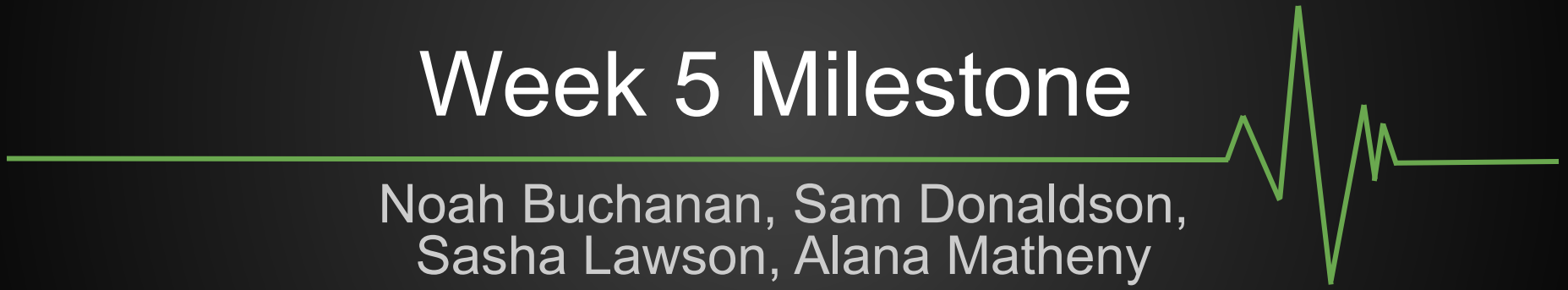
VizFit:

Week 5 Milestone

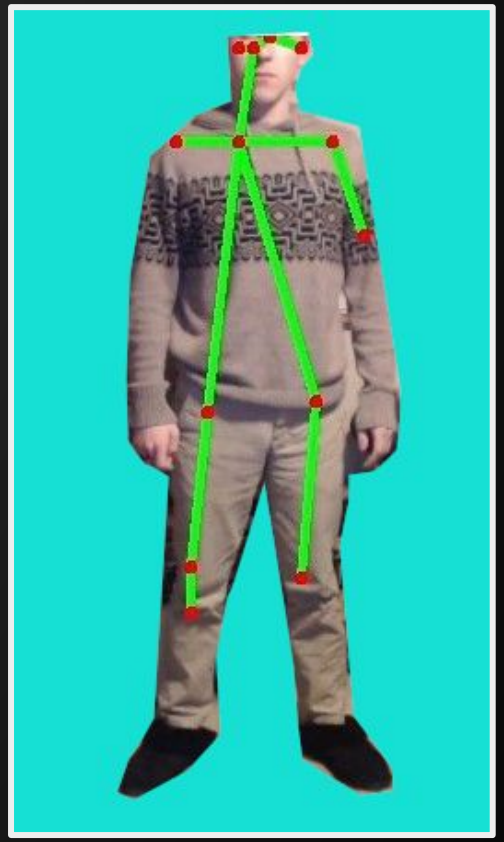


Noah Buchanan, Sam Donaldson,
Sasha Lawson, Alana Matheny

Pose Models

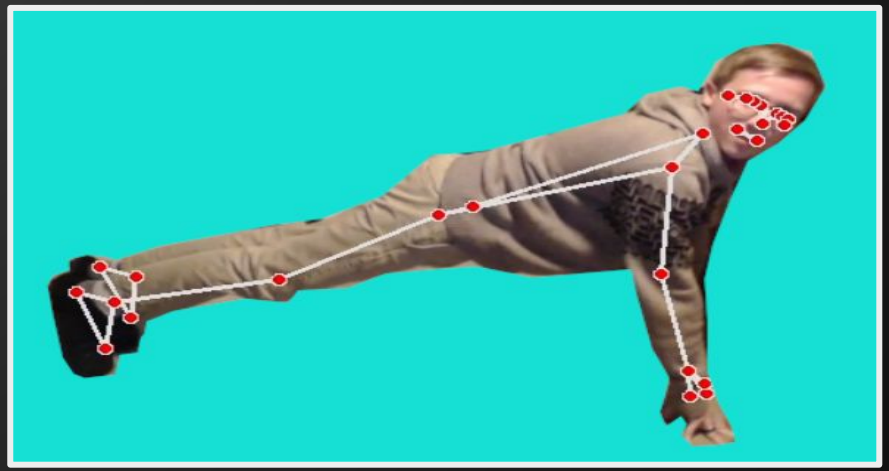
Model 1: Using OpenPose[3]

