects.

This architecture divides a system into a set of intercommunicating nodes, which are in turn organized into packages, providing a modular and extensible way to organize and distribute code, making it easier for developers to share and reuse code across projects. Beneath others, defining standard interfaces and the component-based development made it possible to build up a large and active community that constantly extends ROS's vast array of code libraries, hardware drivers, documentation and support. The community supports a continuous exchange between scientists and developers of new products.

V. Proof of concept

Our proof of concept represents an indoor navigation system developed specifically for blind and visually impaired individuals.

A. Indoor navigation system

The following components were used in this particular use case.

- *bHapticsX40 vibration vest by bhaptics®* to provide haptic feedback for navigation instructions.
- *An ultra-wideband (UWB) real time location system (RTLS) by Pozyx®* to determine the indoor position and orientation of a person using anchors placed in the room and a tag attached to the person.

- *Smartphone App* for configuring the system and for recording and navigating along routes. Its compass feature can also be used as an alternative to the Pozyx tag for providing orientation information.
- A *Raspberry Pi 4* is used for computing operations such as handling services for route recording, providing heading correction for navigation instructions and feedback generation through vibration modes.

To operate the system, all components must be connected to the same network. The Raspberry Pi serves as the primary hub for most of the nodes required in the ROS 2 ecosystem. The vibration vest, with a Pozyx tag attached, can transmit its current position and orientation data to the Raspberry Pi.

A smartphone app provides necessary communication interfaces to the ROS 2 ecosystem, allowing the user to change the parameters of the system and so reconfigure it at runtime. In addition, he can record new routes and navigate along already recorded ones. When a route is selected, the navigation system running on the Raspberry Pi estimates the nearest navigation point available on the route and calculates a heading correction based on the real-time data from the Pozyx tag. This heading correction is translated into an appropriate vibration pattern on the vest to indicate the direction, in which the user should move for safe navigation along the predefined path. As an alternative, audio feedback displayed over the headphones connected to the smartphone can be used for navigation instructions.

The system can be adapted to the model presented in section IV by breaking it down into its components. Here the vest takes on the role of the *display*, the Pozyx tag or the smartphone compass corresponds to the *sensors* and the remaining features of the smartphone application represent the *controls* part. The *internal machine state* is formed by the navigation algorithm and business logic running on the Raspberry Pi. The part of the business logic responsible for recording, persisting and retrieving routes can be seen as an *additional source of information* that enriches the internal machine state and the digital environment model it contains (see Fig. 3).

To demonstrate the practical use of a simulation environment in relation to the development of ETAs in general and specifically with ROS 2, the simulation tool *CARLA*, which is widely used in autonomous driving research, was used. It is also open source and, in addition to existing maps, actors and assets, allows the creation of custom scenarios and the free configuration of environmental factors such as weather and lighting conditions. It also offers a range of different virtual sensors such as LIDAR, cameras, GPS, etc. However, the biggest advantage for the concept proposed in this paper is CARLA's built-in integration with ROS 2 via a bridge. Using predefined ROS 2 topics, it is possible to both read simulated sensor data and control the movement of virtual actors such as pedestrians. In the use case presented here, CARLA replaces the indoor environment and the Pozyx system for determining position and orientation (see Fig. 4). This makes it possible, for example, to test the display components separately without having to deal with sensor inaccuracies or the influence of a test person's behaviour.

Now we will look at the development process and architecture of this system and how ROS 2 supports it and helps to achieve component-based reusability and interchangeability.

B. Development process

Starting with the core functionality, navigation, the necessary nodes, inputs and outputs were defined. The central node provides a single output, a heading correction value. To provide correct and up-to-date values, it requires the route to be followed and the current position and orientation of the user. Inputs and outputs lead to the definition of their respective interfaces and the nodes that provide the necessary inputs. This means that the navigation logic consists of three nodes and has five interfaces (see Fig. 5).

1) *Navigate route* (action), provided by the navigation and called by the user over the smartphone app
2) *Position* (topic), provided by the Pozyx RTLS or the CARLA simulation
3) *Orientation* (topic), provided by the Pozyx RTLS, the smartphone compass or the the CARLA simulation
4) *Load route* (service), provided by the Route Management and called by the navigation logic
5) *Heading correction* (topic), provided by the navigation system and consumed by the feedback device (bHapticsX40 or headphones)

The result is a fully functional navigation system with a freely configurable setup of sensing and user interface devices – even swapping devices on the fly is possible. Each device requires its own node or set of nodes to transfer data to and from the ROS 2 ecosystem and to satisfy the interfaces owned by the navigation service. For example, the software for the bHapticsX40 vest currently consists of two nodes: A driver node responsible for connecting to the vest via Bluetooth, and a feedback node that translates the heading correction feedback into different motor patterns (see Fig. 6).

To enhance usability beyond direct control via command line terminals, a user interface application must access a service client node. The interaction between the user interface and the service client node is the least clean implementation detail, as ROS 2 does not inherently support direct user interaction.

As the manual creation of routes as sets of coordinates was rather tedious and error-prone, the second service, for route recording, was conceived. It allows the user to record their current location and save it as a route for later retrieval by the navigation service. This route recording service was easily implemented using the existing nodes for the navigation service and proved to be a significant improvement over manually entering coordinates. At this stage, control of both services was limited to launching the required nodes with a set of parameters. To increase control and make it more dynamic, separate control nodes with additional user interfaces were next designed and implemented. As the number of nodes and possible configurations increased, it became necessary to

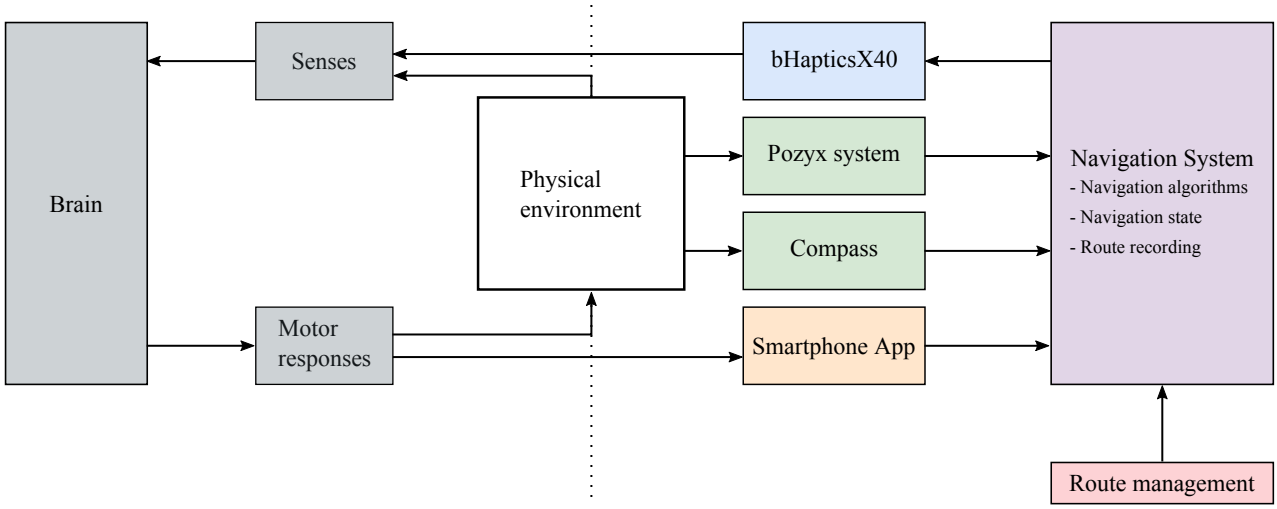


Fig. 3. Indoor navigation system according to Fig. 2

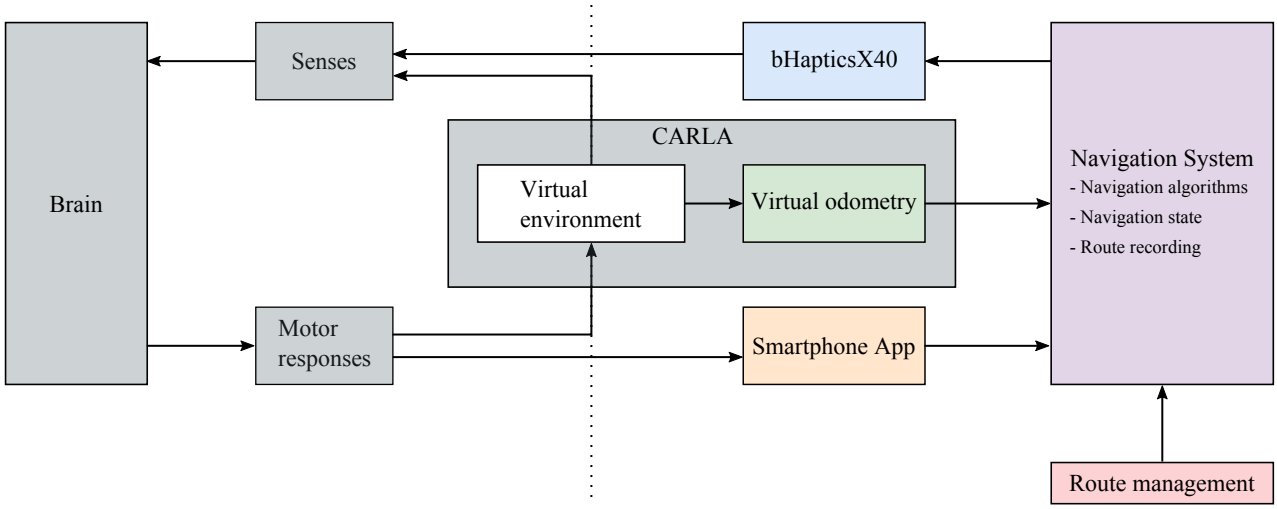


Fig. 4. Indoor navigation system with CARLA simulation as environment and sensors substitution

organize the startup configurations using a modular system of ROS 2 launch files. A semi-automated deployment method allowed different distributions of nodes among hardware components to be tested. The actual (graphical) user interfaces in the form of the smartphone app were the last components to be implemented.

C. Architecture

The resulting architecture follows a microservices approach. For example, the existing system with two services, the navigation itself and a utility for recording the route, can be easily extended, both by adding new types of services and by redundancy of existing ones. This guarantees the degree of scalability and elasticity required by possible use cases, such as indoor navigation in public buildings.

The internal structure of the existing services has many similarities. Both consist of controller, business logic and helper components realized by ROS 2 nodes. The controller nodes provide the user-facing interfaces necessary to control the services and translate ROS external user input for the ROS 2 system. The business logic nodes produce the service functionality, possibly with the help of utility nodes. They interface with the controller nodes via ROS 2 interfaces, i.e., actions and services. By structuring services as a collection of nodes, a single service can be distributed across several separate hardware systems if a specific use case requires it. If this flexibility is not required, the nodes of a service can be run on a single system and configured to run in shared-memory mode to optimize performance. The trade-offs can be considered on a case-by-case basis without changing the node implementation.

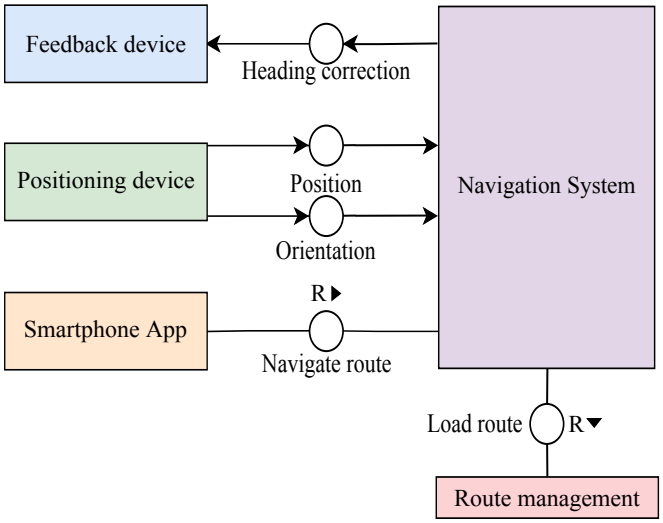


Fig. 5. Abstract FMC model [7] of a navigation system according to Fig. 2

The general trade-offs of the chosen architecture can be summarized as follows: Future requirements for new additional functionality, scalability and elasticity can be easily met. Components, especially sensing and user interface devices, can be added and replaced at low cost. Performance is limited by the degree of distribution chosen for a particular deployment. Even with only two services, the actual implementation is structurally and operationally complex.

