Project Medical Wearables
Report

Forward Head Posture Detection Using IMU Sensors

Date:

8 June 2022

Author:

Hye Jin Kim

Faculty of Engineering, Computer Science, and Psychology

M.Sc. Cognitive Systems

I. Introduction

Today, computers including smartphones and tablets are now an integral part of our daily lives. In a fast-paced global environment, it is difficult to complete tasks without the use of such devices. Although it is natural to learn how to use computers, there is not much emphasis on using the devices with good body posture. In an ideal body posture, the body parts are properly aligned, maintaining the natural curvature of the spine with the neck upright and shoulders parallel with the hip (NHS, n.d.). Poor body posture is found to be more common among people who work with computers for an extended period of time, and one common health problem that can result is Forward Head Posture (FHP) (Kang et al., 2012). FHP is defined as the exaggerated anterior curve in the lower cervical vertebrae and exaggerated posterior curve in the upper thoracic vertebrae. Negative side effects of maintaining FHP for long periods of time include non-traumatic chronic neck pain (Silva, Punt, Sharples, Vilas-Boas, & Johnson, 2009) and an overall imbalance of the musculoskeletal system (Griegel-Morris, Larson, Mueller-Klaus, & Oatis, 1992).

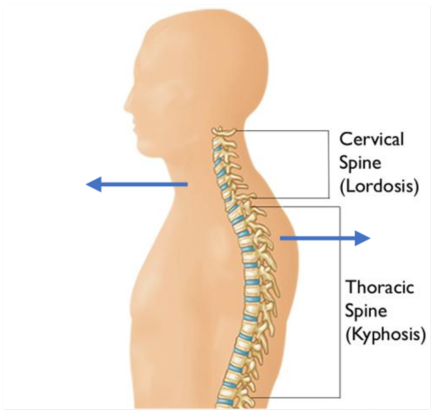

Figure 1: Illustration of the spine adapted from American Academy of Orthopaedic Surgeons (n.d.). The blue arrows indicate the direction in which the cervical and thoracic vertebrae are exaggerated in FHP.

Because of the negative symptoms of FHP, many people want to correct their posture. However, most of the time, one is not conscious of what kind of posture they are in. Therefore, one research question that has been ongoing is how to detect FHP in real-time, and how to give this detection feedback to the user of the FHP detection solution.

There exist several proposed solutions for FHP detection and prevention using various combinations of Medical Wearables and measurements. These include, but are not limited to, the assessment of the craniovertebral angle (CVA) using two inertial body sensor networks (BSNs) (Kim, Chen, & Lach, 2011) as well as assessing the degree of anterior curve of the lower cervical vertebrae with a commercial depth camera (Lee, Cho, Kim, Yoon, & Choi, 2014). While both proposals use a measurement threshold algorithm with high accuracy results, the generalization is uncertain. Depending on the person’s physique and how the person sets up the sensors, the threshold may change. Therefore, detection methods that generalize well to new users and situations should be investigated. One such method is using machine learning algorithms that can adapt to unseen data.

Currently, not much research has been done on FHP detection using machine learning. One machine learning solution to FHP detection uses seven features for processing data from a three-axis accelerometer, three-axis magnetometer, and neck angle measured from a camera (Han, Jang, & Yoon, 2019). Although the solution was able to detect FHP with high accuracy, the aspect of real-time feedback to users was not considered. Real-time feedback is important for users' compliance with medical solutions. In the study by Kim et al. (2011), bio-feedback helped decrease the percentage of time participants spent in FHP. Therefore, along with a generalizable method for FHP detection, how to convey the detection status to users should be considered.

From this, a new research question arises: how can the FHP detection algorithm be developed so that it both generalizes well and gives good real-time user feedback? To address this question, the current paper combines previous research to propose a new solution using a combination of machine learning and real-time visualizations. To narrow the research question, FHP while sitting at a desk is only considered.

