1. Project Title
   HomePiano: A Comparative Study between the Efficiency of Random Forest and Support Vector Machines in a Monocular Augmented Reality Piano Hand-Key Detection Program.

2. Team Members

Responsibility	Member
Data Preprocessing	Chan
Modelling / Training the Model	Chan
Model Evaluation	Waks
Documentation	Waks
Program Implementation (For the Augmented Reality Piano)	Waks (OpenCV, MediaPipe) Chan (Audio Player)

3. Problem Statement

   It is commonly known that for a person to learn how to play the piano, they must own one, especially when harnessing the required skills to master the piano. Thus, aspiring students who are interested in learning how to play the piano are hindered due to the price needed to afford one, making us lose potential pianists and prodigies.

   However, challenges arise when it comes to creating a hand-gesture sensor to create a digitalized piano. Such is tracking hand pose in the world of computer vision, notably gesture recognition utilizing colored gloves or markers on hands to track finger position and depth [1]. However, there has been a rise in non-invasive, monocular setups, such as those by [2], that utilized an RGB camera to detect piano gestures to implement an Augmented Reality (AR) piano but faced difficulties in detecting different hand-shape variations and finger occlusion.

   The issue of depth remains an issue for its AR implementation, as differentiating between a hover and a pressing action by relying on a singular camera that can only capture frames in a 2-dimensional plane makes it all the more puzzling. Other model-fitting methods utilize calibration to solve for the limitation of depth but are sensitive to initialization [3][4], with other implementations utilizing a second camera or sensor to measure the depth of the hand [5][6].

   The studies mentioned are useful for creating an AR piano but often require additional sensors or cameras to function properly. To make the model accessible to

those who cannot afford two cameras, we intend to utilize the models for the following: (1) MediaPipe's hand-detection and tracking model for marking hand joints and tracking complex hand gestures. (2) Python's OpenCV2 image recognition model to detect ArUco markers for the piano's borders and the depth or distance of the marker to the camera. Then, (3) Random Forest (RF) and Support Vector Machine (SVM) models to combat the depth limitation by using OpenCV2's ArUco marker size and MediaPipe's landmark coordinate distances (landmark size heuristics) to distinguish between a hover and pressing action of a finger.

These models will work hand-in-hand to create an AR piano using pre-existing hand-detection and image recognition models, together with RF and SVM classification models, to develop a program that allows people to print out a piano-keyboard cutout to play the piano using a single finger, whilst having a camera track their hand movements in real time.

4. Project Objectives and Scope

   HomePiano aims to implement an augmented reality (AR) piano program that can detect a printed-out piano key placed on a flat surface, where it can play piano-key audios when detecting a finger pressing action on the printed piano key, whilst simultaneously being able to differentiate between a press and hover action over the printed piano keys.

   Moreover, given the timeframe allotted for this project, we, the researchers, will limit its implementation to:

   - Detecting piano keys using one finger
   - A Top-down view program implementation run on a laptop utilizing a web camera to detect the hands and piano keys
   - The printed piano cut-out will only consist of 7 keys, from A to G, and will be able to detect only white keys.

5. Dataset Description

   Dataset Name: HomePiano Hand Dataset

   Description: We will create our own dataset, called the HomePiano Hand Dataset, which is made by taking pictures of our hand pressing over the prepared 7-key printed paper piano. The reason for creating our own dataset is to establish the (1) ground truth on which finger is hovering or pressing in the image and (2) the distance of the hand to the camera.

   A key limitation of existing datasets is that they do not contain the ground truth of which fingers are hovering or touching the table/ground, and even if we feed the image to the readily available MediaPipe model, it alone cannot determine the depth of the hand regardless of the hands found in the dataset due to the differing hand sizes; thus, this can only be resolved by utilizing the ArUco markers as determinants for the depth factor.

We will then use the joint-distance provided by MediaPipe's hand-marking model and OpenCV2's ArUco markers to determine the paper's depth to make a model that is capable of distinguishing finger pressing and hovering actions, given the finger-joint and paper distances.

Thus, given its limitations, we intend to create our own dataset, ensuring the ground truth labeling of hovering and pressing actions over the printed 7-key piano, together with the depth of the hand with respect to the camera.

Phone Used: Infinix Hot 40 Pro
Megapixels: 6.8 MP
Focal Length: 5.25mm
ISO sensitivity: Automatic
Shutter Speed: Automatic
Aperture: f/1.8
Quality / Pixels: 3936 x 1728 pixels

ISO sensitivity and Shutter Speed will be set to automatic to allow for diverse distances collected over different data points.