VI. Results and Evaluation

Looking at the architecture of the indoor navigation system just presented, one can see the benefits of reusability and interchangeability of individual components presented in section IV for the HMI-ETA model. The bHapticsX40 vibration vest, which acts as a *displays* component and is represented by a package in ROS 2, will serve as an example. The navigation package provides an azimuth and elevation angle via the heading correction topic, which is consumed by the bHapticsX40 package. By defining this asynchronous data channel independent of the display component in the form of a ROS 2 topic, a low degree of coupling is achieved between the package providing the navigation feedback and the package consuming it. This loose coupling allows interchangeability on both sides of the data channel. On the one hand, the display component can be replaced by a headset package, for example, which communicates the feedback to the user acoustically using a text-to-speech algorithm. In this case, both the topic and its message format, as well as the navigation package, could be reused. On the other hand, the indoor navigation of pre-recorded routes could be replaced by any other navigation package (e.g., outdoor navigation using GPS and an external map service), as long as it respects the heading correction topic as an interface to the display component. Again, neither the data channel nor the display component (whether vest or headphones) needs to be touched.

However, ROS 2 supports modularized development of reusable components not only at the level of entire components of the HMI model, but also within the individual components. Fig. 6 illustrates this clearly. It shows the structure of the bHapticsX40 component as a composition of two ROS 2 nodes and the physical vibration vest. The separation into a ROS 2 driver node, which handles the actual Bluetooth communication with the vest, and a feedback generation node, which transforms the received heading correction into a concrete vibration pattern, again promotes loose coupling of the software components. While the former is independent of the concrete overall system and can therefore be reused at any time, the latter can be exchanged in the presented project depending on, which of the two feedback representation modes is used. If, on the other hand, you were to use a similar vest made by a different manufacturer that uses a Wi-Fi connection instead of Bluetooth, replacing the driver node would allow you to quickly reuse the entire system.

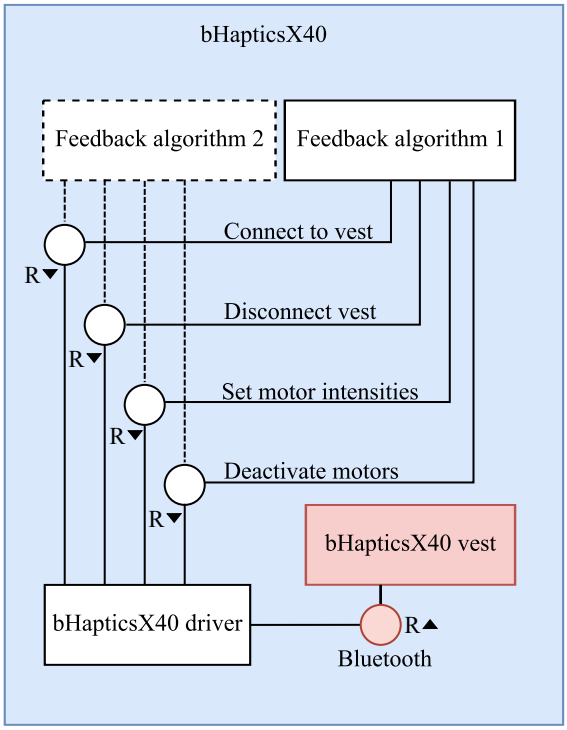


Fig. 6. FMC model of the vibration vest implementation regarding exchangeable algorithms. The red parts are visualized for a full overview and not part of the software system. The bHapticsX40 component is a display device according to Fig. 2

In addition to the simple definition of generic communication interfaces and the resulting loose coupling, the interchangeability of components in ROS 2 is also made possible by the definition of launch files. This is where the entire system is assembled and configured using so-called *launch arguments*. Components in the form of packages and nodes can be added to or removed from the system as needed and can be adjusted by specifying node parameters.

Generally speaking, ROS 2 proved to be very accessible.

The first few tutorials provided enough knowledge to design and implement the whole navigation system with the ROS 2 Python API. Designing a distributed system did not require any additional work thanks to the node structure and the topic, service, and action interfaces. This pre-structuring also assisted with the definition of clean interfaces and division of labour within the team. Deployment, together with launch configurations, was less accessible and considerably less well documented. Only the myriad of existing ROS 2 projects and the associated launch files provided any orientation in this regard.

The lack of documentation in some cases may be due to the fact that ROS 2 is an open-source framework. Although the ROS 2 community is usually very active and helpful, there is no guarantee of support compared to proprietary systems, which can make the lack of documentation all the more problematic. However, the open-source status does have some advantages, including complete transparency in the provision of the source code. This means that issues or vulnerabilities can be discovered and addressed more quickly by the community. It also allows everyone to contribute to the development and to share knowledge and expertise, features that could drive forward ETA research. However, it must be recognized that building up a community can be a long and arduous process. Even ROS, which was developed at Stanford University in 2007 and evolved into its now well-known successor ROS 2 in 2015, did not immediately have the reputation it has today and took years to build such a large community.

The types of problems encountered in robotics have many similarities with those encountered in the indoor navigation project. This means that many of the robotics-oriented packages created for ROS 2 could be adapted accordingly. An example of this is the *tf2* package provided by the ROS 2 community, which makes it possible to track the temporal evolution of several interdependent coordinate systems and to perform transformations between these frames in a simple and efficient way. This is an essential component in robotics, as such calculations are the basis for calculating the individual joints of a robot arm, for example. In the indoor navigation project presented here, tf2 was able to help transform coordinates from the global coordinate system of the Pozyx system to that of the person being navigated, and thus determine a heading correction.

For the same reason, there are already some packages for hardware components for ROS 2 that allow the integration of different sensors from different manufacturers, although this is not visible in the proof of concept. Examples include camera, LIDAR or ultrasound drivers that can be used in robotics as well as ETA development without much expertise or training.

Simulation in ROS 2 is also well-supported. Nodes are configurable for a simulated environment without the need for any code changes. The debugging tools within the ROS 2 ecosystem proved to be extremely helpful and easy to use, as well. Not surprisingly, GUI functionality is an aspect not supported within the ROS 2 ecosystem, but various types of bridge tools provide the possibility to access ROS 2 interfaces.

In order to be able to evaluate the assumptions for the evaluation of a system in reality, an evaluation of 4 different user interfaces was carried out. The user interfaces were used in simulation and in reality. This experimental setup is shown in Fig. 7 and Fig. 8. In addition to the existing nodes, there was a metric node during the test execution, which recorded the positions of the subjects every second and determined the deviations from the ideal route. In addition, the times between waypoints of the route and the total time were recorded. In simulation a single route was run, in reality a total of four. Only sighted people were used to perform the test, and they ran the routes in reality with their eyes open so that there was no risk of injury. Future evaluation runs will be made with people who have different visual impairments. In the simulation, the other extreme was chosen and there was no feedback on position or movement, except for statements about whether someone was standing or moving, or rotating around a point. This minimized the perception of one's presence in the virtual environment. According to Witmer and Signer, this has a weak but consistent effect on task performance, the task in this case being the traversal of a route [11]. The four display components are:

Device 1 A bHapticsX40 vibration vest that continuously vibrates in the target direction.

Device 2 A bHapticsX40 vibration vest that vibrates continuously. It vibrates to the left or right depending on the direction, in which a user is to turn, or straight ahead if a user is aligned with the target point.

Device 3 A text-to-speech interface that announces the target direction as the time. Only 12 or 6 o'clock are replaced by the statements "Perfect" or "Turn Around".

Device 4 A text-to-speech interface that aligns a person to the target direction using "Left", "Right", and "Perfect" instructions.

This paper focuses more on the development and evaluation of ETA components than on the results of evaluations of specific ETA components. Therefore, the focus is on the implementation of the evaluation and the lessons learned. During the execution, each test subject had to complete eight scenarios, each resulting from the use of the four different display components in both simulated and real-world environments. Four different routes were used in reality to prevent the participants from knowing them after the first scenario. Therefore, the individual points of a route had to be navigated, which were spread out in space. The simulation utilized only one route due to the lack of feedback regarding the environment or the subject's movements. To implement a scenario, an entire system must be started along with the simulation if required. The availability of ROS 2 launch files can be helpful while starting the whole system. Launch files are utilized to initiate the entire system with all software nodes and the scenario's parameters. This method enhances reproducibility by describing a device's configuration and parameterization in its constellation. It minimizes the susceptibility to errors

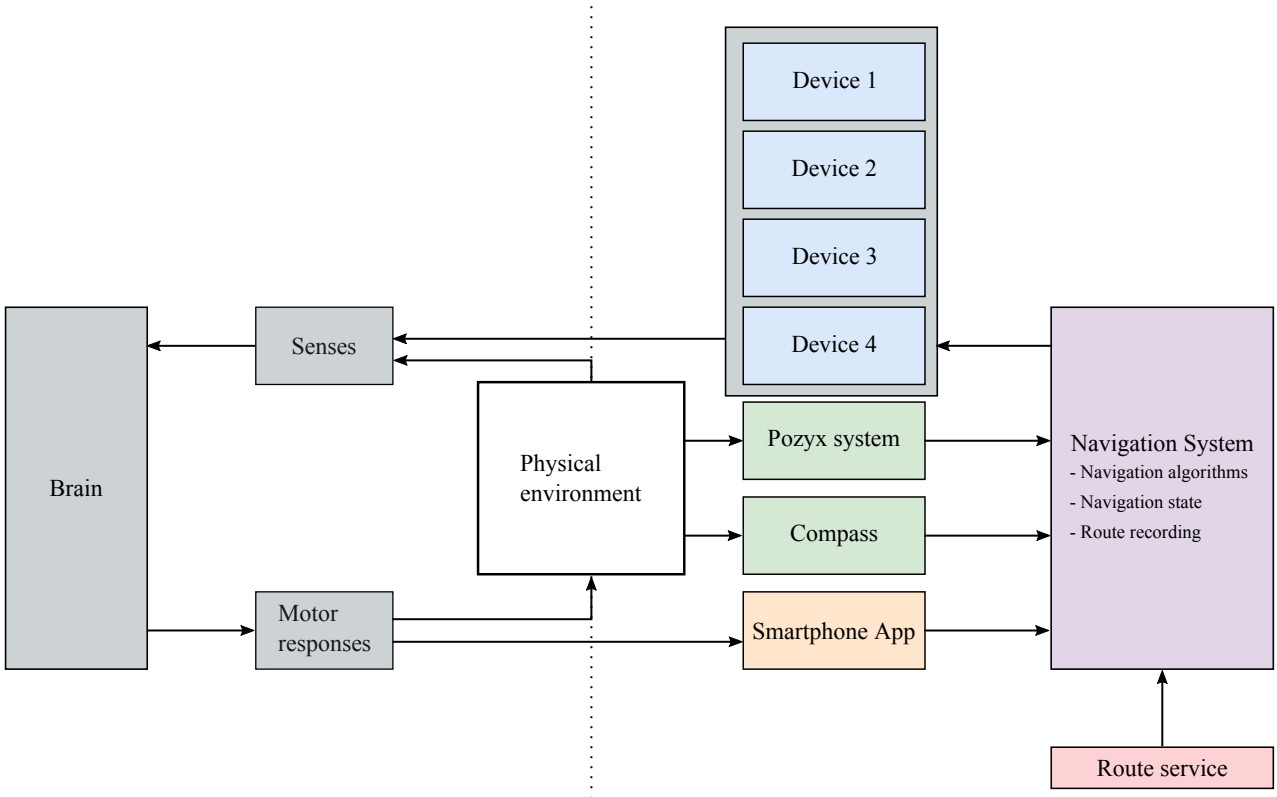


Fig. 7. Four alternative user interfaces for a real-world navigation task.