II. Method

a. Measurement Criterion

For real-time visualizations of posture, it is important to choose a measurement criterion that is able to capture the dynamic movements in 3D space. One measurement criterion that is able to do so is quaternions. Quaternions are used to describe 3D orientations and rotations in space and are a 4D extension of complex numbers (Rose, n.d.). They are the preferred method for representing 3D rotations in computer graphics because other methods such as matrix rotations consume more memory, are slow in computation, and can suffer from what is known as gimbal lock (Serrano, 2015) where one degree of freedom (DoF) is lost when one axis overlaps with another (HSU, 2022).

$$\text{Complex: } x + yi$$
$$\text{Quaternion: } w + xi + yj + zk$$
$$\text{Normalized Quaternion: } (w, x, y, z)$$

Figure 2: General representation of quaternions where w, x, y, z $\in \mathbb{R}$ and I, j, k are mutually orthogonal imaginary unit vectors (Rose, n.d.). The w component represents the angle of rotation while the x, y, and z components represent the axis of rotation. In the current paper, each IMU sensor encodes normalized quaternions (w, x, y, z).

Although previous research show CVA is a reliable measurement criterion for FHP detection (Kim et al., 2011), quaternions may be a better method because they can provide greater flexibility in terms of visualizing posture and means for real-time feedback to users. Therefore, quaternions are chosen for the measurement criterion for FHP detection in the current study.

b. Data Collection

In accordance with the definition of FHP, it is hypothesized that in order to detect FHP, there should be at least two sensors placed on the two parts of the spine that are exaggerated during FHP (cervical and thoracic spine). Therefore, two sensors are used to obtain quaternion information to capture the movements of the two exaggerated spine parts. Among the sensors that record quaternion information data, two IMU Valedo sensors, a Medical Wearable that is already commercially used to help relieve back pain, are chosen for the current study (Hocoma, n.d.). They are chosen because they are small with a size of 42 mm x 32 mm x 16 mm (L x W x H), lightweight at only 18 grams each, and are powered by Bluetooth Low Energy (BLE) to obtain normalized quaternion information. Thus, not only will the users be comfortable wearing the sensors, but it will be simple to extract the quaternion information. The sensors are placed on the parts of the spine that are exaggerated during FHP (one sensor on the back of the neck and one sensor on the upper thoracic spine) with double-sided medical tape.

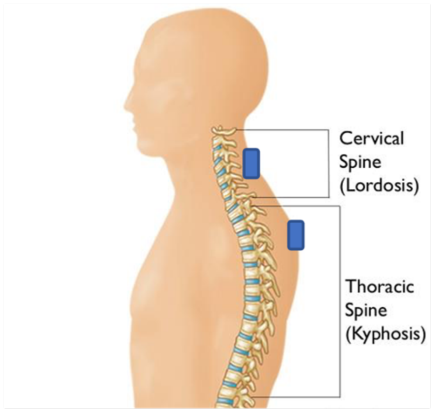

Figure 3: IMU sensor (in blue) placement used for data collection.

Each sensor has a 3-axis gyroscope, 3-axis accelerometer, and a 3-axis magnetometer to detect movement. With Bluetooth Low Energy (BLE), the normalized 4D quaternion orientation information of each sensor is extracted by subscribing to a notify characteristic Universally Unique Identifier (UUID) provided by Hocoma. Data is logged using the BlueSee application tool by Synapse which allows users to connect to and interact with BLE devices using UUIDs (Synapse Product Development, n.d.). BlueSee application was chosen because it provides easy connection and interaction with Bluetooth devices and also simultaneous reading of the two sensors which is important for sensor synchronization.

As a pilot study, only one subject is used for data collection. This is for initial tests to check whether there is any difference in the quaternion information data collected by the two sensors. For efficient labeling, data is collected for multiple sessions in either the ideal sitting posture (hereinafter referred to as good posture) or in variations of FHP. Variations of FHP include the head and/or body tilted to the side and the head resting on hand while in FHP. These variations are considered for better generalization and visualization of FHP because previous research only considers FHP in the upright position. Each session lasted approximately one to five minutes. The time duration for each session is short because sensors record information in microseconds.

Data is collected until a dataset of 10,000 data points (quaternion coordinates) is obtained with balanced class labels (5,000 data points in good posture and 5,000 data points in FHP). The dataset size was arbitrarily chosen as a starting point because it is a common dataset size for an average machine learning problem. The class labels are balanced to prevent overfitting the machine learning algorithms to one label.