- Problems:
  - Doesn’t recognize profile views
  - Doesn’t recognize hands or feet accurately
  - Very unreliable when constantly moving



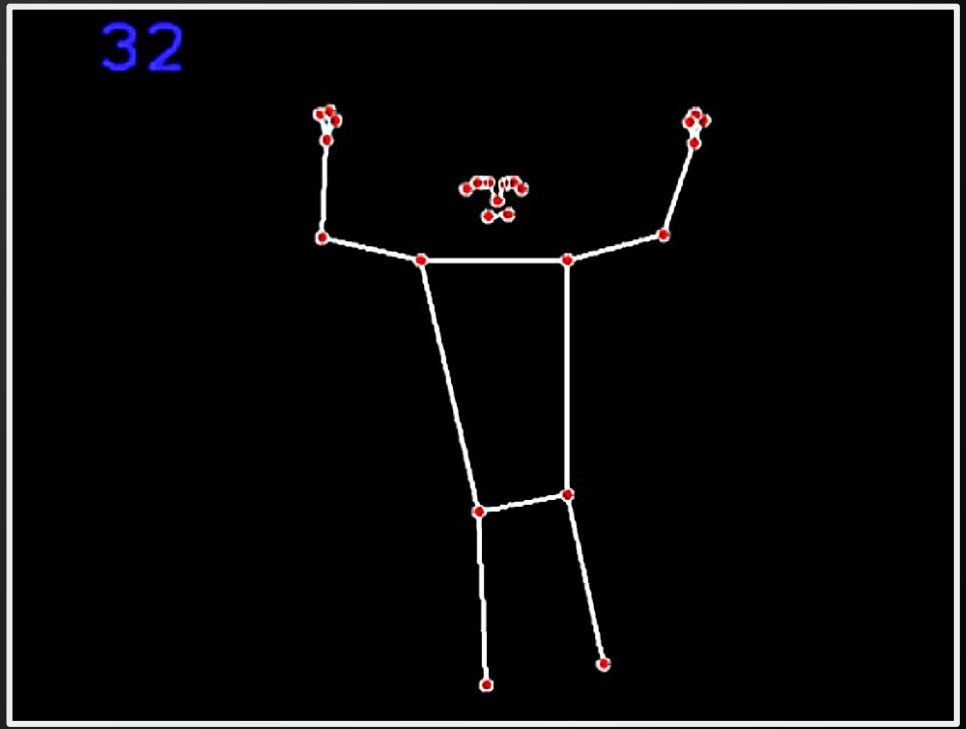
Model 2: Using MediaPipe[2]

- Features:
  - Does recognize profile views
  - Does recognize hands and feet
  - Somewhat reliable when constantly moving



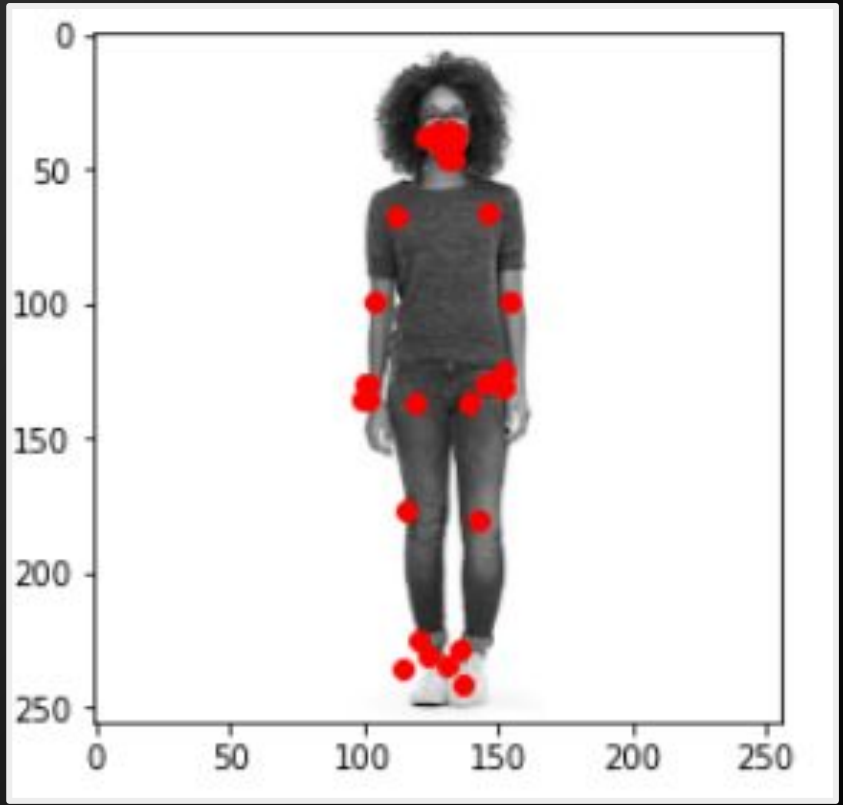
Model 2: Using MediaPipe[2]

