Automated Spinal Segmentation and Remote Monitor Calibration for Surgical Assessment

Final Project Report

EN.601.456 Computer Integrated Surgery II

Group 8:
Damiano Marsilli, Arijit Nukala, Jonathan Young

Abstract

The spine is a mobile structure that allows the body to bend, twist, and lift. However, spine surgeons currently do not have a method of quantitative motion analysis for diagnosis and surgery pre-planning. Current spinal imaging techniques involve X-Rays, CT scans, and MRIs, which provide static images that are not representative of the spinal curvature of the patient during routine daily activities. As a result of limited imaging options, spinal fusion surgeries are often improperly prescribed. Our mentor's team, CurveAssure, prototyped a device that can be placed on the back of patients to give doctors an insight into the behavior of the spine when the patient is moving over a longer period of time. However, to provide actionable insights for doctors, sensor readings must be augmented with spinal imaging techniques to produce a personalized quantitative motion model.

I. Motivation

Current methods of accessing eligibility for spinal fusion surgeries are complex and unclear. Up to 1.62 million instrumented spinal fusion procedures occur each year [1]. However, these surgeries suffer from complication and revision rates as high as 50% and 36% respectively [2]. One contribution to this shocking statistic is that these procedures are often improperly prescribed. 17% of spinal fusion surgeries are performed on patients who should not have been recommended for the procedure in the first place [3]. In fact, the variability regarding the eligibility assessment can drastically change for patients. One study showed that even expert assessments can drastically change for a given patient from one day to another. These patients often suffer from negative outcomes as well as costly medical bills that are unnecessary.

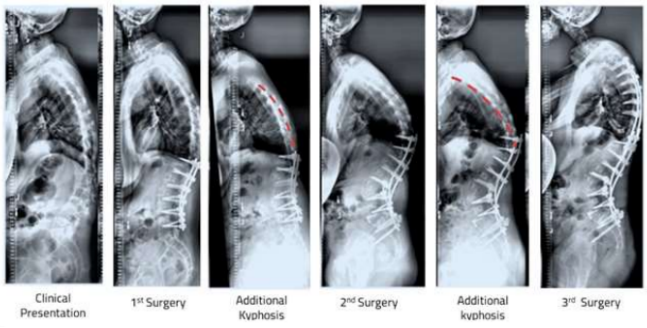

Figure 1 - Patient With multiple revisions [4]

In the example above, a patient had to undergo multiple surgeries, as the first two surgeries did not produce the desired outcome.

Therefore, this project aims to decrease the improper prescription of spinal surgery, by providing doctors with a dynamic view of the patient's spine. By placing the sensors of these patients and utilizing our model and transfer function, our team is able to produce a comprehensive image of the behavior of the spine, thus providing doctors with a better means of assessing if their patient is eligible for a procedure. Moreover, by deploying our solution to the cloud, our team ensures the solution is accessible to doctors around the world without the need for specialized computational equipment.

II. Prior Work

Our mentor’s team, CurveAssure, has acquired Delsys sensors for the purposes of acquiring telemetry from keypoints along the spine. They’ve also produced a method of collecting information from patient movements in various ways to gain a holistic picture of their spine. Current methods of assessing the spine are static, which provides an incomplete picture of the

spine. Therefore, their developed method of spinal movements cohesively brings together a holistic comprehension of spinal behavior. Using the protocol, CurveAssure acquired both video footage and sensor readings of patient spinal movement. From this, they have left the task to this CIS team to make use of the video footage and sensor readings to perform calibration for them.

III. Goals

Our project aims to augment Curve Assure's spinal IMUs with computer vision techniques. Our project has multiple phases. The first phase entails the development of a method capable of estimating a series of spinal key points on the patient using monocular video. A pre-trained convolutional neural network (CNN) will be augmented with a spinal keypoint prediction head, which will be trained using a transfer learning approach to produce spinal key points. The second phase corresponds to the development of a transfer function that maps IMU data to spinal keypoint angles. This transfer function will be used to correlate IMU sensor data with the estimated spinal curve, producing a quantitative model of the patient's movement in 3D space. The spinal key points from phase 1 will serve as initial calibration estimates. Lastly, the last phase entails deploying our method on the cloud and developing a pipeline that enables users to upload video and sensor data to the cloud and receive a quantitative model of the patient's spine. An overview of our approach can be found in Figure 2.