The programming language Python (version 3.7.13) is used for data preprocessing, visualization, algorithm development, and implementation (Van Rossum & Drake, 2009).

c. Data Preprocessing

The notify characteristic UUID from the IMU sensors is encoded in little-endian format which contains information about the quaternion components (W, X, Y, Z), the timestamp in microseconds of the local sensor clock, and the status of the sensor. Upon subscribing to the notify characteristic UUID using

BlueSee, the raw data records the notebook (computer) timestamp along with the encrypted sensor data in hex format which is interpreted into human-readable data.

After interpreting the hex format of each sensor data, the data stream of the two sensors is examined for any error in data collection where the sensors are upside down by checking the signs of the quaternion coordinates. After making sure the sensor data are in the same orientation, the two sensors are synchronized for each trial. To synchronize the sensors, the first recorded notebook timestamp of each sensor is compared and matched. If the first notebook timestamp for sensor A is recorded earlier than sensor B, the start of sensor A data is redefined to match sensor B data, excluding data collected at an earlier notebook timestamp. Then, the delta difference of the first recorded local sensor timestamps Δts of the two sensors at the matched notebook timestamp is calculated. Based on Δts, the sensor data are aligned by delaying the earlier signal.

To describe the difference in 3D rotations of the two quaternions, the relationship of the two quaternions, also known as the quaternion difference, is calculated. The quaternion difference is used for the model input instead of the two quaternions because the resulting quaternion will allow for simplification of the real-time feedback visualizations. With the synchronized sensor data, the quaternion difference is obtained by multiplying the first quaternion with the inverse of the second quaternion (anushak, 2018). The inverse of normalized quaternions is the quaternion conjugate (J., 2012). The resulting quaternion coordinate is used as the four features (w, x, y, z) of the model input. 80% of the final dataset is used for training while the remaining 20% is used for testing, which is a common train/test split distribution used in machine learning (T. D., 2020).

$$\Delta q = q_1 * q_2^{-1}$$
$$q^{-1} = q'$$
$$\Delta q = q_1 * q_2'$$

Figure 4: Quaternion difference equation where q^{-1} is the inverse of the quaternion and q' is the conjugate of the quaternion.

d. Data Visualization

Before model selection, data is visualized for a better understanding of the dataset. For data visualization, the Python plotting library Matplotlib (version 3.5.2) is used (Caswell et al., 2022).

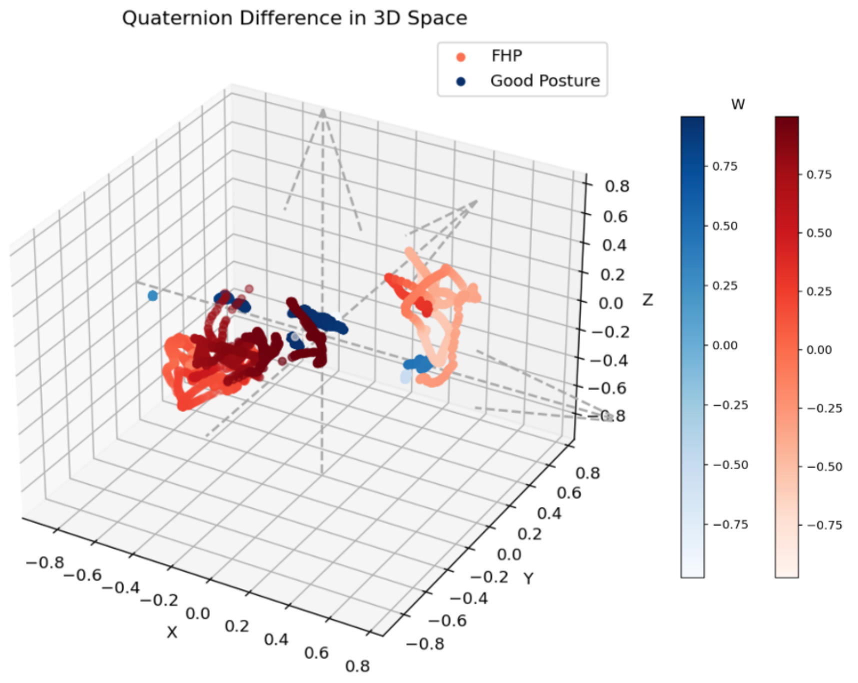

Figure 5: Quaternion difference of the two IMU sensors visualized in 3D space.