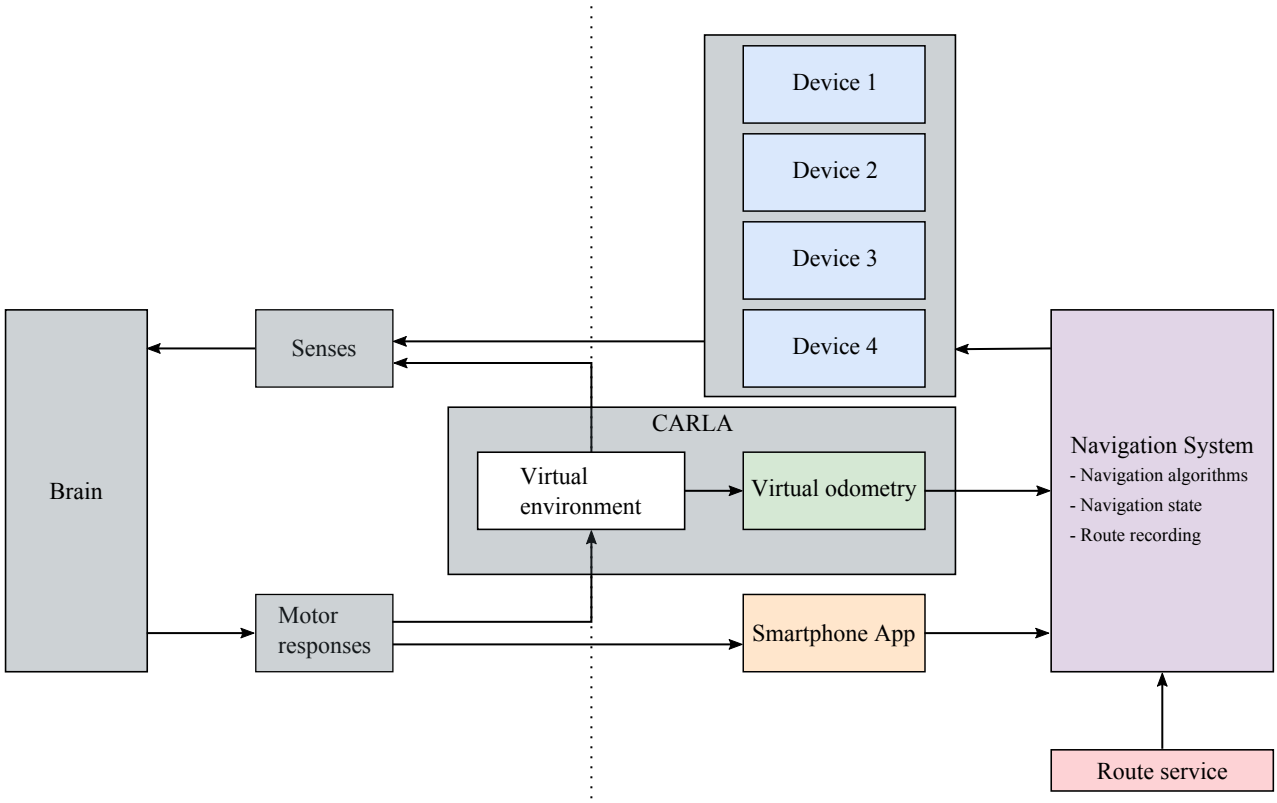


Fig. 8. Four alternative user interfaces for a navigation task in simulation.

that may occur during manual start and parameterization of the components. The participants benefit from reduced waiting times for the preparatory work during a scenario change.

In the experiments, a software node was launched for recording metrics. The software node measures the distance, the user's pose, and the current time. Based on this information, the software node makes a recording every second and writes it to a file, which can be reviewed later. Consequently, information about the user's pose and the distance deviation from the target is available every second. Initially, this data is used to compare devices or algorithms with each other. Subsequently, the information serves as the basis for observations that can be used to enhance the execution of experiments or achieve better results.

Further findings suggest that statements regarding the use of assistive devices in simulation are initially incomparable to their real-world use. This may be attributed to the experimental setup, i.e., participants could work in the real world with full sensory support but relied entirely on the assistive device's feedback without any other sensory perception in simulation. Feedback on the degree of turning can be highly beneficial, particularly when comparing the effectiveness of device 2 compared to device 1 in terms of distances and times per meter, to reach the waypoints of the route (see Fig. 9). The current angle of orientation is represented in device 2 as $\pm 22.5°$. Since the participants could not perceive the angle of rotation in the simulation, the midpoint between these $\pm 22.5°$ was often not determined correctly. Incorporating the results of Device 3 and Device 4 could expose a contradiction at this point. Nevertheless, based on a survey of the participants, it became evident that processing the time information required additional effort, contributing to the further complication of this issue. The times between waypoints support these results.

One notable observation made during the implementation process and supported by the data evaluation is the significant difference in freedom of movement between reality and simulation. This can be seen by comparing Fig. 9 and Fig. 10. Examining the distances to the target route, the simulation range of 1-4 meters (see Fig. 9) exhibits less precision compared to the 0-0.9 meter range in reality (see Fig. 10). In reality, a subject has complete control over their speed and orientation. This is illustrated in Fig. 3 through direct interaction and perception with the surroundings. This type of interaction is more restricted, but only if the environment is simulated. At that point, an interface like a keyboard determines, which movement takes place in the simulation. Yet, the simulation, not the subject, has control over the motion as it interprets a keystroke. The statement is equivalent to the argumentation above in terms of necessary feedback. Nonetheless, the effect differs. Navigation and other algorithms rely on data from the virtual environment, where only rough motor movements are possible. Therefore, tolerances must be larger than those in reality. This was particularly noticeable in our case in terms of navigation, as it heavily relies on user behavior. The system only switches to the next point on the path when a point within

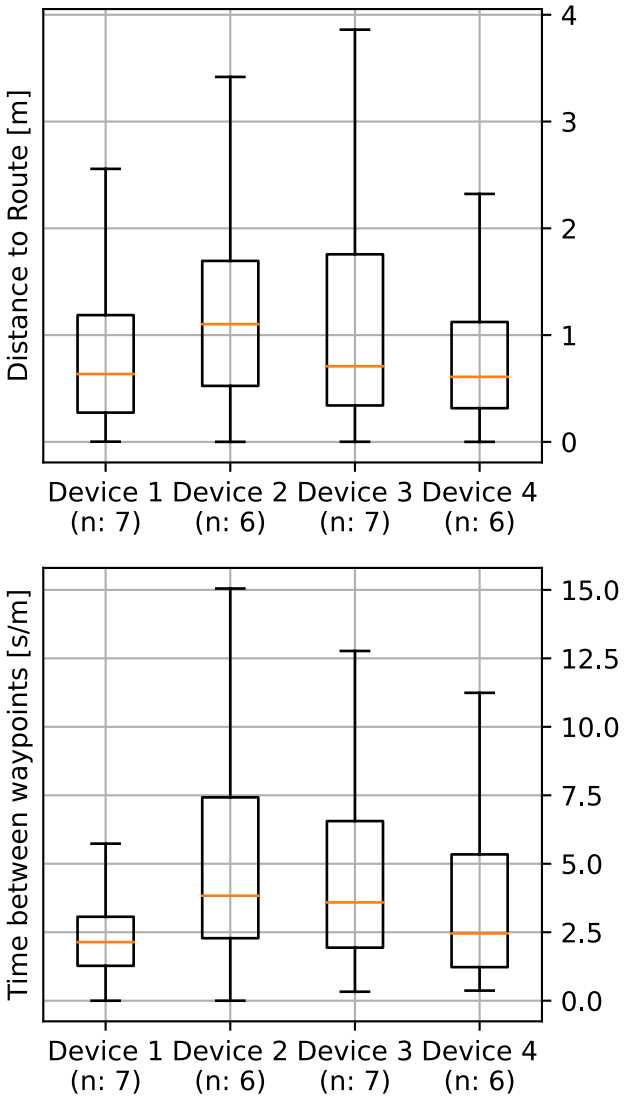


Fig. 9. Simulation results related to the devices. The distance to the route denotes the distance measured towards the designated route every second, expressed in meters. The value of n denotes the number of datasets utilized for the plot. The time between waypoints describes the time spent between waypoints divided by the length between the starting point and the end point.

a radius is reached. While not ideal, this serves as a good example of algorithms that rely on the user's motor skills.

The benefits of interchangeable subsystems in evaluation have been demonstrated. When evaluating display components, a greater level of comparability can be achieved by ensuring identical test setups and data generation procedures. This is necessary because differences in results, despite equivalent interfaces, may be caused by various influences at each level of data generation or even in the test setup. Fig. 7 and Fig. 8 demonstrate that the presented development approach enables a purely interchangeable display component, and that the results are comparable, especially with an unchanged rest of the system. When comparing reality with simulation,

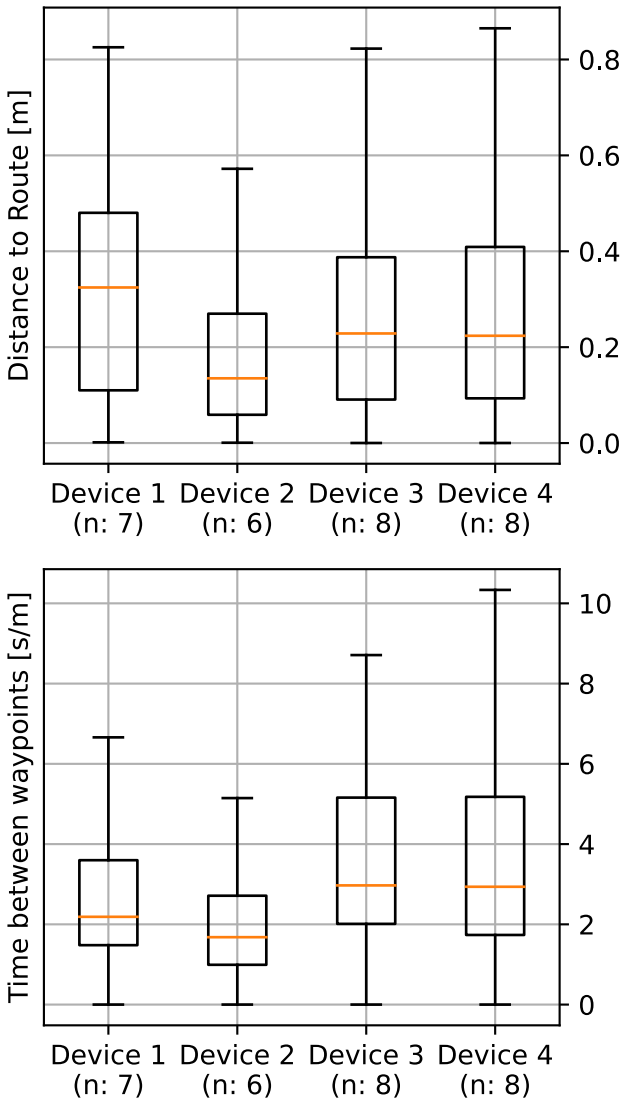


Fig. 10. Real-world results related to the devices. The distance to the route denotes the distance measured towards the designated route every second, expressed in meters. The value of n denotes the number of datasets utilized for the plot. The time between waypoints describes the time spent between waypoints divided by the length between the starting point and the end point.

problems arise due to the strong influence of virtual presence and motor activity on the results.

In summary, ROS 2 provides the necessary building blocks for loose coupling thanks to the provided architecture components such as topics, nodes, packages and launch files, thus supporting a modularized, component-based development of ETAs from the ground up. The challenges and work areas known from (mobile) robotics, which also need to be addressed in ETA development, such as navigation or environmental perception, are facilitated by the tools and drivers already available in ROS 2, allowing most of the time to be spent on the actual development of innovative ideas.

VII. Conclusion

While research into ETAs and mobility aids for the blind and visually impaired in general has produced numerous research papers and demonstrators over the past decades, reviews of these technologies show that the wheel is often reinvented. Both the individual hardware components and the algorithms used to generate feedback, among other things, are redeveloped instead of shared. A lack of exchange between research groups and the use of different development ecosystems means that these software and hardware components often have to be developed and integrated from scratch. This takes up valuable resources that are then not available elsewhere, e.g., for developing innovative concepts or testing them with visually impaired users.

This paper presented an approach to this problem by introducing a framework for a component-based development of ETAs that promotes the reusability and interchangeability of components across projects within a standardized ecosystem. To this end, a mode for identifying ETA component borders by using a human-machine interaction view was presented (see Fig. 2). The components were identified as displays, sensors, controls, a machine internal state and additional information services. It can be concluded that an ETA is generally suitable for decomposition into loosely coupled building blocks. This subdivision can also be seen in systems already known from related literature. Furthermore, individual components can be replaced by a simulation, allowing certain other components to be tested in a more flexible and risk-free manner.