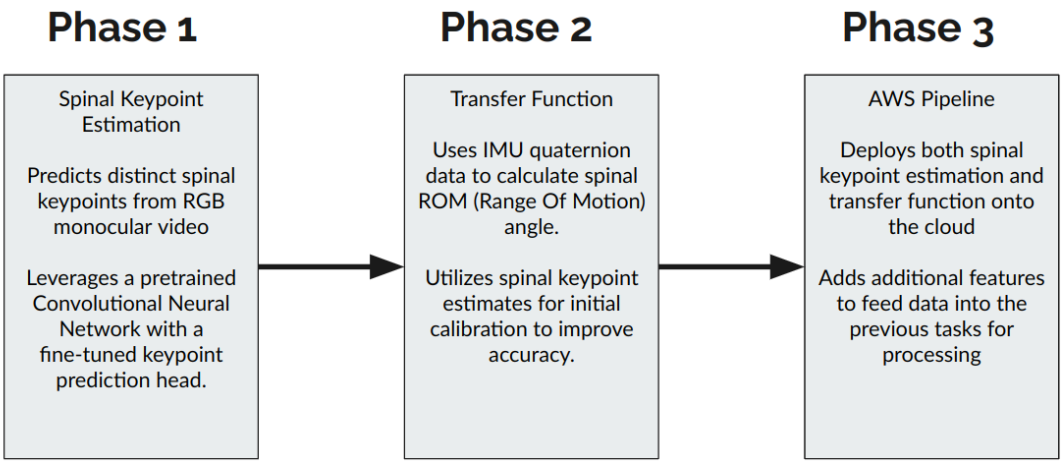

Figure 2 - Overview of Goals

IV. Technical Approach

A. Spinal Keypoint Estimation

The task of spinal keypoint estimation involves the consistent detection, and tracking, of spinal key points on a patient from monocular video. Formally, given a video, represented as a tensor of shape $V = (F, C, H, W)$, where F is the number of frames, C is the number of channels (3 for RGB, 1 for grayscale), and H, W are the height and width of the frames respectively. Then, we must produce a function f such that $f(V) = K$, where K is a tensor of shape $(4, F, 2)$, corresponding to 4 key points found for each frame for both axes (u, v) in image coordinates. A schematic of the module can be found in Figure 3

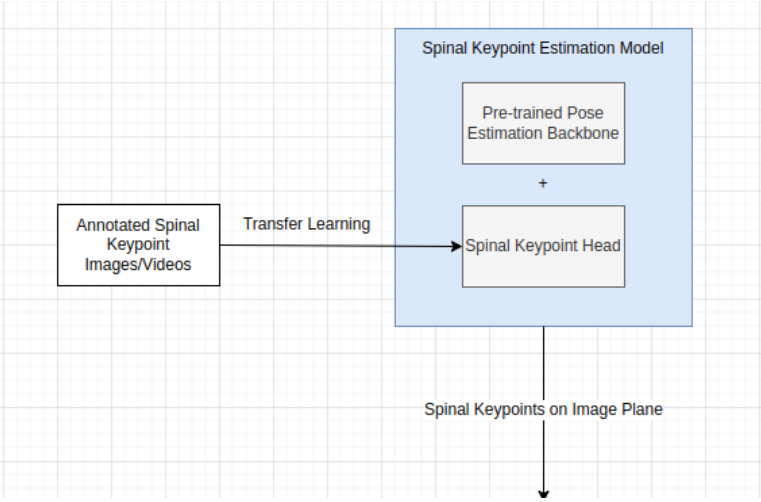

Figure 3 - Spinal Keypoint Estimation Module Schematic

To produce f, we leverage deep learning and the paradigm of transfer learning. Specifically, we will take a ResNet50 convolutional neural network pre-trained on ImageNet[4] to serve as our feature-detection backbone. The backbones will provide a latent video representation that encapsulates information useful to the task of pose estimation. Then, we will concatenate our own spinal-keypoint head, which will be used for the downstream task of spinal keypoint prediction. The spinal keypoint head consists of two fully connected layers, with a ReLU non-linearity in between.