When data is visualized in 3D space, the quaternion difference data of good posture form small clusters near the X axis with little variation. On the other hand, there are two big clusters of FHP data that are in general, higher in variability in all four components of the quaternion difference. Because it is difficult to interpret 4D data in 3D space, the quaternion components are further visualized in 2D space, using the six possible combinations of quaternion component pairings.

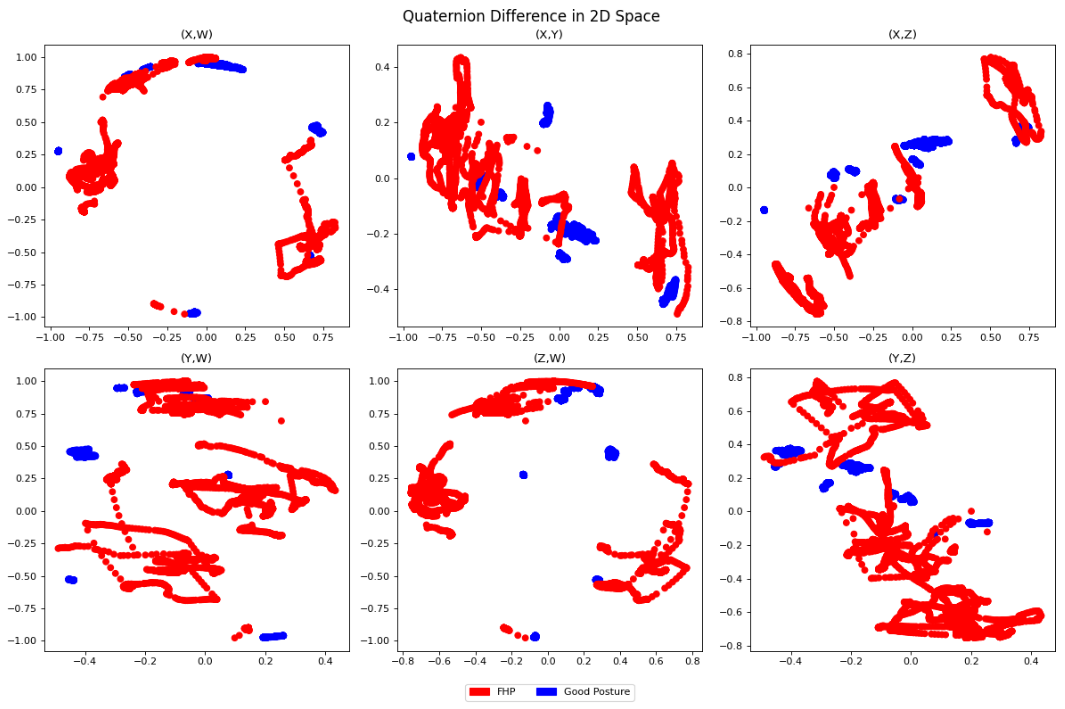

Figure 6: Quaternion difference of the two IMU sensors visualized in 2D space using six of the possible combinations of four quaternion components.

From the visualizations of the quaternion difference pairings in 2D space, there are a few indications that can be inferred. First, the back-and-forth movement, which is the rotation around the y axis, is portrayed in the wide range of values of y in FHP. However, the range of values in good posture is narrow compared to FHP. This is because there are only small to no movements occurring in data collection of good posture. Second, it appears that the component pairings (x, y) and (x, z) have a similar correlation in both good posture and FHP, while the y and z component relationships differ.

e. Algorithm

For the classification algorithm, the machine learning Python package Scikit-Learn (version 1.0.1) is used (Pedregosa et al., 2011). Two classification models are tested: Support Vector Machine (SVM) and Decision Tree. These models are tested because it was shown that they were able to successfully detect FHP in previous research (Han, Jang, & Yoon, 2019). SVM tries to find the optimal hyperplane to separate the data into its respective class (Pupale, 2018). In binary classification, the SVM algorithm finds support vectors which are the points from both classes that are closest to the boundary line. The distance between the boundary line and the support vectors is referred to as the margin. During training, the model tries to maximize this margin. Among the kernels that are used with SVM, the most general Radial Basis Function (RBF) kernel for non-linear data is used. The RBF kernel computes the similarity (distance) of the support vectors (Sreenivasa, 2020).

$$K(X_1, X_2) = exp(-\frac{||X_1 - X_2||^2}{2\sigma^2})$$

Figure 7: Radial Basis Function (RBF) kernel equation used in non-linear SVM model, calculating the similarity of two points (Sreenivasa, 2020). The numerator represents the squared Euclidean distance between points X_1 and X_2, and the hyperparameter variance σ is tuned accordingly.