- Removed background

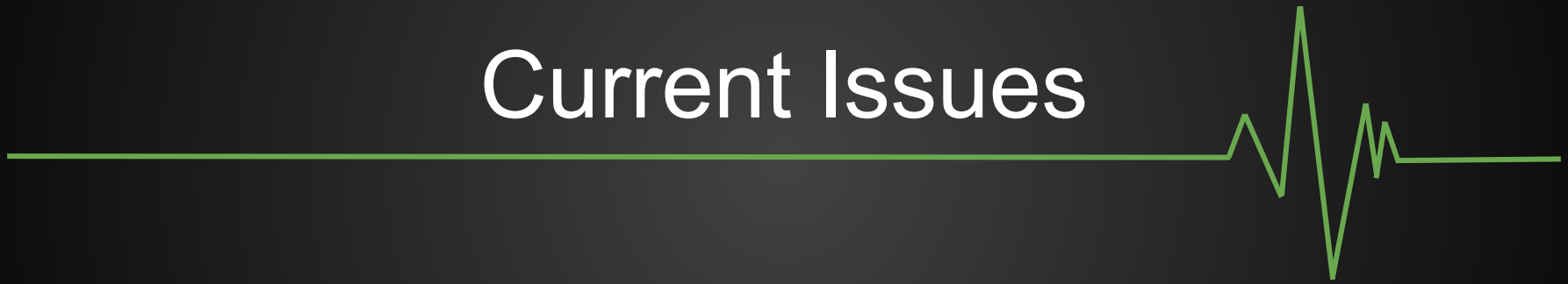


Our Custom Model

- Features:
  - 3D pose estimation is our long term goal
    - 3D pose estimation is more accurate for fitness purposes
  - Short term goal is to get a very accurate 2D estimation
  - Somewhat reliable when constantly moving



Current Issues



Datasets and Time

- Labeled data means that our keypoints are decided by the dataset we train on

- With the lack of useful datasets, potentially creating our own dataset could be a time consuming task

Speed and Accuracy

- The main goal is accuracy, but our model speed could affect accuracy indirectly

- We could be missing potentially crucial frames of information important to classification if our model is too slow in making predictions with a live feed of data

Methods Used

Labeling

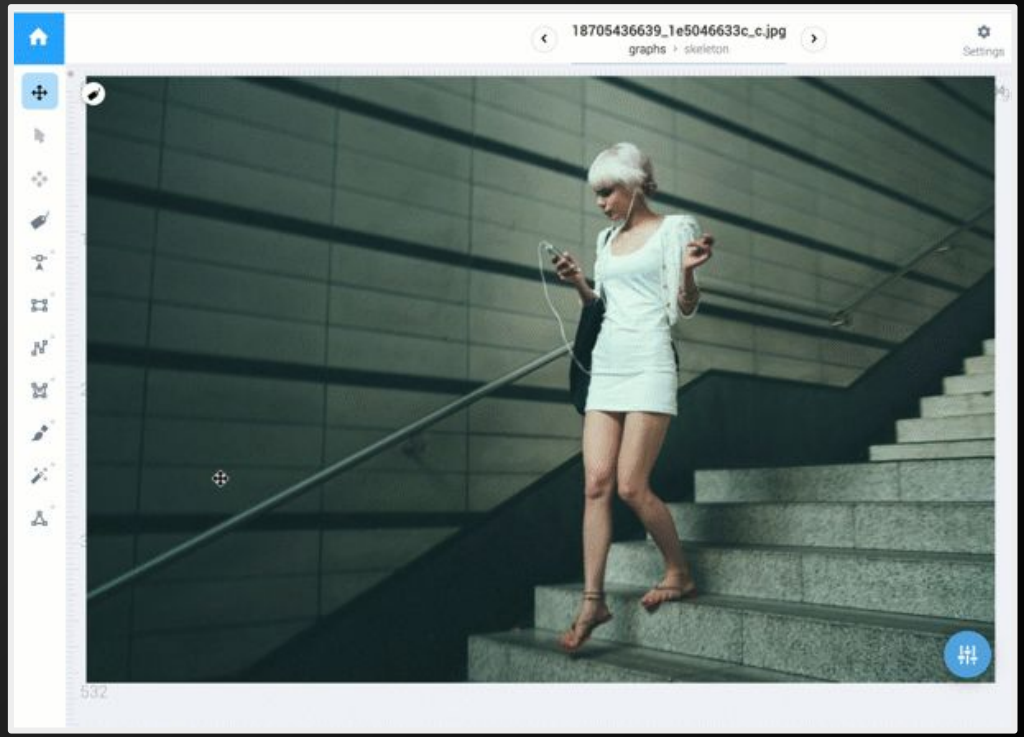
- Our data labels for pose estimation will simply be scaled values of (x,y) coordinates of however many key-points will be included as well as a confidence value for each key-point coordinate
  - [0.932,0.1823,0.987] -> [scaled x, scaled y, confidence]