During training, we freeze the ResNet backbone and train our spinal keypoint in a standard supervised setting using gradient descent [5]. The model is trained using mean-squared error loss [6]. To evaluate training, we will be monitoring both training loss and validation loss, to ensure we are not overfitting to the training data. At the end of training, we perform a qualitative analysis on the key points produced.

B. Transfer Function

The transfer function utilizes patient IMU data in conjunction with estimated spinal key points to produce clinically relevant spinal range of motion (ROM) data. The transfer function takes inputs from the computer vision system's predicted 3D spine key points, represented by the tensor in the shape $(4, F, 2)$ as previously mentioned, and the IMUs orientation data, represented by the quaternion $q_t = (q_1, q_2, q_3, q_4)$, to compute estimated joint angles for each of the four major spinal regions in the sagittal and coronal planes.

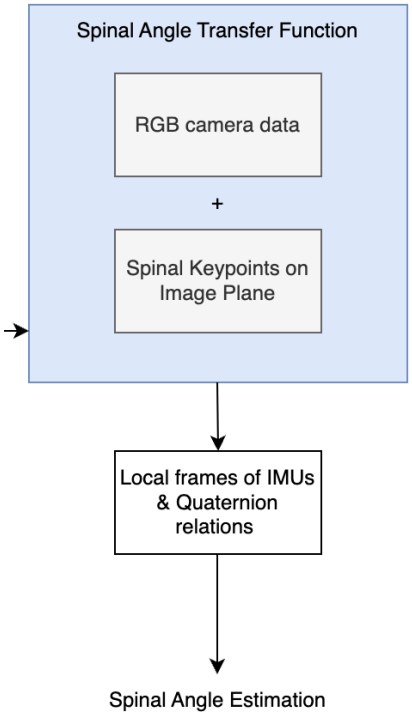

Figure 4 - Spinal ROM Transfer Function

We developed a simple transfer function that takes IMU orientation data as an input to calculate angles between each of the spine segments. We can calculate the plane of movement given IMU orientations in that plane with the following calculations:

1. For the cervical IMU placed on the neck, reorient each axis to match those of the other IMUs
2. For the other IMUs placed on the spine, consider all orientations and optimize the accuracy by using the constraints given
3. Compute relative angles between the IMUs using the transfer function

The transfer function uses initialization and calibration equations described in Franco, L. et al's paper to create constraints for the angle estimation from the IMU data.

C. AWS - Amazon Web Services, Pipeline

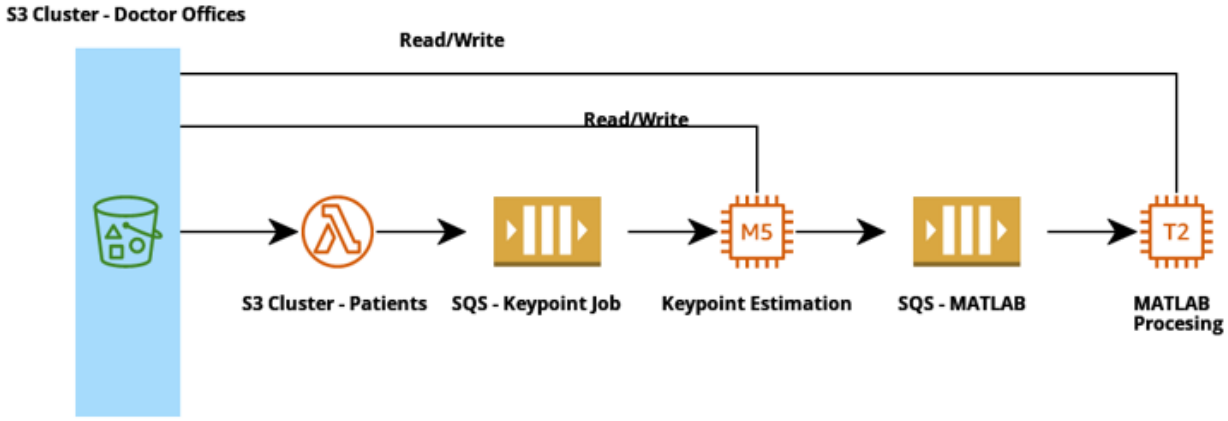

Figure 5 - AWS Pipeline (MATLAB)