There are two main hyperparameters of SVM using RBF Kernel: gamma and C. Gamma is the inverse of the radius of influence of the support vectors. As gamma increases, the radius of influence decreases, and the model tends to overfit. C is a regularization parameter that controls the tradeoff between classification accuracy and margin maximization. As C increases, the margin size decreases and accuracy increases, but overfits if the value is too large (RBF SVM parameters, n.d.).

Based on the validation accuracy as a function of gamma and C provided as an example by Scikit-Learn (RBF SVM parameters, n.d.), a range of gamma and C values were selected, and a grid search cross validation was performed to find the best parameter values. Because the dataset size used in the current research is small, low values of both gamma and C were selected to prevent overfitting.

Decision Trees are another popular machine learning classification method, constructed of nodes and branches (Z. A., 2020b). At each node, a feature is evaluated according to certain metrics defining the impurity of a node to split it into branches. Two commonly used metrics for categorical decision trees are the Gini and Entropy index. Gini index value with the interval [0, 0.5] indicates the frequency of an element being mislabelled when it is labeled at random. The optimum split is chosen for the lowest Gini index and the process of splitting ends when the Gini index is 0. The same concept applies to the entropy index where the entropy value with the interval [0, 1] indicates the amount of disorder in a group. Gini index is selected because it is computationally faster than entropy with similar results (Aznar, 2020).

$$GiniIndex = 1 - \sum_j p_j^2$$

Figure 8: Gini index equation used in categorical decision trees for splitting the nodes into branches where p_j is the probability of class j (Aznar, 2020).

There are three main hyperparameters of Decision Trees: the maximum depth of the tree, the minimum number of samples required to split a node, and the minimum number of samples required to be a leaf node (node with no child nodes) (Mithrakumar, 2019). Increasing the maximum depth of the tree will make the model more complex and can cause overfitting if it is too deep. For the minimum number of samples required to split a node or for it to be a leaf node, underfitting can occur if the number is too high (Fraj, 2017). Therefore, a range of small-valued Decision Tree hyperparameters was selected, and a grid search cross-validation was performed.

The following figure shows the overall summarized pipeline from data collection to algorithm implementation.

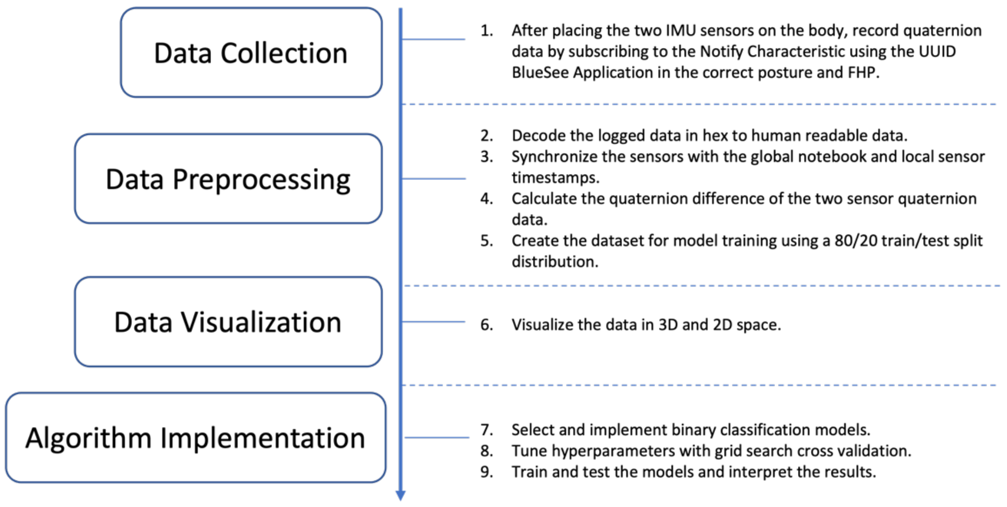
Data Collection
1. After placing the two IMU sensors on the body, record quaternion data by subscribing to the Notify Characteristic using the UUID BlueSee Application in the correct posture and FHP.
Data Preprocessing
2. Decode the logged data in hex to human readable data.
3. Synchronize the sensors with the global notebook and local sensor timestamps.
4. Calculate the quaternion difference of the two sensor quaternion data.
5. Create the dataset for model training using a 80/20 train/test split distribution.
Data Visualization
6. Visualize the data in 3D and 2D space.
Algorithm Implementation
7. Select and implement binary classification models.
8. Tune hyperparameters with grid search cross validation.
9. Train and test the models and interpret the results.