ROS 2 was proposed as an existing open-source framework to support the component-based development of ETAs. The development with ROS 2 and the component-based development was shown in the form of an indoor navigation system. This example uses UWB technology for localization with a vibration vest taken from the virtual reality gaming domain, to present the advantages and disadvantages of such an ETA development. This paper also shows an evaluation of that system and highlights the possibilities of exchangeable feedback devices as well as CARLA as an open-source simulation environment used in autonomous driving. It is highlighted that the results of a simulation highly depend on the feedback available to the user inside the simulation, like when changing orientation or position.

Two features of ROS 2 proved to be particularly important advantages. One is the background of ROS 2, which is mainly in mobile robotics. The overlap between the problems addressed in robotics and ETAs, and the technologies used to address them, is remarkably large. Examples include real-time navigation and environmental perception, for which ROS 2 already provides appropriate sensor drivers, standardized interfaces and algorithms, and tools for testing and visualizing the systems. The use of simulations is also commonplace in robotics.

On the other hand, ROS 2 is designed to support a component-based development. A loose coupling between ROS 2 nodes and packages is enabled by the definition of

custom asynchronous data channels and launch files, in which components can be configured and integrated. This creates the reusability and interchangeability identified in this paper as important drivers of ETA development.

Despite the fact that ROS 2, as an open-source framework, has limitations, such as a lack of documentation or support, it enables rapid results and innovation, not least because of the large and active community, especially for research.

Considering the possible implications for the future development of ETAs using ROS 2, the framework we have started and presented would need to be extended with even more components and system compositions to further illustrate how generic and versatile it is. To facilitate exchange between different research groups, an open platform could be created where components, algorithms and complete systems could be published in the form of ROS 2 nodes, packages and launch files. However, to ensure compatibility between components from different developers and systems, a more concrete policy for their creation needs to be formulated by defining rules and specifications. Then new components could be easily integrated. Although, experience with other open-source frameworks has shown that building a community to collaboratively share knowledge and expertise can be challenging.

References

[1] F. v. Zabiensky, G. Fridman, S. Reuter, O. Özdemir, M. Kreutzer, and D. Bienhaus, "A Framework for Developing Modular Mobility Aids for People with Visual Impairment: An Indoor Navigation Use Case," presented at the SMART ACCESSIBILITY 2023, The Eighth International Conference on Universal Accessibility in the Internet of Things and Smart Environments, Apr. 2023, pp. 10–18.

[2] S. Khan, S. Nazir, and H. U. Khan, "Analysis of Navigation Assistants for Blind and Visually Impaired People: A Systematic Review," IEEE Access, vol. 9, pp. 26712–26734, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3052415.

[3] M. D. Messaoudi, B.-A. J. Menelas, and H. Mcheick, "Review of Navigation Assistive Tools and Technologies for the Visually Impaired," Sensors, vol. 22, no. 20, p. 7888, Oct. 2022, doi: https://doi.org/10.3390/s22207888.

[4] B. Kuriakose, R. Shrestha, and F. E. Sandnes, "Tools and Technologies for Blind and Visually Impaired Navigation Support: A Review," IETE Technical Review, vol. 39, no. 1, pp. 3–18, Jan. 2022, doi: https://doi.org/10.1080/02564602.2020.1819893.

[5] M. Hersh, "Wearable Travel Aids for Blind and Partially Sighted People: A Review with a Focus on Design Issues," Sensors, vol. 22, no. 14, p. 5454, Jul. 2022, doi: https://doi.org/10.3390/s22145454.

[6] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, W. Woodall, "Robot Operating System 2: Design, architecture, and uses in the wild," Science Robotics vol. 7, May 2022.

[7] S. Wendt, "FMC: Home of Fundamental Modeling Concepts," http://www.fmc-modeling.org/ (accessed Dec. 12, 2023).

[8] M. Quigley et al., "ROS: an open-source Robot Operating System," in ICRA workshop on open source software, 2009, vol. 3, no. 3.2, p. 5.

[9] K. Wyrobek, "The Origin Story of ROS, the Linux of Robotics," IEEE Spectrum, Oct. 31, 2017. https://spectrum.ieee.org/the-origin-story-of-ros-the-linux-of-robotics (accessed Dec. 12, 2023).

[10] B. H. Kantowitz and R. D. Sorkin, "Human Factors: Understanding People-System Relationships," 1st edition. New York Chichester Toronto: Wiley, 1991. ISBN: 978-0471095941.

[11] Bob G. Witmer, Michael J. Singer, "Measuring Presence in Virtual Environments: A Presence Questionnaire," Presence: Teleoperators and Virtual Environments, vol. 7, no. 3, pp. 225-240, June 1998. doi: https://doi.org/10.1162/105474698565686.

Comprehensive Machine Learning Analysis of Key Residues in Variants and Polymorphisms for Ace2-Spike Interaction in SARS-CoV-2

Ana Luísa Rodrigues de Ávila, Ana Carolina Damasceno Sanches, Levy Bueno Alves, Silvana Giuliatti
Departament of Genetics. Ribeirão Preto Medical School, University of São Paulo, USP
Ribeirão Preto, Brazil
e-mail: ana.avila@usp.br, anacarolina2001@usp.br, levybuenoalves@usp.br, silvana@fmrp.usp.br

Arthur Scorsolini Fares
Departament of Engineering, Computing and Exact, University of Ribeirão Preto, UNAERP
Ribeirão Preto, Brazil
e-mail: arthur.fares@sou.unaerp.edu.br

***Abstract*— The invasion of SARS-CoV-2 into host cells depends on the interaction of the Spike protein with the human angiotensin-converting enzyme 2 (Ace2). Specific Ace2 polymorphisms have been associated with increased susceptibility to SARS-CoV-2, potentially affecting the risk of infection and the severity of COVID-19. Furthermore, SARS-CoV-2 has a high probability of mutating and adapting to the environment. However, the effect of these genetic variations on the stability and affinity of the Spike-Ace2 interaction is not well understood. For a deeper understanding of this interaction, molecular dynamics simulations are used. Despite generating extensive data, these simulations do not easily facilitate the identification of essential residues that influence protein interaction. To address this challenge, we combined molecular dynamics simulations and supervised machine learning techniques to identify the residues that are subtly important in the interaction and dynamics of the complexes. The molecular dynamics simulations revealed subtle trajectory variations, emphasizing key residues and loop regions residues. While complexes show stable behavior with slight differences, machine learning techniques offer deep insights into how genetic variations in both the virus and host receptor influence the interaction region of these proteins.**

Keywords-COVID-19; Bioinformatics; Virus-host interaction; Polymorphism; Variants.

I. INTRODUCTION

On March 11, 2020, the World Health Organization characterized COVID-19 as a pandemic, identifying it as an infectious disease caused by the Severe Acute Respiratory Syndrome of Coronavirus-2 (SARS-CoV-2) [1] [2]. To date, August 2023, more than 691 million cases have been confirmed, with the global death toll surpassing 6.9 million [3]. In Brazil, listed as one of the most impacted countries, records exceed 37 million confirmed cases and almost 704 thousand deaths [4]. COVID-19 is a respiratory disease primarily transmitted through virus-containing particles that are expelled when an infected person coughs, sneezes, or talks. The severity of the disease can vary from mild cases to severe cases that can lead to Acute Respiratory Distress Syndrome and, in more serious situations, organ failure [5]. People with pre-existing comorbidities and/or who are experiencing some degree of immunosuppression are generally more susceptible to developing severe forms of the disease. Although some people may experience the severe form of COVID-19, others remain asymptomatic [5][6].

The entry of the virus into the host cell is one of the most important processes in viral infection. The virus establishes interactions with specific receptors present on the cell surface, followed by a fusion or endocytosis process, which enables the release of its genetic material into the cell cytoplasm. This viral entry step is a critical target in the development of vaccines and antiviral drugs, as inhibiting or blocking this process can effectively prevent or limit viral replication and spread. The invasion of SARS-CoV-2 into host cells depends on the interaction of the Spike protein with the human angiotensin-converting enzyme 2 (Ace2), which is present in the cell membrane. Certain polymorphisms of the Ace2 protein have been associated with increased susceptibility to SARS-CoV-2 [5][6]. These genetic variations in Ace2 can influence how effectively the virus attaches and enters into host cells, potentially affecting the risk of infection and the severity of the resulting COVID-19 disease.

Furthermore, SARS-CoV-2 has a high probability of mutating and adapting to the environment [7]. The virus has multiple Variants of Concern (VOC) during the course of the pandemic, each with specific mutations that have raised global health concerns. Notable VOCs include Omicron (B.1.1.529 – several countries), Alpha (B.1.1.7 - United States), Beta (B.1.351 – South Africa), Gamma (P.1 - Brazil) and Delta (B.1.617.2 - India) [2]. In addition, there were region-specific variants of interest, such as the P2 (or Zeta variant) (B.1.1.28.2), which was detected in the Rio de Janeiro city, Brazil, in October 2020. The mutations observed in SARS-CoV-2 variants, in conjunction with the Ace2 polymorphisms, raise questions about whether genetic variability of both the virus and the host could explain the different degrees of severity observed in infection cases.

Understanding the complex interaction between viral mutations and host genetic variations is crucial to unraveling the factors that influence disease outcomes. One of the areas of investigation is how these variations impact the stability and affinity of the Spike-Ace2 complexes, which are critical for viral entry into host cells. Certain mutations in the Spike

can increase the ability of the virus to interact more tightly with the Ace2 receptor, potentially leading to increased viral replication and infectivity. Unraveling the mechanisms by which these mutations influence viral entry and replication could open new avenues for therapeutic interventions.

Molecular dynamics (MD) simulations offer valuable information for exploring the effects of mutations and evaluating the stability and affinities between complex structures. However, the trajectories resulting from these simulations generate large amounts of data from thousands of atoms at each time interval. The analysis of complex trajectories can be performed through various approaches, including temporal trajectory analysis, evaluation of thermodynamic properties, and investigation of the bonds and interactions present.

Despite these analytical approaches, the highly dimensional nature and noisy output for the simulations present significant challenges in extracting crucial features from the trajectories. Consequently, it becomes difficult to gain a deeper understanding of molecular processes, such as regions or residues that may subtly contribute to protein interactions. Interpreting and extracting significant information from these trajectories requires robust analysis and is not a simple task.

Machine learning techniques are utilized to analyze extensive datasets, helping to identify crucial distinctions between trajectories obtained in MD simulations, even when these differences are subtle. Fleetwood et al. (2020) demonstrated the utility and potential of machine learning techniques in understanding biomolecular processes [8]. Their work involved the successful application of both supervised and unsupervised methods to investigate three distinct biological systems. In the field of viral interactions with human hosts, Pavdola et al. (2021) employed MD simulations and machine learning techniques to investigate the differences in how SARS-CoV and SARS-CoV-2 interact with the human Ace2 receptor [9]. Inspired by these studies, our research aimed to further explore the interaction between SARS-CoV-2 and Ace2, aiming to fill the knowledge gap about the interaction between viral mutations and genetic variations of the host.

In our previous study, we investigated the effects of genetic variability in SARS-CoV-2 on the interaction with wild type Ace2 [1]. Extensive MD simulations were performed to evaluate the stability of the formed protein complexes and, subsequently, supervised machine learning methods were used considering the trajectories obtained in the simulations as input data. In this study, we expand our analyzis to address the interaction of Spike variants with wild type Ace2, in addition to including new investigations into the effects of Ace2 polymorphisms. The combining of MD simulations and machine learning methods has allowed us to gain deeper insights into how genetic variations in both the virus and the host receptor can impact the region of interaction between these essential proteins.

The structure of this work is outlined as follows: Section II elucidates the methods employed at each stage of this study. Moving forward, Section III elaborates on the results and subsequent discussions, while Section IV summarizes the conclusion and describes the next steps of this research. The article concludes with acknowledgements.

II. Material and Methods

In this section, we will outline the methods employed to perform molecular dynamics simulations and implement machine learning architectures.