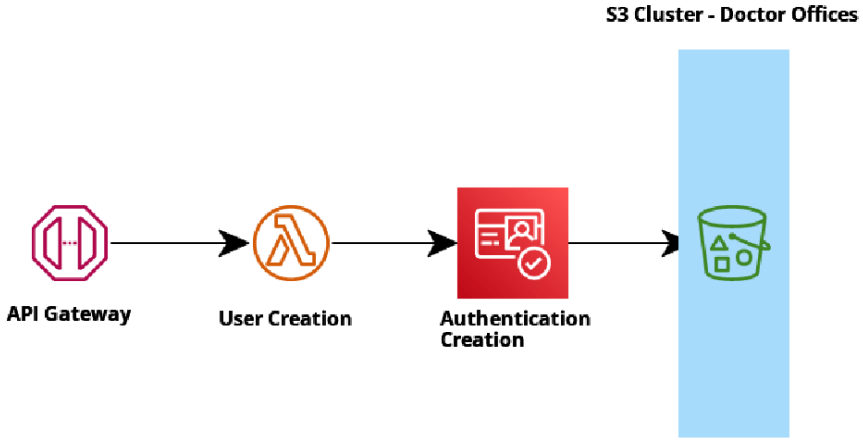

Figure 6 - AWS Pipeline (User Creation)

The task of the Amazon Web Services (AWS) pipeline is to enable both patients and doctors to process their information on the cloud without the need for computers to be utilized locally on hand. This enables flexibility for both the patient and the doctor, where patients are directly able to upload their data with ease and doctors are likewise able to retrieve that data efficiently. Patients are having their data recorded at home, which requires them to upload their data to the cloud. The specific purpose of cloud utilization for this project is to host and integrate the spinal keypoint estimation and the transfer function.

The pipeline currently follows the diagram above. First, users are able to upload their data to an S3 bucket. This S3 bucket, however, has a number of permissions that only allow it to be accessed by a patient's user group on AWS Cognito. These user groups represent a doctors

office, with each user in that group being a patient. The Cognito groups are defined under the same identity pool - which is also designed to be utilized later for a full stack or application based deployment. *Figure 6*

A lambda function is triggered to notify an SQS queue for an incoming job. This lambda function can receive a number of S3 buckets as triggers, where a trigger indicates a new file being uploaded. The lambda function is utilized to check if the correct pairs of file, both the IMU and mp4 files, have been uploaded. Once it has been confirmed that the pairs of files have been uploaded, a job notification is sent on a queue. This queue contains all the jobs to be processed by the EC2 instances. *Figure 5*

An EC2 then handles the transfer function and will grab jobs at a specified time from the SQS queue. This requirement is defined by CurveAssure, where jobs will be processed during a specific time. The EC2 instance then grabs the jobs of the SQS queue. A singular EC2 instance is utilized for the time being because of MATLAB license limitations. It is more preferable to use a scalable cluster of EC2s however. Once the files have been processed, another SQS queue will receive notifications that a new job has arrived.

A similar process occurs for the EC2 handling the keypoint estimation. The EC2 will grab the job from the queue that the MATLAB EC2 deposited its information in, process it, and drop it in the same S3 bucket as the input.

It should be noted that this pipeline is highly modular. One can also attach events such as SNS topics for emailing when a job is completed, or a scalable EC2 group for faster processing. This pipeline is currently not autoscaling because of the MATLAB limitation. This process is in place, and is awaiting implementation of the transfer function to be converted into python to be deployed. *Figure 6*

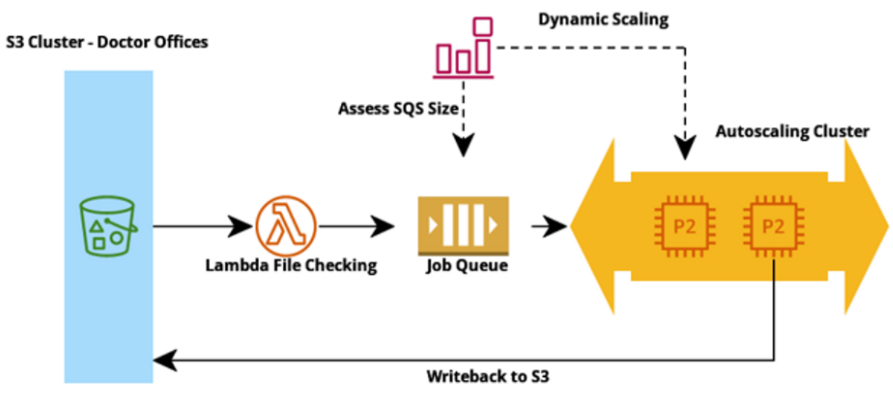

Figure 7 - AWS Pipeline (Scaleable)

V. Key Activities & Deliverables

Our final deliverables are shown in the table below.