Figure 9: Overall project pipeline from data collection to algorithm implementation.

III. Results

The results of the best parameter SVM model obtained from the grid search cross-validation (C = 1, gamma = 0.1) revealed an overfitting phenomenon with test accuracy of 1.0. Lowering the values of C and gamma did not fix the overfitting problem. This may indicate several problems in either the model selection and/or dataset. The results of the best parameter Decision Tree model obtained from the grid search cross-validation (maximum depth of tree = 4, minimum number of samples for leaf node = 1, minimum number of samples to split node = 2) showed high test accuracy of 0.889, and did not appear to be overfitting.

	SVM	Decision Tree
Accuracy	1.00	0.89

Figure 10: FHP detection accuracy scores for the two classification models SVM and Decision Tree.

For a more detailed analysis of the Decision Tree model, the confusion matrix is calculated which reveals that 222 data points for good posture were misclassified. This may be a result of good posture data having less variability than FHP data.

	Decision Tree	
	False	True
Negative (FHP)	1000	0
Positive (Good Posture)	222	778

Figure 11: Confusion matrix of the Decision Tree model predictions.

For user feedback visualization, the two sensors' quaternion difference at a certain point in time and space is plotted and shown in real-time. The point is labeled as FHP or good posture according to the classification results from the trained model. With this kind of visual feedback, the user can check their posture and correct it whenever they see that they are in FHP. The results from the real-time demonstrations revealed that while the Decision Tree model was able to classify better than SVM, both SVM and Decision Tree models showed overfitting behavior on the training data and were not able to accurately and reliably predict the real-time transitions.

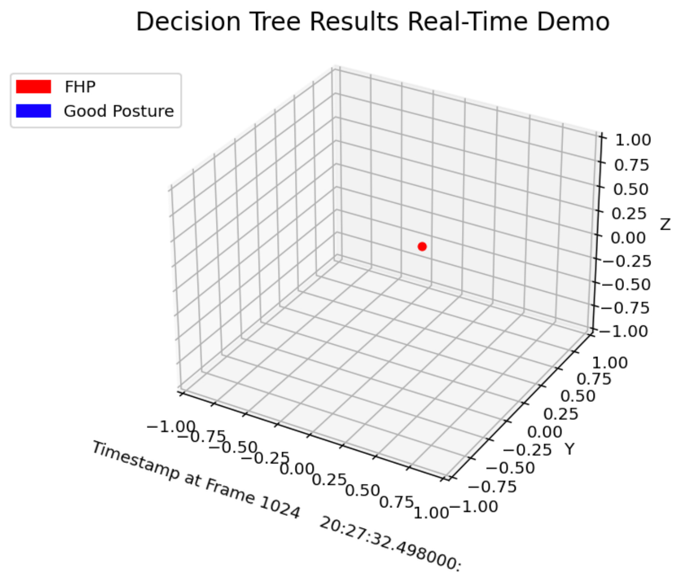
Decision Tree Results Real-Time Demo
FHP
Good Posture
Z
Y
Timestamp at Frame 1024 20:27:32.498000:

Figure 12: Screenshot of the real-time feedback animation displayed to the user as they are wearing the sensors. A point in space represents the quaternion difference of the two streams of sensor data. This point moves position and changes color in time, depending on the quaternion component values and classification results from the model.

IV. Discussion

The aim of the current study was to provide a new solution for FHP detection, focusing on generalization and real-time user feedback visualization. Results of the study revealed that although the test accuracy was high for both the SVM and Decision Tree models, the real-time user feedback visualization indicated that both models overfitted to classify data as mostly good posture. Therefore, while the quaternion information of the sensors helped give real-time user feedback through visualization, the classification algorithms using quaternion information to differentiate between FHP and good posture were not generalizable as intended.