A. Molecular Dynamics

The tertiary structure of the complex Spike receptor-binding domain (RBD) and Ace2 (PDB ID: 6LZG) was obtained from the Protein Data Bank [10] and the Modeller software v9.23 [11] was used to fill the missing atoms and residues. The mutant complexes for Spike variants and Ace2 polymorphisms were generated by using the UCFS CHIMERA software version 1.14 [12]. Seven complexes Ace2_Spike-RBD were analyzed: Ace2_Spike-RBD (Wild complex), Ace2_Spike-RBD variants (Omicron, Delta and P2) and Ace2_Spike-RBD complexes with the Ace2 polymorphisms (K26R, R219C, K341R). The Ace2 polymorphisms were selected using the Genome Aggregation Database (GnomAD – https://gnomad.broad.institute.org) and Brazilian Online Mutations Archive (ABraOM – http://abraom.ib.usp.br). The selection of non-synonymous Ace2 mutations was carried out using the following criteria: amino acid residues located in the region of the peptidase domain of Ace2 (19-614 residues) and identified, according to the literature, as critical residues in the interaction between Ace2 and Spike-RBD, polymorphisms found in samples of the Brazilian population deposited in the ABraOM database and which are in high frequency in the population according to GnomAD data.

The systems were solvated in a cubic box with a minimum distance of 1.25 nm from the solute to the edge of the box. GROMACS package version 2020.5 [13] was used in the MD simulations of complexes. The force field used was CHARMM36 [14]. The molecules were solvated with TIP3P water molecules and neutralized by adding the appropriate number of Na+Cl ions considering the ionic concentration of 0.15 M. The energy minimization was performed using the steepest descent method with a maximum force of 1000 $KJ.mol^{-1}.nm^{-1}$. After minimization, the systems were equilibrated in two stages: a canonical NVT set followed by an isothermal-isobaric NPT set. The NVT equilibrium was performed with a constant temperature of 300 K for 500 ps. The NPT equilibrium was performed with a constant pressure of 1 bar and a constant temperature of 300 K for 500 ps. The v-rescale and Parrinello-Rahman algorithms were utilized to keep constant temperature and pressure. The production step was conducted at 300 K for 100 ns and the trajectories were saved every 10 ps.

The Root Mean Square Deviation (RMSD) and Root Mean Square Fluctuation (RMSF) were calculated as metrics to evaluate the structural stability and dynamic fluctuations of the systems. While RMSD measures the average distances between matching atoms in two structures, usually comparing frames obtained during MD simulations with the initial frame (t = 0 ns), RMSF calculates the average squared

fluctuations of atom positions in relation to their average positions throughout a simulation.

To estimate the binding energy and determine the energetic contributions of residues in protein-protein interactions, we utilized the MM/PBSA method [15]. For the MM/PBSA calculations, we included all frames from the final 10 ns of production for each complex. We selected the last 10 ns of the simulations, as during an extended simulation, the systems attain a state of dynamic equilibrium where relevant properties for calculating binding free energy become stabilized. This time window is appropriate for sampling the conformation and properties of the system, not requiring high computational power. Additionally, a shorter time interval helps minimize the effects of initial system fluctuations, allowing for a more accurate estimation of the binding energy.

B. Machine Learning

Based on Fleetwood et al. [8], correlation matrices of filtered contact maps from MD trajectories were used as inputs for supervised ML techniques. In order to reduce the influence of a single model and enhance the stability of our results, we utilized two different classification strategies: Multilayer Perceptron (MLP) and Random Forest (RF). Both methods were used to identify residues that contribute to the difference in dynamic behavior between the complexes (Fig. 1).

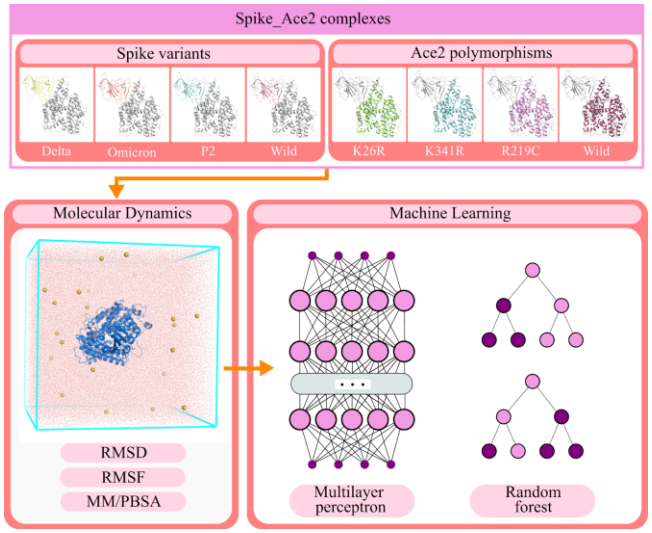


Figure 1. Workflow of the Machine Learning methods used in this study.

The MLP is an artificial neural network with multiple layers between the input and output layers. It is particularly suited for capturing complex, non-linear relationships in data. On the other hand, RF is an ensemble learning technique that builds multiple decision trees and combines their outputs through majority voting. The performance and capability of RF to manage noisy and incomplete data make it a valuable resource in various scenarios. By utilizing both RF and MLP, we ensure a comprehensive exploration of intrinsic relationships and distinctive patterns within the dataset, which ultimately results in more accurate predictions and valuable insights in our analysis.

What constitutes the input features for the MLP are correlation matrices obtained from contact distances between Ace2 and Spike-RBD residues. The distances were calculated as the minimum distance between the heavy atoms of residues in the interaction region and then filtered, leaving only the distances less than 15 Å, in order to establish a predetermined range of analysis for the studied regions. The values were then inverted and normalized to be used for the calculation of the correlation matrix, which was also filtered. Correlations over 0.9 were discarded, as the objective was to identify residues that are not easily recognized as significant contributors to the interaction.

In the MLP, four additional profiles were generated for each complex using bootstrapping, aiming to enhance the classifiers performance. As a result, five profiles were obtained for each complex.

The MLP was implemented using the open-source ML library Scikit-learn in Python [16]. We also used the data analysis and manipulation library Pandas [17], and the numerical computing library NumPy [18]. For the structure of the MLP, 8 hidden layers were used, with 100, 75, 50, 40, 30, 20, 10, and 5 neurons respectively, each with the ReLU function as activation. The labels (Ace2_Spike-RBD complex) were one-hot encoded to represent categorical data numerically and the training process used the Adam optimizer [19] to adjust node weights. A train-test split was applied, with 80% of the data in the training set and 20% in the test set.

This network was trained with each of the profiles, resulting in 5 total MLPs for the complexes with Spike-RBD variants and Wild type (WT) and 5 for the complexes with the Ace2 polymorphisms and WT.

Since the classification task itself does not directly indicate which were the important features that influenced the prediction, an explanation algorithm was applied for the model. The one selected was the Layer-Wise Relevance Propagation (LRP) [20] with the LRP-0 rule. This algorithm indicates which inputs had the most impact on a specific prediction made by the model, obtaining this through the allocation of a normalized relevance score to each individual feature. Therefore, making the decision-making of the neural network more transparent.

The RF classifier, also implemented with Scikit-learn, receives as inputs the distances matrix, since it uses an internal bootstrapping process to produce consistent profiles, and the number of decision trees was set to 100. Our model utilized the Gini impurity coefficient, ranging from zero to one. Zero indicating a pure split and one indicating maximum impurity. We aimed to select splits that would lower Gini impurity, resulting in more homogeneous distribution of classes within the leaves of the tree.

To calculate the importance of a specific state in the RF model, the one-versus-the-rest approach was employed. This strategic method decomposes the problem into multiple binary classification instances and endeavors to discriminate each individual case.

III. Results and Discussion

The outcomes achieved at each step of our work will be detailed in the subsequent subsections.

A. Analysis of trajectory stability

The simulation data was used to compute RMSD for Ace2_Spike-RBD complexes, considering two distinct approaches: the first one considering the interaction between Ace2 WT and Spike-RBD variants; and the second one analyzing the interaction between Ace2 polymorphisms and Spike-RBD WT. Fig. 2 shows the RMSD values for the Ace2 and Spike-RBD proteins.

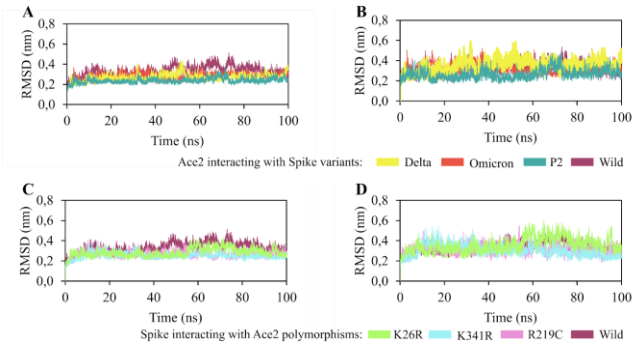


Figure 2. Analysis of the RMSD trajectories obtained in the MD simulations. (A) RMSD of Ace2 WT interacting with Spike-RBD variants; (B) RMSD of Spike-RBD variants interacting with Ace2 WT; (C) RMSD of Ace2 polymorphisms interacting with Spike-RBD WT; (D) RMSD of Spike-RBD WT interacting with Ace2 polymorphisms.

In the Ace2 chain trajectory (Figs. 2A and 2C) involving the interaction with Spike-RBD, the RMSD showed similar values (between 0.2 nm and 0.4 nm) and low standard deviation, implying their stability. The trajectory of the Ace2 WT had a subtle higher RMSD value compared to the other complexes, revealing greater structural variation over the analyzed time. The Spike-RBD trajectories (Figures 2B and 2D) of all complexes remained in equilibrium, with RMSD values between 0.2 and 0.6 nm. All analyzed trajectories of the Spike-RBD complex interacting with Ace2 polymorphisms similarly exhibited stability throughout the simulation.

Stable trajectories indicate that the simulation is converging to an equilibrium state, where the properties of the systems stop showing significant variations. This stability enhances the reliability and precision of the simulation data, providing valuable insights into the behavior and interactions of the studied complexes.

B. Analysis of the Atomic Position Variation

The RMSF represents the degree of variation in the position of a given atom during the course of time. Higher values of RMSF per residue characterize greater flexibility, and vice versa [21]. The RMSF results of the interaction between Ace2 WT and Spike-RBD variants, along with the interaction between Ace2 polymorphisms and Spike-RBD WT, are presented and discussed in sections 1 and 2, respectively.

1) Mobility Analysis of Ace2 WT Associated with Spike-RBD Variants.: The RMSF analysis revealed the residues that exhibited the most significant fluctuations in the trajectories of the Ace2 protein (Fig. 3A) and Spike-RBD (Fig. 3B). On the Ace2 trajectories, the most pronounced fluctuations occur predominantly in loop regions, specifically in residues near Pro138, Gln287, Asn290, Gln340, and Phe428 (Fig. 3C). Notably, Gln340 exhibited the highest peak in the trajectory of the Delta variant (0.67 nm), with a difference of 0.36 nm from the other trajectories. Among these residues, Gln340 is the only one located in a loop relatively close to the Spike-RBD.

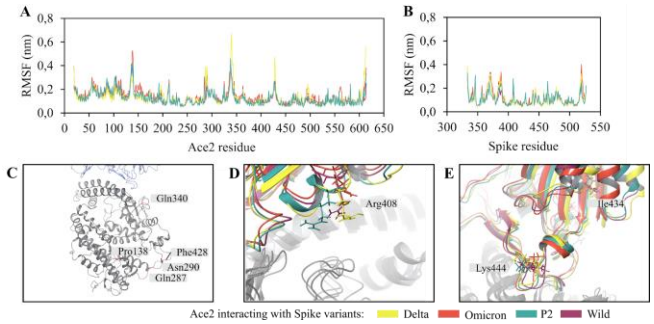


Figure 3. Analysis of residual RMSF from Ace2 protein interacting with Spike-RBD variants. (A) Residual RMSF of Ace2 WT in the Ace2_Spike-RBD complex; (B) Residual RMSF of Spike-RBD variantes in the Ace2_Spike-RBD complex; (C) Fluctuations in Ace2 WT loop regions; (D) Fluctuations in Arg408 residue of Spike-RBD variants; (E) Fluctuations in residues Ile434 and Lys444 of Spike-RBD variants.

Regarding the Spike-RBD trajectories (Fig. 3B), notable fluctuations were observed for the Arg408, Ile434 and Lys444 residues. The RMSF values showed that Arg408 residue of the Delta variant obtained a slightly lower RMSF value (0.13 nm) compared to the WT (0.21 nm), Omicron (0.25 nm) and P2 (0.28 nm) variant complexes. This residue is situated in an alpha-helix, near the interaction interface of Spike-Ace2 (Fig. 3D).