	Deliverable	Done?
Minimum (4/7)	Linear transfer function for translating IMU data to estimated spinal joint angles	**Done**
	Spinal estimation model capable of detecting key points on the spine from monocular video	**Done**
	Detailed documentation of interface and code structure between the two major components (transfer function & spinal estimation)	**Done**
Expected (4/28)	Bounding box model that can identify and track the IMU sensors in monocular RGB video.	**Done**
	Amazon Web Services (AWS) workflow and infrastructure for model training and testing for future development	**Done**
	Detailed procedure for clinical use of the product including patient orientation, patient movement, and sensor location designed to produce the most descriptive spinal model of the patient	**Done**
Maximum (TBD)	AWS pipeline that allows for the remote upload of patient videos and IMU data and performs analysis in the cloud, returning the spinal model's output	**Done**
	Detailed procedure and method for sensor calibration, so as to reduce the impact of sensor drift and position prediction errors over the 48 hour data collection period	**Ongoing**
	~~Software that creates a 3D volumetric model of the patient, enabling the demonstration of their posture/activity to surgeons utilizing either Blender or Maya~~	**Removed**

A. Adherence to Deliverables

We note that some of our final deliverables do not represent our initial deliverables. First, the spinal curve estimation model has been replaced with a bounding box model that can identify and track the IMU sensors in the video frame. Through careful discussion with our mentors, it was concluded that this approach would be more valuable in calibrating the patients spinal range of motion by providing an estimate of the location of the IMU in space. Moreover, the 3D volumetric model rendering was removed from the maximum deliverables as our mentors felt our efforts were better served calibrating the IMUs and developing the necessary AWS pipeline. Overall, we are satisfied with our progress, having achieved all of our expected deliverables and made significant progress on our maximum deliverables.

B. Progress Evaluation

Overall, our team's progress was mostly on schedule. There were little obstructions or unanticipated actions that our team had to deal with. In the beginning there were some hindrances, dependencies wise, in terms of getting data on time and getting our AWS credits and accounts ready in time to start our work. However we quickly adjusted to those setbacks and were able to resume being on schedule. Further dependencies, such as acquiring patient data, were done in a timely manner.

VI. Experimental Setup

Our team tested this system by uploading sample patient validation data, or performing various unit tests of the cloud system.

A. Spinal Keypoint Testing Setup

Spinal keypoint testing was done locally on a personal workstation with GPU access. The trained models were instantiated in a Python script. CUDA was used to perform inference on a GPU to ensure shorter testing times.

The trained spinal keypoint model was evaluated to ensure keypoint predictions were reasonably accurate. To this end, a subset of labeled data, referred to as the test dataset, was held out during training. This data was used to validate the accuracy and generalizability of our model. As a performance metric, we use Mean-Squared-Error between the predicted keypoints and the ground-truth annotations.

B. Transfer Function Testing Setup

Transfer function testing was done locally on a workstation with Matlab access. Testing was done using test data provided by Franco, L et. al. Results confirm that spinal range of motion

estimation completed via the transfer function is close to the ground truth range of motion data computed with infrared trackers.

Future work can analyze the performance of the transfer function using collected patient data using both IMU and IR data to analyze the accuracy of our transfer function with the optimization constraints introduced by the spinal keypoint data from our model.

C. Cloud Testing Setup

Cloud testing was done through uploading patient data and at different edge cases, such as if the patient failed to upload both sensor and video files required for the pipeline, to analyze the effectiveness and error handling of the system. These edge cases were done by creating sample scripts that simulate users uploading their data or logging into the system.

To analyze the effectiveness of our scaling algorithm, our team flooded the system with 450 patient jobs, and took note of the scaling behavior of the virtual instance cluster. Scaling is important in cloud pipelines to handle large volumes of information that is uploaded to the system. Scale testing is done through a python script designed to simulate an influx of patient data being uploaded.