There are several limitations of the current study that could have led to such results. The most significant limitation is the lack of variation and size in the training data since data was collected from only one subject. This resulted in high test accuracy for both the SVM and Decision Tree models but overfitted to new test data in the real-time demo. Because the body structure is different from person to person, it is important to collect data from more participants. Furthermore, good posture and FHP data were collected separately in different trials. This resulted in intermediate postures between good posture and FHP absent from training. This lack of variety in data can explain why the intermediate postures between FHP when transitioning from the good posture in the demo results are not considered, and the models have a hard time predicting correct results after movement occurs. In future studies, data should be collected such that the quaternion data in all postures in-between good posture and FHP are recorded in the same session.

Apart from the dataset variation and size, another reason why the models may have shown weak performance is possible logical errors in code that can occur in various steps of the project pipeline. There may be an error in decoding the hex format data into human-readable data, sensor synchronization, and/or data leakage. Although the sensor status was not considered in this study, it should be investigated whether the sensor status is causing unusual behavior during data collection. Overall, before implementing algorithms, it is important that the dataset is accurately portraying the intended information. These steps should be carefully considered in future research.

Comparing the current results of the SVM and Decision Tree models, simpler models may be a better model selection choice due to the small number of features used as vector inputs and the small number of classes. Simpler models can decrease the overfitting tendencies and help generalize better to real-time data.

Other limitations include the sensor placement location and the number of sensors used for FHP detection. Although the location and number were chosen based on the definition of FHP, it may not be the most optimal. Because the sensor placement location for both sensors is at the back of the body, it can be difficult to attach the sensors by oneself. Therefore, future studies should investigate whether there is a better sensor placement location for FHP and how many sensors are necessary for reliable results.

In terms of real-time feedback for users, much can be improved upon. For one, the classification is not actually done in “real-time” because the trained model is processing pre-recorded data due to the technical nature of the BlueSee application used for reading the data from the sensors. Software should be developed that can feed the sensor data directly into the models to actualize the “real-time” element of the feedback. Furthermore, other bio-feedback such as audio or haptic feedback could be incorporated to help prevent FHP. Apart from real-time feedback, accumulated data statistics can be logged and displayed for the user, revealing information such as the amount of time the user spent in FHP while wearing the sensors. There are still many unanswered questions in the area of FHP detection and prevention, and research on the optimal solution for FHP detection using real-time user feedback visualizations should be done.

V. References

American Academy of Orthopaedic Surgeons. (n.d.). Kyphosis (Roundback) of the Spine. OrthoInfo. Retrieved from https://orthoinfo.aaos.org/en/diseases--conditions/kyphosis-roundback-of-the-spine/

apushak. (2018, January 19). Get the difference between two quaternions and add it to another quaternion [Online forum post]. Unity Forum. https://forum.unity.com/threads/get-the-difference-between-two-quaternions-and-add-it-to-another-quaternion.513187/

Aznar, P. (2020, December 2). Decision Trees: Gini vs Entropy. Quantdare. https://quantdare.com/decision-trees-gini-vs-entropy/#:%7E:text=In%20this%20post%2C%20we%20have,entropy%20criterion%20are%20slightly%20better

Caswell, T. A., Droettboom, M., Lee, A., Sales De Andrade, E., Hoffmann, T., Klymak, J., Hunter, J., Firing, E., Stansby, D., Varoquaux, N., Nielsen, J. H., Root, B., May, R., Elson, P., Seppänen, J. K., Dale, D., Lee, J. J., McDougall, D., Straw, A., & Ivanov, P. (2022). Matplotlib (3.5.2) [Python library]. Zenodo. https://matplotlib.org/

Fraj, M. B. (2017, December 20). InDepth: Parameter tuning for Decision Tree. Medium. https://medium.com/@mohtedibf/indepth-parameter-tuning-for-decision-tree-6753118a03c3

Griegel-Morris, P., Larson, K., Mueller-Klaus, K., & Oatis, C. A. (1992). Incidence of Common Postural Abnormalities in the Cervical, Shoulder, and Thoracic Regions and Their Association with Pain in Two Age Groups of Healthy Subjects. Physical Therapy, 72(6), 425–431. https://doi.org/10.1093/ptj/72.6.425

Han, H., Jang, H., & Yoon, S. W. (2019). Novel Wearable Monitoring System of Forward Head Posture Assisted by Magnet-Magnetometer Pair and Machine Learning. IEEE Sensors Journal, 20(7), 3838–3848. https://doi.org/10.1109/jsen.2019.2959817