Arg408 is adjacent to residue 417, which has mutated into the Omicron variant, resulting in an amino acid switch from lysine to asparagine (K417N). In the Spike WT, Lys417 forms a very stable salt bridge with the aspartate at residue 30 of the Ace2 receptor. The replacement of lysine for asparagine or threonine largely disrupts binding at this position, as it induces a loss of the salt bridge at this position [22].

As for the Omicron variant complex, the Ile434 residue showed a slightly higher fluctuation (0.12 nm) compared to the Spike-RBD WT (0.7 nm), in addition to Delta (0.7 nm) and P2 (0.6 nm) variants. Ile434 is located in a beta-sheet, but it is not close to the Spike-RBD (Fig. 3E). Isoleucine, being a non-polar amino acid, plays an important role in the structural stabilization of proteins due to hydrophobic interactions within its interior. Furthermore, Ile434 is close to the S375F and N440K mutations in the Omicron variant.

Such mutations decrease the protein stability, which may explain the greater fluctuation observed in this variant [22].

The Lys444 residue of the Delta variant had the highest fluctuation peak of 0.20 nm, followed by the Omicron variant (0.16 nm), P2 variant (0.16 nm), and Spike-RBD WT (0.14 nm). Lys444 is located in a loop close to Gly446, Tyr449, Gln498, Thr500 and Asn501 in Spike WT, which are involved in polar interactions with Ace2 (Fig. 3E) [23]. The L452R mutation of the Delta variant, which is relatively close to the Lys444 residue, reduces protein stability [24].

2) Mobility Analysis of Spike WT Associated with Ace2 Polymorphisms: The RMSF results for Ace2 polymorphisms and Spike-RBD are shown in Figs. 4A and 4B, respectively. The most prominent RMSF fluctuations are observed in loop regions, which were previously highlighted in the previous section and are represented in Fig. 3C. However, a few subtle fluctuations have been identified near the Ace2 polymorphisms regions, such as those observed in Gln325, Trp328, and Arg582.

The Gln325 residue is situated in an alpha-helix near the Spike-RBD and exhibited a slightly reduced fluctuation in the trajectory of the complex with the K341R polymorphism (0.18 nm), followed by WT (0.26 nm), R219C (0.27 nm), and the largest, K26R (0.30 nm). Similarly, Trp328 also showed a smaller fluctuation in the trajectory of the K341R polymorphism, with a difference of 0.16 nm compared to the other complexes. The tryptophan residue is located in an alpha-helix next to the Spike-RBD. Both Gln325 and Trp328 are positioned near the K341R polymorphism (Fig. 4C).

The residue Arg582 of the Ace2 WT obtained a slightly lower RMSF value (0.07 nm) compared to the complexes with the polymorphisms K26R (0.25 nm), R219C (0.20 nm) and K341R (0.19 nm). Residue Arg582 is located in an alpha helix close to the R219C polymorphism (Fig. 4D).

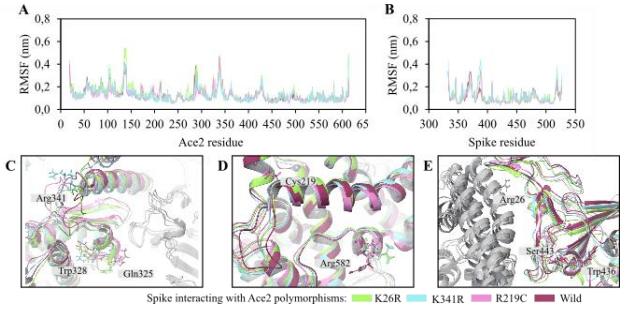


Figure 4. Analysis of Residual RMSF from Spike WT interacting with Ace2 polymorphisms. (A) Residual RMSF of Ace2 polymorphisms in the Ace2_Spike-RBD complex; (B) Residual RMSF of Spike-RBD in the Ace2_Spike-RBD complex; (C) Fluctuations in residues Gln325 and Trp328 of Ace2 polymorphisms; (D) Fluctuations in Arg582 residue of Ace2 polymorphisms; (E) Fluctuations in Trp436 and Ser443 residues of Spike-RBD.

In the RMSF plot of the Spike-RBD chain (Fig. 4B), the Ace2_Spike-RBD complex featuring the K26R polymorphism demonstrated a slightly elevated RMSF value at residue Trp436 within the Spike protein (with a 0.1 nm difference) compared to complexes involving other polymorphisms. The Arg26 polymorphism represents a polar residue located within an alpha helix, near to the Spike-RBD, while Trp436 is an aromatic amino acid with non-polar characteristics, positioned within a beta sheet structure (Fig. 4E). Additionally, the complex with the G211R polymorphism exhibited marginally higher RMSF values in the Ser443 residue, showcasing a difference of 0.27 nm in relation to the other polymorphisms. Situated adjacent to Spike-RBD within an alpha helix, the residue Ser443 emerges as an uncharged polar element.

C. MM/PBSA Binding Free Energy Analysis

The MM/PBSA calculation was performed to estimate the binding energies between the Ace2 and Spike proteins, along with to comprehend the factors contributing to the stability or instability of the interaction. The binding energy values are summarized in Table 1, revealing that the Omicron variant demonstrates a more affinity with Ace2, showing a binding energy of -2572.23 ± 144.92 KJ.mol^{-1}, followed by the Delta and P2 variants, which record values of -1875.71 ± 132.97 and -1837.21 KJ.mol^{-1}, respectively. The analysis of MM/PBSA residual energy decomposition has revealed that it is the electrostatic interactions that predominantly influence the stability of the Spike-RBD variants in the protein-protein interaction. On the other hand, the values obtained for the Van der Waals energy components, solvation polar energy and SASA did not show significant variations. Regarding the interaction of Spike-RBD WT with Ace2 polymorphisms, the estimated values for binding energy did not exhibit significant variations. This can be attributed to the point-wise changes in Ace2 polymorphisms, which were not sufficient to differentiate the binding interaction energy.

TABLE I. RESIDUAL DECOMPOSITION AND BINDING ENERGY (GIVEN IN KJ.MOL^{-1}) IN THE PROTEIN-PROTEIN INTERACTION.

ACE2	SPIKE	ΔEvdW	ΔEelec	Polar sovatation energy	SASA energy	Binding energy
Wild	Wild	-321.60 ± 22.54	-1429.29 ± 59.43	785.31 ± 107.76	-40.91 ± 3.42	-1006.49 ± 115.09
Wild	Delta	-315.86 ± 19.37	-2046.18 ± 73.33	527.73 ± 131.85	-41.40 ± 3.56	-1875.71 ± 132.97
Wild	Omicron	-328.23 ± 20.67	-2866.46 ± 96.49	664.30 ± 135.75	-41.83 ± 3.54	-2572.23 ± 144.92
Wild	P2	-322.53 ± 24.99	-2190.35 ± 71.59	719.82 ± 115.05	-44.16 ± 3.58	-1837.21 ± 119.87
K26R	Wild	-323.33 ± 26.13	-1243.89 ± 76.66	612.92 ± 157.23	-41.41 ± 3.32	-995.71 ± 140.33
K341R	Wild	-333.92 ± 21.91	-1394.38±78.38	801.89 ± 159.18	-42.52 ± 3.73	-968.94 ± 171.39
R219C	Wild	-337.37 ± 20.15	-1433.61 ± 65.65	669.35 ± 174.18	-41.37 ±4.13	-1142.99 ± 161.39

Fig. 5 illustrates the per-residue energy decomposition through MM/PBSA of the analyzed complexes. Although Ace2 residues showed subtle variations in interaction with the Spike variants (Fig. 5A), variations within Spike residues themselves were more pronounced (Fig. 5B). Specifically, the residues at positions 408, 417, 440, 452, 478, 484, 493, and 498 exhibited significant energetic variations.

In the Spike-RBD chain (Fig. 5B), the Omicron variant shows a slightly decreased binding energy at the Arg408 residue in comparison to WT and other variants. Furthermore, Lys417 residue in the Omicron variant shows a weaker binding energy than in the other complexes (-4.72 KJ.mol^{-1}), with a discrepancy greater than 200 KJ.mol^{-1}. As previously mentioned, the substitution of lysine with asparagine, as observed in the Omicron variant, disrupts a

highly stable salt bridge interaction between Lys417 and Ace2 residue 30, consequently destabilizing the complex. In addition, the N440K mutation present in the Omicron variant records a value of -225,52 KJ.mol^{-1}, differing by more than 200 KJ.mol^{-1} from the other complexes.

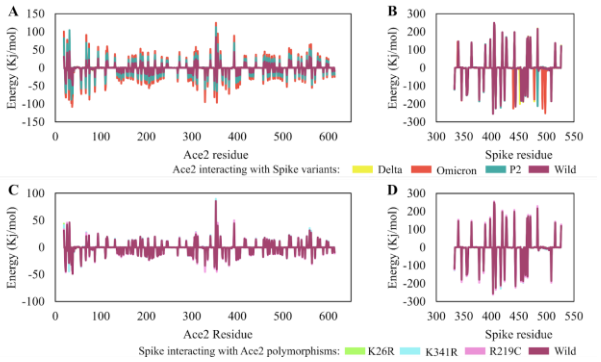


Figure 5. MM/PBSA per-residue energy decomposition for the trajectories obtained in the MD simulations. (A) Contribution energy of Ace2 WT residues in interaction with Spike-RBD variants; (B) Contribution energy of Spike-RBD variants residues in interaction with Ace2 WT; (C) Contribution energy of Ace2 polymorphism residues in interaction with Spike-RBD WT; (D) Contribution energy of Spike-RBD WT residues in interaction with Ace2 polymorphisms.

In the Delta variant complex, the L452R mutation showed a binding energy of -200.19 KJ.mol^{-1}, while in the other complexes the residue reached values around -1 KJ.mol^{-1}. The residue L452R is located in the hydrophobic region of the Spike protein and does not interact with the Ace2 receptor. However, there is a possibility that the mutation induces structural changes by promoting its interaction with the Ace2 receptor [25].

The T478K mutation, identified in the Delta and Omicron variants, resulted in binding energies of -186.69 and -175.78 KJ.mol^{-1}, respectively. In contrast, the WT and P2 complexes showed binding energies around -1 KJ.mol^{-1}. Replacement of the polar and uncharged threonine residue with the positively charged basic amino acid lysine increases the electrostatic potential contribution within the Spike-RBD, promoting stronger affinity with the Ace2 receptor. Furthermore, the elongated lysine side chain could increase the steric effects of the Delta variant, potentially elucidating the increased interaction between Spike-RBD and the Ace2 receptor [26].

The Glu484 residue is mutated in the Omicron variant, involving the replacement of glutamine by alanine, and likewise in the P2 variant, but with the replacement of glutamine by lysine. Notably, the E484K mutation in the P2 variant produced a higher binding energy (-211.87 KJ.mol^{-1}), contrasting with the E484A mutation observed in the Omicron variant (-1.76 KJ.mol^{-1}). The E484A mutation within the Omicron variant abolishes the weak binding of Glu484 to Ace2 in the WT, while mitigating the destabilization arising from conceivable electrostatic repulsion between Glu484 from WT and Glu35 from Ace2 after transition to Ala484. Consequently, the E484A mutation has no influence on the binding free energy, in contrast to the E484K mutation observed in P2, which increases the interaction.

Q493R and Q498R are mutations present in the Omicron variant that showed a reduction in binding energy, with values of -226 and -252 KJ.mol^{-1}, respectively, with a difference of more than 200 KJ.mol^{-1} from the Delta and WT. The combination of the Q498R and N501Y mutations significantly increases Ace2 binding capacity due to the formation of two new strong salt bridges between Arg493 and Arg498 of Omicron, and Glu35 and Asp38 in Ace2 [27].

Regarding the complexes between Ace2 polymorphisms and Spike-RBD WT, no significant values were observed in the MM/PBSA results for the residues in question (Figs. 5C and 5D). However, the residues Val343 and His345, located in the Ace2 chain of the complex containing the K341R polymorphism, showed slightly weaker binding energy compared to the other complexes (Fig. 5C). Val343 and His345 are situated close to the K341R polymorphism, all within the loop region and relatively near the Spike-RBD.