VII. Results

A. Spinal Segmentation Results

We trained multiple model architectures in attempt to find the most performant spinal keypoint estimation models. Specifically, we modified the hidden dimension of our keypoint prediction head. Increased hidden dimensions result in more complexity, and thus better performance. It should be noted, however, that increasing the model complexity adds to the running time of our model. We show the best accuracies for each of the models below. As mentioned previously, the metric used is mean-squared error (MSE).

Hidden Dim.	MSE (25 epochs)	MSE (50 epochs)
0	38.5	34.1
250	18.2	14.5
500	12.3	9.6
1000	8.4	7.3

As shown in our results, an increased hidden dimension does improve performance substantially, but with diminishing returns as the dimension increases. We also show our training loss for the 1000 dimensional model below:

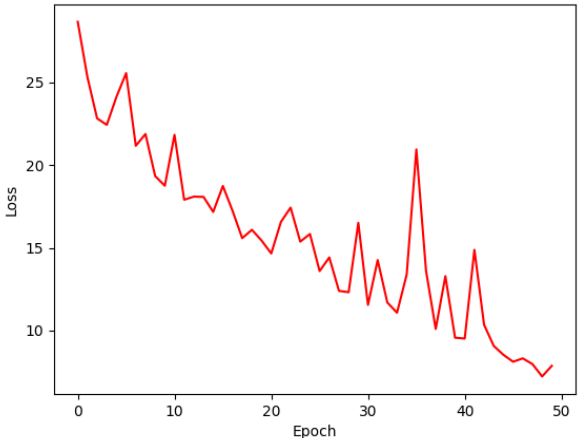

Figure I - Keypoint Estimation Model Loss Curve

B. Transfer Function Results

The transfer function result ROM files are documented in Github. Using the given Matlab file, users will be able to input test data as a .csv file and compute ROM.mat outputs for analysis from the function.

The current state of the transfer function only enables kinematic constraints without the spinal keypoint model optimization. However, the spinal keypoint labels can be passed into the code for later implementation.

C. AWS Pipeline Results

The AWS pipeline, figure 7, was successful in processing information from patients and providing results in a timely and robust manner. Users were easily able to upload their files and obtain their results. There was no failure reported with the EC2 (remote instances), when they attempted to process the data.

The AWS pipeline, figure 7, that was focused on scaling was highly successful as it was able to recognize when more instanced needed to be scaled up, as well as decreasing them when there was no longer the need to utilize the instances. Scaling occurs when's theres a large influx of patient jobs that need to be processed. Below is an increasing scaling example:

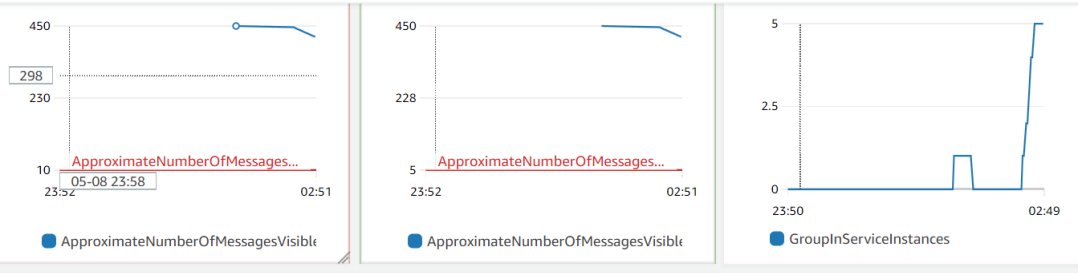

Figure 8- AWS Pipeline (Scaling)

From left to right, number of jobs in the queue, number of jobs in the queue, number of instances running (scaled)

Overall, our team was very pleased with the performance and handling of a variety of scenarios by our pipeline.

VIII. Significance

Our project laid down the foundation for CurveAssure to continue working on our existing work to develop a system to obtain a model of the behavior of the spine. The documentation that our team developed, alongside the existing code and other related infrastructure, will enable them to continue on our work.

Our keypoint estimation model was able to recognize key points on the spine, as well as provide guidance and direction for Curveassure for collecting more data for keypoint estimation alongside the techniques required to produce such a model. Though the accuracy can be improved, the model nonetheless is already there, and requires additional data to strengthen its accuracy.