- As for the exercise recognition model, the labels will likely just be the name of the exercise, or the name of the exercise and a binary value for a correct or incorrect performance
  - [‘pushup’,’1’] -> [exercise, performance]

Methods of Labeling

- Labeling manually using Supervisely
  - Allows us to customize the wireframe and key-points to easily overlay onto an image
- Automatically label images using a pre-trained model that we have working

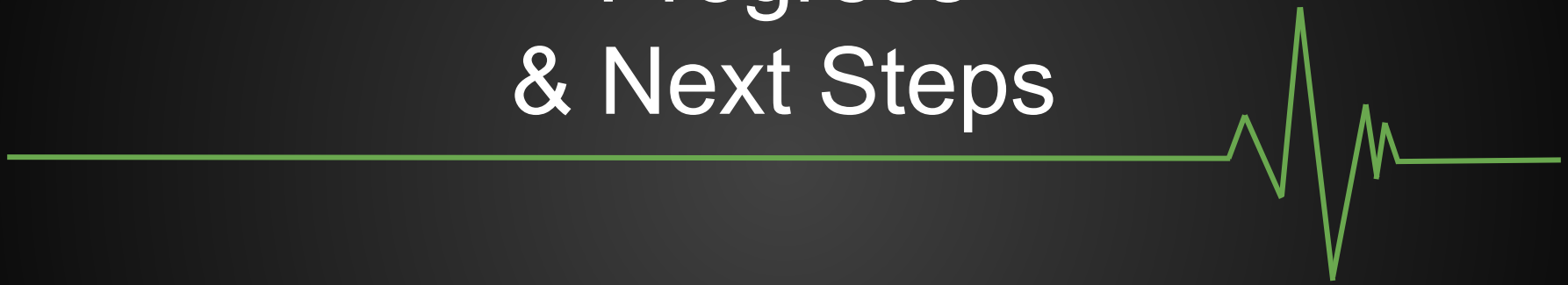
[1]



Methods of Training

- Convolutional neural networks
  - Exact architecture is still being determined
  - Overfitting prevention
    - Dropout layers to make the model relearn a portion of the weights throughout training
  - Small kernel sizes (3x3, 1x1)
    - Allows more convolutional layers and more output filters to encode more information with the same if not less amount of parameters as larger kernels

Progress
& Next Steps



Progress to Date

- Backend API
- Vue Application
  - Front-end and back-end API connection
- Wireframe Model
  - Model 1
  - Model 2
  - Custom Model

Next Steps

- Server setup
- Database setup
- Make our pose estimation model as accurate as possible
  - This model must be very accurate before we can move to the exercise recognition model because the data used is the predicted coordinates of keypoints
- Connecting database and application
- Extra documentation

References

[1] Supervise.ly. “Introducing Keypoints Labeling in Supervisely.” Medium, Supervisely, 8 June 2019. [Online]. Available: https://medium.com/deep-systems/introducing-keypoints-labeling-in-supervisely-de6c4459157d.

[2] Latest Pose Estimation Realtime (24 Fps) Using ... - Youtube. [Online]. Available: https://www.youtube.com/watch?v=brwgBf6VB0I.

[3] Athtripathi. “Using Openpose Library in Python with Openpose Tensorflow and Opencv.” Medium, Geek Culture, 9 Sept. 2021. [Online]. Available: https://medium.com/geekculture/using-openpose-library-in-python-with-openpose-tensorflow-and-opencv-10a5496c359a.

Questions