Hocoma. (n.d.). Valedo – back pain therapy with Xsens Technology. Xsens. Retrieved from https://www.xsens.com/cases/valedo-back-pain-therapy-xsens-technology

HSU, I.-M. (2022, April 19). Gimbal lock explanation and how to work with it. Medium. Retrieved from https://blog.devgenius.io/gimbal-lock-explanation-and-how-to-work-with-it-3b61c0bac024

J. (2012, June 25). Understanding Quaternions. 3D Game Engine Programming. https://www.3dgep.com/understanding-quaternions/

Kang, J. H., Park, R. Y., Lee, S. J., Kim, J. Y., Yoon, S. R., & Jung, K. I. (2012). The Effect of The Forward Head Posture on Postural Balance in Long Time Computer Based Worker. Annals of Rehabilitation Medicine, 36(1), 98–104. https://doi.org/10.5535/arm.2012.36.1.98

Kim, T., Chen, S., & Lach, J. (2011). Detecting and Preventing Forward Head Posture with Wireless Inertial Body Sensor Networks. 2011 International Conference on Body Sensor Networks, 125–126. https://doi.org/10.1109/bsn.2011.41

Lee, J., Cho, E., Kim, M., Yoon, Y., & Choi, S. (2014). PreventFHP: Detection and warning system for Forward Head Posture. 2014 IEEE Haptics Symposium (HAPTICS), 295–298. https://doi.org/10.1109/haptics.2014.6775470

Mithrakumar, M. (2019, November 11). How to tune a Decision Tree? - Towards Data Science. Medium. https://towardsdatascience.com/how-to-tune-a-decision-tree-f03721801680

NHS. (n.d.). Common posture mistakes and fixes. Retrieved from https://www.nhs.uk/live-well/exercise/strength-and-flexibility-exercises/common-posture-mistakes-and-fixes/#:~:text=The%20idea%20is%20to%20keep,feet%20about%20hip%20distance%20apart

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python (1.0.1) [Python package]. Journal of Machine Learning Research, 12(85), 2825-2830. https://scikit-learn.org/stable/

Pupale, R. (2018, June 16). Support Vector Machines (SVM) — An Overview. Medium. https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989#:%7E:text=SVM%20or%20Support%20Vector%20Machine,separates%20the%20data%20into%20classes

RBF SVM parameters. (n.d.). Scikit-Learn. https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

Rose, D. (n.d.). Rotation quaternions, and how to use them. Retrieved from https://danceswithcode.net/engineeringnotes/quaternions/quaternions.html

Serrano, H. (2015, January 12). Quaternions in Computer Graphics. Harold Serrano. https://www.haroldserrano.com/blog/quaternions-in-computer-graphics

Silva, A. G., Punt, T. D., Sharples, P., Vilas-Boas, J. P., & Johnson, M. I. (2009). Head Posture and Neck Pain of Chronic Nontraumatic Origin: A Comparison Between Patients and Pain-Free Persons. Archives of Physical Medicine and Rehabilitation, 90(4), 669–674. https://doi.org/10.1016/j.apmr.2008.10.018

Sreenivasa, S. (2020, October 12). Radial Basis Function (RBF) Kernel: The Go-To Kernel. Medium. https://towardsdatascience.com/radial-basis-function-rbf-kernel-the-go-to-kernel-acf0d22c798a

Synapse Product Development. (n.d.). Version (1.66). BlueSee. Retrieved from https://www.synapse.com/bluesee.

T.D. (2020, January 31). The 80/20 Split Intuition and an Alternative Split Method. Medium. https://towardsdatascience.com/finally-why-we-use-an-80-20-split-for-training-and-test-data-plus-an-alternative-method-oh-yes-edc77e96295d

Van Rossum, G., & Drake, F. L. (2009). Python (3.7.13) [Computer software]. CreateSpace. https://www.python.org/

Z.A. (2020b, March 8). Decision Trees Explained. Medium. https://towardsdatascience.com/decision-trees-explained-3ec41632ceb6

VI. Links to resources

- Gitlab Repository: https://gitlab-mi.informatik.uni-ulm.de/jnj06/iomt/-/tree/main/ (only the Final_Submission folder should be considered for assessment)