Our transfer function laid down the foundation of methods to compute quaternions and other positional data for calibration, and methods to merge both the sensor data and the spinal keypoint data together for calibration.

Lastly, our cloud pipeline laid down the foundation for fast deployment of their product on a large scale. All infrastructure for data upload, processing, and computation have all been implemented. Furthemore, the associated documentation for these processes have also been written, enabling the cloud pipeline to be expanded. Due to the modular design of this pipeline, additional features can also be provided. A side project of this pipeline was user management and creation for patients and doctors alike. Though this system is not fully implemented, documentation and existing code provides a strong foundation for guidance on how to move forward with this area.

IX. Dependencies

A table of our full dependencies can be found in Figure 8. Overall, almost all of our dependencies were met, with the exception of labeled infrared data. As a result, we could not explore a calibration methodology that incorporated IR readings. However, our mentors acknowledged this. This data was not needed to run the transfer function and can be acquired later to measure accuracy of the transfer function.

	Dependency	Need	Status	Followup	Contingency Plan	Deadline
Software	Amazon Web Services (AWS) - Accounts	Need access to computing platforms	**Acquired**	N/A	Limited Deployment and Utilization of AWS using personal Free Tier Account	Mar 13th
	Pre-trained pose models	Backbones will be used to extract useful image features	**Acquired**	N/A	Project Necessity, no Contingency	Mar 6th
	AWS Credits	Need credits in order to run computing resources	**Acquired**	Contact Evan Haas to request credits	Limited Deployment and Utilization of AWS using Free Tier resources	Mar 13th
Footage for Testing and Validation	Videos for Training Data	Videos will be used to train our spinal keypoint head	**Acquired**	Contact Evan Haas for additional footage	Our team will film our own videos, using agreed upon specifications for filming	Mar 6th
	IMU Data for Training	Data will be used to fit our transfer function	**Acquired**	N/A	If we require additional IMU data, we will ask Evan Haas for some	Mar 6th
	IMU Data for Testing with Infrared Ground truth (Validation)	Data will be used to validate accuracy of our transfer function	**Not Acquired**	N/A	If infrared ground truth data is not available, we will manually measure the patient's spinal ROM	April 20th
	Video Data for Testing (Validation)	Videos will be used to validate & fine-tune hyperparameters for our spinal keypoint head	**Acquired**	Contact Evan Haas for additional footage	Our team will film our own videos, using agreed upon specifications for filming	April 20th

	Dependency	Need	Status	Followup	Contingency Plan	Deadline
Software	Amazon Web Services (AWS) - Accounts	Need access to computing platforms	**Acquired**	N/A	Limited Deployment and Utilization of AWS using personal Free Tier Account	Mar 13th
	Pre-trained pose models	Backbones will be used to extract useful image features	**Acquired**	N/A	Project Necessity, no Contingency	Mar 6th
	AWS Credits	Need credits in order to run computing resources	**Acquired**	Contact Evan Haas to request credits	Limited Deployment and Utilization of AWS using Free Tier resources	Mar 13th
	Labeled Data	Ground truth labels for the training dataset	**Acquired**	Await Evan for more data/AWS Mechanical Turk	Project Necessity, no Contingency	April 15th
	Labeled Data (for testing)	Ground truth labels for the validation dataset	**Acquired**	Await Evan for more data/AWS Mechanical Turk	Project Necessity, no Contingency	April 20th

Figure 8 - Dependencies Graph

X. Team

Our mentors consisted primarily of Evan Haase and Antony Fuleihan from the Center for Bioengineering Innovation and Design. In early discussions, Dr. Yazdi, Dr. Theodore, Dr. Williams and Dr. Khalsa were engaged for validation feedback. Our team consisted of Damiano Marsili, an undergraduate majoring in Computer Science and Mathematics, Arijit Nukala, an undergraduate majoring in Biomedical Engineering and Computer Science, and Jonathan Young, a MSE student majoring in Computer Science.

XI. Roles and Management Plan

The roles were divided among the three phases established above. Damiano Marsili led the development of the spinal keypoint estimation module, Arijit Nukala led the development of the transfer function, and Jonathan Young led the development of the AWS infrastructure. In the latter portions of the project, it became evident that the transfer function was more involved than anticipated, and Jonathan assisted Arijit with the task.