Format: JPEG

Size and Features: The created dataset will consist of at least 500 pictures of the hands of two people (researchers) over a printed 7-key paper piano. The metadata will include the list of fingers pressing the paper piano for each image. Additionally,

Preprocessing Needed: No, as most of the data will be outputs made by the MediaPipe model and OpenCV2's distance calculation.

Dataset: HomePiano Distance Dataset
Description: A dataset will be generated that contains the finger-joint, finger-wrist, and ArUco camera distances and the label for the finger pressed. This dataset will be used to train the different RF and SVM models.

The dataset will be generated by (1) feeding the image to MediaPipe for the finger-joint and finger-wrist distances, formulated by using the x and y coordinate distance formula: np.sqrt((x2 - x1) ** 2 + (y2 - y1) ** 2), where x1 and y1 is the finger dip or wrist, and x2 and y2 is the finger tip.

Then (2), forming the distance formula for the ArUco markers: (Known Distance x Known Pixel Width) / Current Pixel Width

Format: CSV

Size and Features: The dataset will be 5x as large as the HomePiano Hand Dataset, as we intend to segment each finger to be one entry to the dataset. Features will include joint_distance, finger_wrist_distance, depth, and finger_gesture

Any preprocessing needed? Yes, as we will integrate the labels from the HomePiano Hands Dataset's metadata into the HomePiano Distance Dataset, to prepare the binary classification labels for training

6. Proposed Machine Learning Approach
   Type of Problem: Classification

   We intend to use Random Forest and Support Vector Machines, with their different parameters (e.g., number of trees and kernels), as mediums for predicting press and hover actions made by the user.

   We've decided to use RF and SVMs due to their ability to capture non-linear relationships; similarly, they are less prone to overfitting as compared to their baseline models (e.g., Naive Bayes and Decision Trees). Moreover, we wanted to compare the two models in terms of accuracy and speed with respect to the nature of our dataset. Random Forest works best with tabular and large datasets, whereas SVMs thrive more on high-dimensional data. Additionally, we want to determine which model can correctly classify a pressing and hovering action while keeping up with the constant predictions being made per frame scanned in the program.

   We intend to use these different variants of Random Forest models based on:
   - N-trees (10, 100, 1000)
     - This may change based on the recommendations found in different related literature.
     - Note that the number of trees affects the speed of the prediction
   - Splitting Criteria (Gini, Entropy, Classification Error)

   As well as different SVMs:
   - Vanilla SVM
   - Kernelized
     - Linear
     - Polynomial
     - Radial Basis Function (RBF)
     - Sigmoid

   Data that will be fed into the model:

   X_train: joint_distance (Index Tip to Index Dip), finger_wrist_distance (Index Tip to Wrist), finger_name, and depth (Recorded ArUco marker size, or distance of the paper to the camera)

y_train: finger_gesture (Press, Hover)
Output: Binary Classification (Press, Hover)

We will utilize Logistic Regression as the baseline model; however, Random Forests and SVMs will be explored in the program's implementation due to their flexibility in handling non-linear datasets. This is due to the uncertainty in the possible linear relationship of features present in the dataset that may be affected by the variety of hand sizes towards the joint and finger-to-wrist distance, with respect to the distance of the camera to the piano.

7. Evaluation Metrics
   Metrics Used:
   - Confusion Matrix: To determine the number of true positives and negatives, as well as the false positives and negatives
   - Accuracy: To determine, among all classifications, which entries the model predicts correctly
   - Precision: To determine among all predicted finger presses, which among them are *true* presses
   - F1-score: The harmonic mean of accuracy and precision, serving as a great metric for imbalanced datasets
   - Frames per Second: Determines how heavy the model is when integrated into the system of classifying finger gestures

   These metrics will be used to determine the model's accuracy and efficiency in classifying the two-finger gestures, provided the distances between finger joints and the ArUco marker from the camera. These metrics will also be applied to the different RF and SVMs

8. Tools and Technologies
   Programming Language: Python

   Python Packages used:
   MediaPipe: Google Hand-Marking model
   OpenCV2: Python package for camera access, and ArUco's image recognition and border-making capabilities
   Pandas: Python package consisting of tools such as CSV reading and writing capabilities, as well as dataset manipulation tools.
   Scikit-learn: Python package used for creating the models.
   DroidCam: An application that allows your phone's camera to act as a webcam for your computer
   Matplotlib & Seaborn: Visualization tools used to see the distribution of data points from the HomePiano Distance Dataset