D. Discriminatory Residues of Spike Variants through Machine Learning

Table 2 shows the five most significant residue pairs for each complex. In the MLP, the importance values of each residue pair were derived by calculating the average of the associated LRP-0 attributes. On the other hand, in the RF, the importance was evaluated based on the reduction of Gini impurity. Key residues responsible for variations in binding between Spike-RBD variants and Ace2 WT were identified, some of them already reported in previous studies.

TABLE II. IMPORTANT RESIDUES OBTAINED FROM MLP AND RF FOR SPIKE VARIANTS INTERACTING WITH ACE2 WT. THE PAIRS HIGHLIGHTED IN BOLD ARE SUPPORTED BY THE LITERATURE AND ARE DISCUSSED IN THIS STUDY.

MLP			
Spike variants	ACE2	SPIKE	Importance
Wild	SER106	GLY485	1.00
	VAL107	PHE486	0.99
	GLN89	SER477	0.98
	SER19	**PRO479**	**0.89**
	ALA71	GLU484	0.82
Delta	ASP30	GLU484	1.00
	GLN24	LYS417	0.71
	GLY352	ARG408	0.68
	ALA65	SER443	0.62
	ASN33	GLN498	0.60
Omicron	GLU329	SER438	1.00
	GLN42	SER349	0.96
	TYR381	GLY502	0.92
	GLY352	ASN448	0.91
	GLY354	GLY504	0.90
P2	PRO321	ARG403	1.00
	SER19	**ASN477**	**0.92**
	SER19	**PRO479**	**0.87**
	GLN325	SER371	0.84
	GLU37	THR415	0.81

RF			
Spike variants	ACE2	SPIKE	Importane
Wild	SER19	VAL483	1.00
	SER19	CYS488	0.95
	SER19	CYS480	0.60
	SER44	TYR505	0.52
	SER19	GLN474	0.52
Delta	ALA36	ASN501	1.00
	GLY66	ASN501	0.98
	ALA342	THR500	0.93
	ASN103	TYR505	0.71
	LYS68	ASN501	0.64
Omicron	**ALA25**	**ASN417**	**1.00**
	GLN24	**ASN417**	**0.98**
	ILE21	**ASN417**	**0.97**
	LYS353	**ARG498**	**0.94**
	THR27	**ASN417**	**0.88**
P2	**SER106**	**LYS484**	**1.00**
	SER19	CYS480	0.90
	SER105	ASN487	0.87
	GLY104	ASN487	0.80
	SER105	**LYS484**	**0.79**

Fig. 6 illustrates the importance values of the pairs obtained through MLP (Fig. 6.A) and RF (Fig. 6B), highlighting the most relevant residues that distinguish the variants from the WT (Figs. 6C-E).

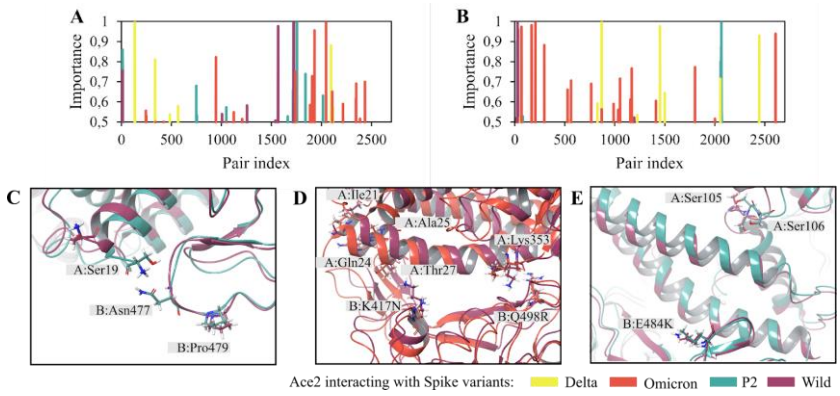


Figure 6. Pairs of residues determined as most important for distinguishing the binding between Ace2 WT and Spike-RBD variants. (A) Pairs identified by MLP; (B) Pairs identified by RF; (C) Pairs of important residues found for Spike-RBD WT and P2; (D) Pairs of important residues found for Spike-RBD WT and Omicron; (E) Pairs of important residues found for Spike-RBD WT and P2. Important residues are composed of highly distinct pairs of distances between Ace2 WT and the studied Spike-RBD variants.

The residue pairs identified by the RF model differed from those identified by the MLP model. However, some residues were identified by both methods. The analysis of the results emphasizes the importance of the main residues of Ace2 in the interaction with Spike-RBD, as mentioned by Ali and Vijayan (2020), which include Gln24, Thr27, Asp30, Glu37, Gln42, and Lys353 [28]. Furthermore, the Ser19 residue of Ace2 protein, which was commonly seen among pairs, is also important. Ser19 participates in a network of hydrogen interactions, particularly interacting with Pro462 [29].

The results obtained by the MLP suggest that Ser19 can subtly interact with residues near Pro462, forming pairs with Pro479 in the Spike-RBD WT, and Asn477 and Pro479 in the P2 variant. The S477N mutation, present in the P2 variant, may favor a greater interaction with Ser19 of Ace2 protein. In addition, a mutation S19P increased the interaction between Ace2 and Spike-RBD [30]. This suggests that the Ser19 residue plays a critical role in modulating the interaction between Ace2 and Spike, potentially influencing the infection capability and viral transmissibility of the P2 variant. The mentioned pairs may be of importance in distinguishing between the binding of Spike-RBD WT and P2 to Ace2 (Fig. 6C).

In the RF model, the mutations K417N and Q498R of the Omicron variant formed interesting residue pairs. The K417N mutation may have significant effects with residues Ile21, Gln24, Ala25, and The27, while the Q498R mutation may affect Lys353 of Ace2 protein (Fig. 6D). Studies indicate that the K417N mutation leads to a reduction in the binding affinity of the Spike to Ace2; however, it is an immune escape mutation, which helps SARS-CoV-2 escape the natural immune defenses of the host, contributing to increased viral transmissibility. On the other hand, Q498R is associated with an increase in viral infection. The Lys353-Arg498 pair had an importance of 0.94. The study by Zhang et al. (2022) suggests that Q498R is structurally incompatible with Lys353 in Ace2, but is structurally adapted to Asp38 [31].

The E484K mutation, found in the P2 variant, is associated with a reduction in neutralizing antibodies. This mutation results in a tighter binding interface between Spike-RBD and Ace2 protein, contributing to an increase in binding affinity [32]. Our results indicate the presence of two pairs related to this residue, namely Ser106-Lys484 with a significance of 1.00, and Ser105-Lys484 with a significance of 0.79 (Fig. 6E).

E. Discriminatory Residues of Ace2 polymorphisms through Machine Learning

Table 3 shows the five pairs responsible for variations in the binding between Ace2 polymorphisms and Spike-RBD. Both the RF and MLP models identified identical residue pairs, such as Phe356-Tyr495 in K26R, and Gly104-Phe486 and Ser105-Tyr489 in K341R. For ACE2 WT and R219C polymorphism, the residue pairs identified by the RF model differed from those identified by the MLP model. However, the residue Phe486 (in R219C) and Gln325 (in WT) were identified in both methods. Notably, the Ser19 residue in the Ace2 protein received high scores for all the polymorphisms, including K26R, K341R, and R219C identified by MLP, in addition to the WT identified by RF.

TABLE III. IMPORTANT RESIDUES OBTAINED FROM MLP AND RF FOR ACE2 POLYMORPHISMS INTERACTING WITH SPIKE-RBD WT. THE PAIRS HIGHLIGHTED IN BOLD AND MARKED WITH ASTERISKS APPEARED IN BOTH MLP AND RF ANALYSES. THE REMAINING BOLD PAIRS ARE ALSO DISCUSSED IN THIS STUDY.

MLP			
ACE2 polymorphisms	ACE2	SPIKE	Importance
K26R	SER19	SER477	1.00
	SER19	ASP467	0.91
	PHE356*	**TYR495***	**0.80***
	LEU351	TYR495	0.41
	ASN194	PHE486	0.39
K341R	**GLY104***	**PHE486***	**1.00***
	SER19	PRO479	1.00
	ALA193	PHE486	0.86
	SER105*	**TYR489***	**0.80***
	ASN103	TYR505	0.77
R219C	SER19	PRO479	1.00
	ASP38	LYS444	0.54
	PHE32	GLY446	0.43
	PHE40	TYR495	0.43
	ALA193	**PHE486**	0.41
Wild	ASN103	TYR505	1.00
	ASN103	TYR489	0.85
	ASN103	GLY485	0.80
	HIS401	TYR505	0.74
	GLN325	TYR449	0.54

RF			
ACE2 polymorphisms	ACE2	SPIKE	Importane
K26R	**PHE356***	**TYR495***	**1.00***
	LYS353	TYR495	0.94
	GLY352	TYR495	0.94
	ASP355	ASN439	0.79
	ASP350	TYR495	0.66
K341R	**GLY104***	**PHE486***	**1.00***
	GLN102	ASN487	0.79
	SER105*	**TYR489***	**0.78***
	LYS26	GLY504	0.73
	GLN102	TYR489	0.65
R219C	SER105	TYR489	1.00
	GLY104	**PHE486**	0.98
	GLN102	GLY485	0.91
	ASN103	GLY485	0.86
	ASN103	TYR489	0.84
Wild	SER19	ALA475	1.00
	GLN325	THR500	0.96
	SER19	GLN474	0.87
	GLN325	GLN498	0.81
	GLN325	ASN448	0.76

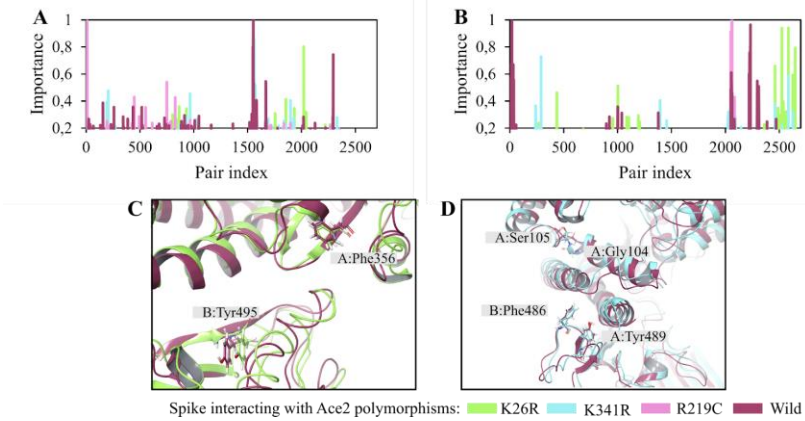


Figure 7. Pairs of residues determined as most important for distinguishing the binding between Spike-RBD WT and Ace2 polymorphisms. (A) Pairs identified by MLP; (B) Pairs identified by RF; (C) Pairs of important residues found for Ace2 WT and K26R; (D) Pairs of important residues found for Ace2 WT and K341R. Important residues are composed of highly distinct pairs of distances between Ace2 polymorphisms and the Spike-RBD.

Fig. 7 shows the significance of the pair values obtained using MLP (Fig. 7A) and RF (Fig. 7B), emphasizing the key residues that differentiate the variants from the WT, as demonstrated in Figs. 7C and 7D.

Ace2 K26R increases susceptibility to SARS-CoV-2 due to a higher binding affinity with the Spike protein [30]. In this polymorphism, Phe326 forms a pair with Tyr495. Phe356 is located near residues Tyr41, Gln42, Lys353, and Arg357, which interact with the Spike-RBD, while Tyr495 participates in a hydrogen bonding network with Ace2. However, there is currently no literature available on the specific importance of Phe356 in the K26R polymorphism when interacting with the Spike-RBD.

The K341R mutation, which replaces lysine with arginine, results in a larger mutant residue, which can cause protrusions [33]. Our research findings highlight significant variations in the pairs Gly104-Phe486 and Ser105-Tyr489. The roles of some of these residues have been documented by Ali et al. (2020) [28]. Specifically, Phe486 participates in important polar interactions with Tyr83 and hydrophobic interactions with Leu79 of Ace2. Tyr489 is involved in polar interactions with Thr27 and Lys31, as well as hydrophobic interactions with Phe28, Tyr83, Thr27, Phe32, and Phe72 from Ace2.