Our team met weekly with Evan Haase and Antony Fuleihan at 4:00 PM on Fridays via zoom. Additional meetings were held at the start of the project for further brainstorming and ideation. We did not feel the need to engage Dr. Theodore, Dr. Yazdi, Dr. Williams and Dr. Khalsa throughout the project, but we are thankful for their availability nevertheless.

Documentation was uploaded to the course wiki, with the cloud documentation being on the CurveAssure Notion page. Code specific documentation, alongside their corresponding code, can be found on the CurveAssure Github.

XII. Next Steps

Our pipeline and the deliverables achieved are a prototype and the beginning of the process for the CurveAssure project. We outline a few tasks outside of the scope of our project that serve as valuable next steps:

The keypoint model needs additional data in order to create a more accurate model. This involves obtaining footage from more patients and a greater diversity of settings and camera angles. Our transfer function needs to incorporate optimization of the calibrated spinal points. The transfer function also needs to be written in python rather than MATLAB to allow for easier deployment. As mentioned above, MATLAB has many restraints, one of which is the license required to use it. The cloud system requires development of a front end system for users, both patients and doctors alike, to interact with the system.

XIII. Lessons

Deployment of the cloud has taught our team the importance of documentation and ensuring that the code works cross-platform. The cloud has also taught our team the importance of reading documentation and being meticulous about the architectural design of our pipeline to maximize costs and efficiency.

Our testing has also shown that when it comes to building models surrounding a patient, these models often need numerous amounts of data concerning the patient. Therefore, a strategic plan of obtaining a diverse amount of data is required.

XIV. Technical Appendix

Refer to the wiki for technical information - https://ciis.lcsr.jhu.edu/doku.php?id=courses%3A456%3A2023%3Aprojects%3A456-2023-08%3Aproject-08

XV. Reading List

1. Franco, L., Sengupta, R., Wade, L., & Cazzola, D. (2021). A novel IMU-based clinical assessment protocol for Axial Spondyloarthritis: a protocol validation study. PeerJ, 9, e10623.

2. Seel, T., Raisch, J., & Schauer, T. (2014). IMU-based joint angle measurement for gait analysis. Sensors (Basel, Switzerland), 14(4), 6891–6909.
3. Moon, K. S., Gombatto, S. P., Phan, K., & Ozturk, Y. (2022). Extraction of Lumbar Spine Motion Using a 3-IMU Wearable Cluster. Sensors (Basel, Switzerland), 23(1), 182.
4. D'Antoni, F., Russo, F., Ambrosio, L., Vollero, L., Vadalà, G., Merone, M., Papalia, R., & Denaro, V. (2021). Artificial Intelligence and Computer Vision in Low Back Pain: A Systematic Review. International journal of environmental research and public health, 18(20), 10909.
5. Lee, Yoonho, et al. "Surgical fine-tuning improves adaptation to distribution shifts." arXiv preprint arXiv:2210.11466 (2022).
6. Güler, Rıza Alp, Natalia Neverova, and Iasonas Kokkinos. "Densepose: Dense human pose estimation in the wild." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
7. Guo, Yunhui, et al. "Spottune: transfer learning through adaptive fine-tuning." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.

XVI. References

1. Zhang, Y. et al. Therapeutics for enhancement of spinal fusion: A mini review. J. Orthop. Transl. 31, 73–79 (2021).
2. Rajaee, S. S., Kanim, L. E. A. & Bae, H. W. National trends in revision spinal fusion in the USA. Bone Jt. J. 96-B, 807–816 (2014).
3. Dhillon, K. Spinal Fusion for Chronic Low Back Pain: A 'Magic Bullet' or Wishful Thinking? Malays. Orthop. J. 10, 61–68 (2016).
4. Deng, Jia, et al. "Imagenet: A large-scale hierarchical image database." 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009.
5. *Sgd*¶. SGD - PyTorch 2.0 documentation. (n.d.). https://pytorch.org/docs/stable/generated/torch.optim.SGD.html
6. MSELoss — PyTorch 2.0 documentation. (n.d.). https://pytorch.org/docs/stable/generated/torch.nn.MSELoss.html