9. Timeline and Milestones

Week	Task/Milestone
Week 1.1	Paper Piano Keyboard Tracking with ArUco Markers
Week 2.1	Coordinate Integration of the Keyboard tracking image transformation to the MediaPipe's hand landmark coordinates
Week 2.2	Rule-Based Detection System
Week 2.3	Creating the Lgow-Latency Audio Engine
Week 3	Creating the dataset (HomePiano Hand & Distance Dataset)
Week 4.1	Training the base model and vanilla RF & SVM models & Evaluation
Week 4.2	Testing for the integration of temporal data (May cause another data collection process)
Week 5.1	Training the different RF and SVM models & Model Evaluation
Week 5.2	Incorporating the models with MediaPipe, OpenCV2, and the Audio Engine
Week 6	Evaluate the model and prepare the report
Week 7	Present Findings in ML conference

10. Expected Challenges and Risks
    Classification Bias:

    Issue: Though the different RF and SVM models will be trained to differentiate between pressing and hovering actions, their implementation serves as a workaround to using several cameras and the limitation of depth present in current models. The model may not be sensitive enough to detect finger gestures at far distances, as the finger-joint distance will diminish with distance.

    Solution: To handle the issue of sensitivity over far distances, we may utilize (1) zooming in on the border of the piano before feeding it to MediaPipe to generate larger distances by the model and adjusting the marker's distance accordingly based on the zoom level. (2) Another is to utilize a temporal feature to combat the sensitivity, such as the rate of change (often called velocity in papers) of the finger to determine how the finger moves along a time frame, rather than using singular independent instances. Finally, (3) is an emerging technique utilizing monocular cameras and models to generate the 3D pose of hands [7], though we might find a way to utilize MediaPipe's 2D landmarks to generate a distance or z metric as another feature for the models.

11. Ethical Considerations
    Data Privacy: N/A. The dataset will be created by the researchers, and no personally identifiable information will be stored in said dataset.

    Informed Consent: A consent form will be presented to participants who will contribute images to the HomePiano Hand Dataset. The consent form will inform the participant of the following information:

(1) What data will be collected: Images of a user's left or right hand (One will be shown in the image)
(2) How the data will be used: They will be used to calculate the distances of joints, and will be used as training and testing data for the different RF and SVM models in classifying finger pressing or hovering gestures over the printed 7-key piano
(3) How the data will be stored and secured: The data will be stored locally and in the cloud, which will be accessible to and only to the researchers of this project.
(4) Rights to withdraw: Participants will have the right to withdraw their consent and have their data deleted at any time; with respect to this, their hands and names will be stored as a reference to the images in the dataset but will not be used in any other part of the project.

This information will be translated into an adviser-approved consent form that will be provided to the participants for their consent to use their hand as training and testing data for the classification models.

12. References

[1] A. Aristidou and J. Lasenby, "Motion capture with constrained inverse kinematics for real-time hand tracking," 2010 4th International Symposium on Communications, Control and Signal Processing (ISCCSP), Limassol, Cyprus, 2010, pp. 1-5, doi: 10.1109/ISCCSP.2010.5463419.

[2] Liang et al., "Barehanded music: real-time hand interaction for virtual piano," Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, pp. 87–94, 2016, doi: 10.1145/2856400.2856411.

[3] A. Tagliasacchi, M. Schröder, A. Tkach, S. Bouaziz, M. Botsch, and M. Pauly, "Robust Articulated‐ICP for Real‐Time Hand Tracking," Computer Graphics Forum, vol. 34, no. 5, pp. 101–114, Aug. 2015, doi: 10.1111/cgf.12700.

[4] Liang et al., "Barehanded music: real-time hand interaction for virtual piano," Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, pp. 87–94, 2016, doi: 10.1145/2856400.2856411.

[5] I. Oikonomidis, N. Kyriazis and A. A. Argyros, "Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints," 2011 International Conference on Computer Vision, Barcelona, Spain, 2011, pp. 2088-2095, doi: 10.1109/ICCV.2011.6126483.

[6] R. Wang, S. Paris, and J. Popović, "6D Hands: Markerless hand-tracking for computer aided design," Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, pp. 549–558, Oct. 2011, doi: 10.1145/2047196.2047269.

[7] F. Mueller et al., “GANerated Hands for Real-time 3D Hand Tracking from Monocular RGB,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 49–59, 2018, doi: 10.48550/arXiv.1712.01057.