IV. Conclusions and Future Work

The interaction between the Spike and Ace2 proteins plays a crucial role in determining the replication rate of SARS-CoV-2 and has implications for the disease progression in infected individuals. The virus exhibits a pronounced propensity for mutations, as evidenced by the emergence of various variants in recent years. Ace2 genetic polymorphisms have the potential to influence susceptibility to the disease, along with its subsequent intensity and clinical outcome. However, a comprehensive understanding of how mutations and polymorphisms impact the stability and interaction dynamics within the SARS-CoV-2-Ace2 complex remains an ongoing effort.

In our research, we combined MD simulations and machine learning techniques to explore the interaction between different SARS-CoV-2 variants and human Ace2 polymorphisms. Through these simulations, we obtained valuable information about the protein-protein interaction. Concurrently, employing machine learning techniques allowed us to pinpoint critical amino acid residues within the binding region that subtly contribute to this interaction.

The MD simulations revealed similar stability patterns among the studied complexes. Furthermore, the resulting trajectories indicated a convergence of the simulations into an equilibrium state. The stability of the Ace2 protein complex with Spike-RBD WT was slightly diminished, as evidenced by the RMSD values, in contrast to the SARS-CoV-2 variant complexes. This observation is consistent with the anticipated effect of mutations in the Spike interaction region leading to increased stability.

Although the most significant fluctuations were observed in loop regions, some residues near the interaction interface exhibited notable fluctuations. Arg408 and Lys444 of the Spike-RBD showed slightly higher RMSF values in the Delta variant. Gln325 and Trp328 residues of the Ace2 protein showed lower fluctuations in the trajectories of the K341R polymorphism, whereas Trp436 and Ser443 exhibited higher fluctuations for WT and G211R, respectively. Significant changes in RMSF in these regions may suggest important conformational alterations for the biological activity in the Ace2-Spike interaction.

The Omicron variant demonstrates a stronger affinity with Ace2, as evidenced by the MM/PBSA values, where the Q493R and Q498R mutations contributed more significantly to the binding energy. Regarding the complexes formed between Ace2 polymorphisms and the Spike-RBD, no significant differences were identified in the MM/PBSA results for the considered residues. These point polymorphisms were not sufficient to generate detectable notable variations using the MM/PBSA method.

Regarding the Machine Learning methods, we achieved a precision score of 1 and loss values below 0.005 for both

approaches using the test dataset. The high precision and low loss on the test data suggest that the model is performing well, but they do not ensure the absence of overfitting. A more comprehensive evaluation, utilizing other data sources such as cross-validation, is necessary to determine the presence of overfitting.

The ML and RF approaches successfully identified key residues from both proteins responsible for differences in binding region, some of which have been previously reported in the literature. This demonstrates that our method was able to identify residues that significantly contribute to the distinction between virus and host interaction due to Spike variants and Ace2 polymorphisms, extending even to those pairs of residues that have been not previously documented in the existing literature.

Our study shows that machine learning can simplify the complexity of virus-host interactions by reducing dimensionality and identifying crucial residues. Our findings indicate that there may be additional important residues beyond those previously considered above that may impact the interaction between Spike and Ace2 proteins. These residues may account for differences in stability and affinity, leading to varying levels of susceptibility to SARS-CoV-2 and resulting in varying degrees of disease severity. In our work, we aim to gain a deeper understanding of the relationship between mutations and the affinity between Spike-Ace2 by not only exploring other variants, but also incorporating various machine learning methods.

ACKNOWLEDGMENT

Fundacao de Amparo a Pesquisa do Estado de São Paulo (FAPESP) for Sponsorship (2022/02782-5; 2022/03038-8) and Superintendence of Technology and Information at USP for the High-Performance Computing (HPC) Computing Resources.

REFERENCES

[1] A. C. D. Sanches, A. L. R. de Ávila, L. B. Alves, and S. Giuliatti, "Machine Learning in the Identification of Key Residues of Variants and Polymorphisms in the Interaction of ACE2 Proteins with Spike of SARS-CoV2", pp. 7-11, Mar. 2023 [BIOTECHNO 2023, The Fifteenth International Conference on Bioinformatics, Biocomputational Systems and Biotechnologies].

[2] World Health Organization. WHO Director-General's Opening Remarks at the Media Briefing on COVID-19 - 11 March 2020. [Online]. Available from: https://www.who.int/director-general/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---11-march-2020/ (Accessed: 04 December 2023).

[3] Worldometers. COVID-19 CORONAVIRUS PANDEMIC. [Online]. Available at: https://www.worldometers.info/coronavirus (Accessed: 04 December 2023).

[4] World Health Organization. Tracking SARS-CoV-2 variants. [Online]. Available at: https://www.who.int/activities/tracking-SARS-CoV-2-variants (Accessed: 04 December 2023).

[5] S. Choudhary, K. Sreenivasulu, P. Mitra, S. Misra, and P. Sharma, "Role of Genetic Variants and Gene Expression in the Susceptibility and Severity of COVID-19," Annals of Laboratory Medicine, vol. 41(2), pp. 129–138, Mar. 2021, doi: 10.3343/alm.2021.41.2.129.

[6] J. Zepeda-Cervantes et al., "Implications of the Immune Polymorphisms of the Host and the Genetic Variability of SARS-CoV-2 in the Development of COVID-19," Viruses, vol. 14(1), pp. 1–34, Jan. 2022, doi:10.3390/v14010094.

[7] R. Peng, L. -A. Wu, Q. Wang, J. Qi, and G. F. Gao, "Cell entry by SARS-CoV-2," Trends in Biochemical Sciences, vol. 46(10), pp. 848–860, Oct. 2021, doi: 10.1016/j.tibs.2021.06.001.

[8] O. Fleetwood, M. A. Kasimova, A. M. Westerlund, and L. Delemotte, "Molecular Insights from Conformational Ensembles via Machine Learning," Biophysical Journal, vol. 118(3), pp. 765–780, Feb. 2020, doi:10.1016/j.bpj.2019.12.016.

[9] A. Pavdola et al., "Machine Learning Reveals the Critical Interactions for SARS-CoV-2 Spike Protein Binding to Ace2", J. Phys. Chem. Lett., vol. 12(13), pp. 5494–5502, Jun. 2021, doi:10.1021/acs.jpclett.1c01494.

[10] H. M. Berman et al., "The Protein Data Bank," Nucleic Acids Research, vol. 28(1), pp. 235–242, Jan. 2000, doi:10.1093/nar/28.1.235.

[11] A. Sali and T. L. Blundell, "Comparative protein modelling by satisfaction of spatial restraints," J. Mol. Biol., vol. 234(1), pp. 779–815, Dec. 1993, doi:10.1006/jmbi.1993.1626.

[12] E. F. Petersen et al., "UCSF Chimera—A visualization system for exploratory research and analysis", Journal of Computational Chemistry, vol. 25(13), pp. 1605-1612, Oct. 2004, doi:10.1002/jcc.20084.

[13] D. V. D. Spoel, E. Lindahl, B. Hess, G. Groenhof, A. E. Mark, and H. J. C. Berendsen, "GROMACS: Fast, flexible, and free", Journal of Computational Chemistry, vol. 26(16), pp. 1701-1718, Dec. 2005, doi:10.1002/jcc.20291.

[14] J. Huang and A. D. MacKerell, "CHARMM36 all‐atom additive protein force field: Valida-tion based on comparison to NMR data," J. Comput. Chem., vol. 34(1), pp. 2135-2145, Sep. 2013, doi:10.1002/jcc.23354.

[15] R. Kumari and R. Kumar, "Open Source Drug Discovery Consortium et al. g_mmpbsa—A GROMACS Tool for High-Throughput MM-PBSA Calculations", Journal of Chemical Information Model, v. 54(7), pp. 1951-1962, May. 2014, doi:10.1021/ci500020m.

[16] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of machine learning research, vol. 12, pp. 2825–2830, 2011, doi:10.48550/arXiv.1201.0490.

[17] W. McKinney, "Data Structures for Statistical Computing in Python," Proceedings of the Ninth Python in Science Conference (PPSC 2010), pp. 56–61, doi:10.25080/Majora-92bf1922-00a.

[18] C. R. Harris et al., "Array programming with NumPy," Nature, vol. 585, pp. 357–362, 2020, doi:10.1038/s41586-020-2649-2.

[19] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," The Third International Conference for Learning Representations (ICLR 2015), SAN DIEGO, Jan. 2017, doi:10.48550/arXiv.1412.6980.

[20] G. S. Montavon, S. Bach, A. Binder, W. Samek, and K. -R. Müller, "Explaining nonlinear classification decisions with deep Taylor decomposition," Pattern Recognition. vol. 65, pp. 211–222, May. 2017, doi:10.1016/j.patcog.2016.11.008.

[21] A. Bornot, C. Etchebest, and A. G. de Brevern, "Predicting protein flexibility through the prediction of local structures", Proteins, vol. 79(3), pp. 839-852, Mar. 2011, doi:10.1002/prot.22922.

[22] S. Kumar, T. S. Thambiraja, K. Karuppanan, and G. Subramaniam, "Omicron and Delta variant of SARS-CoV-2: A comparative computational study of Spike protein," Journal

of Medical Virology, vol. 94(4), pp. 1641-1649, Apr. 2022, doi:10.1002/jmv.27526.

[23] X. Duan et al., "A non-Ace2-blocking neutralizing antibody against Omicron-included SARS-CoV-2 variants", Signal Transduction and Targeted Therapy, vol. 7(23), pp. 1-3, Jan. 2022, doi:10.1038/s41392-022-00879-2.

[24] T. B. Mahmood et al., "Missense mutations in Spike protein of SARS-CoV-2 delta variant contribute to the alteration in viral structure and interaction with hAce2 receptor", Immun Inflamm Dis., vol. 10(9), Sep. 2022, doi:10.1002/iid3.683.

[25] X. Deng et al., "Transmission, infectivity, and neutralization of a Spike L452R SARS-CoV-2 variant.", Cell., vol 184(13), pp. 3426-3437, Jun. 2021, doi:10.1016/j.cell.2021.04.025.

[26] S. D. Giacomo, D. Mercatelli, A. Rakhimov, and F. M. Giorgi, "Preliminary report on severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) Spike mutation T478K", J Med Virol., vol. 93(9), pp. 5638-5643, Sep. 2021, doi:10.1002/jmv.27062.

[27] B. Jawad, P. Adhikari, R. Podgornik, and W. -Y. Ching, "Binding Interactions between RBD of Spike-Protein and Human Ace2 in Omicron variant", bioRxiv 2022.02.10.480009, 2022, doi:10.1101/2022.02.10.480009.

[28] A. Ali and R. Vijayan, "Dynamics of the Ace2–SARS-CoV-2/SARS-CoV Spike protein interface reveal unique mechanisms", Sci Rep, vol. 10(14214), pp. 1-12, Aug. 2020, doi:10.1038/s41598-020-71188-3.

[29] J. de Andrade, P. F. B. Gonçalves, and P. A. Netz, "Why Does the Novel Coronavirus Spike Protein Interact so Strongly with the Human Ace2? A Thermodynamic Answer", Chembiochem, vol. 22(5), pp. 865-875, Mar. 2021, doi:10.1002/cbic.202000455.

[30] K. Suryamohan et al., "Human Ace2 receptor polymorphisms and altered susceptibility to SARS-CoV-2," Communications Biology, vol. 4(1), pp. 1–11, Apr. 2021, doi:10.1038/s42003-021-02030-3.

[31] W. Zhang et al., "Structural basis for mouse receptor recognition by SARS-CoV-2 omicron variant", PNAS, vol. 119(44), pp. e2206509119, Oct. 2022, doi: 10.1073/pnas.220650911.

[32] W. B. Wang et al., "E484K mutation in SARS-CoV-2 RBD enhances binding affinity with hAce2 but reduces interactions with neutralizing antibodies and nanobodies: Binding free energy calculation studies", J Mol Graph Model, vol. 109(108035), pp. 1-15, Dec. 2021, doi:10.1016/j.jmgm.2021.108035.

[33] J. A. Rodríguez et al., "Computational modeling of the effect of five mutations on the structure of the Ace2 receptor and their correlation with infectivity and virulence of some emerged variants of SARS-CoV-2 suggests mechanisms of binding affinity dysregulation," vol. 368, pp. 110244–110244, Dec. 2022, doi:10.1016/j.cbi.2022.110.