Thesis Report 5 : 16 March - 23 March

Goals

- Start working on PAD-LMA mapping using regressions ✅
- Implement Dominance regression into LMA-PAD mapping subsystem ✅
- Improve emotion classification integration into motion learning subsystem ✅

Last Week Leftovers:

None

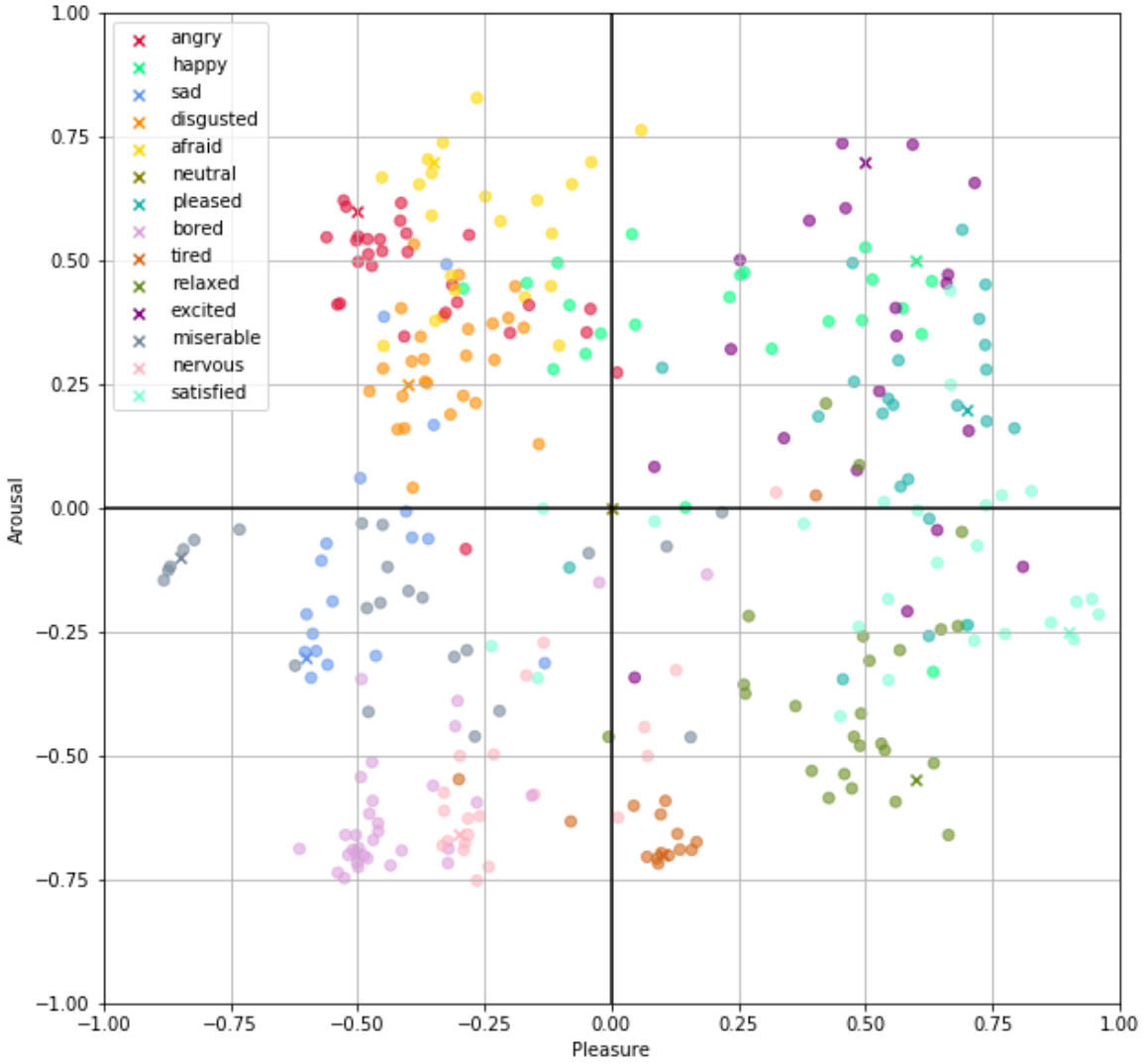
Done

- Added another regressor to map LMA features to the **dominance** coordinate
- Continued tweaking and messing with LMA-PAD Gradient Tree Boosting Regression hyperparameters
- **Managed to achieve under 0.2 Mean Absolute Error on the Test Set for all 3 emotional coordinates**
    - Achieved this by tweaking some hyperparameters and swapping from normalization to standardization

Test Set MAE & MSE

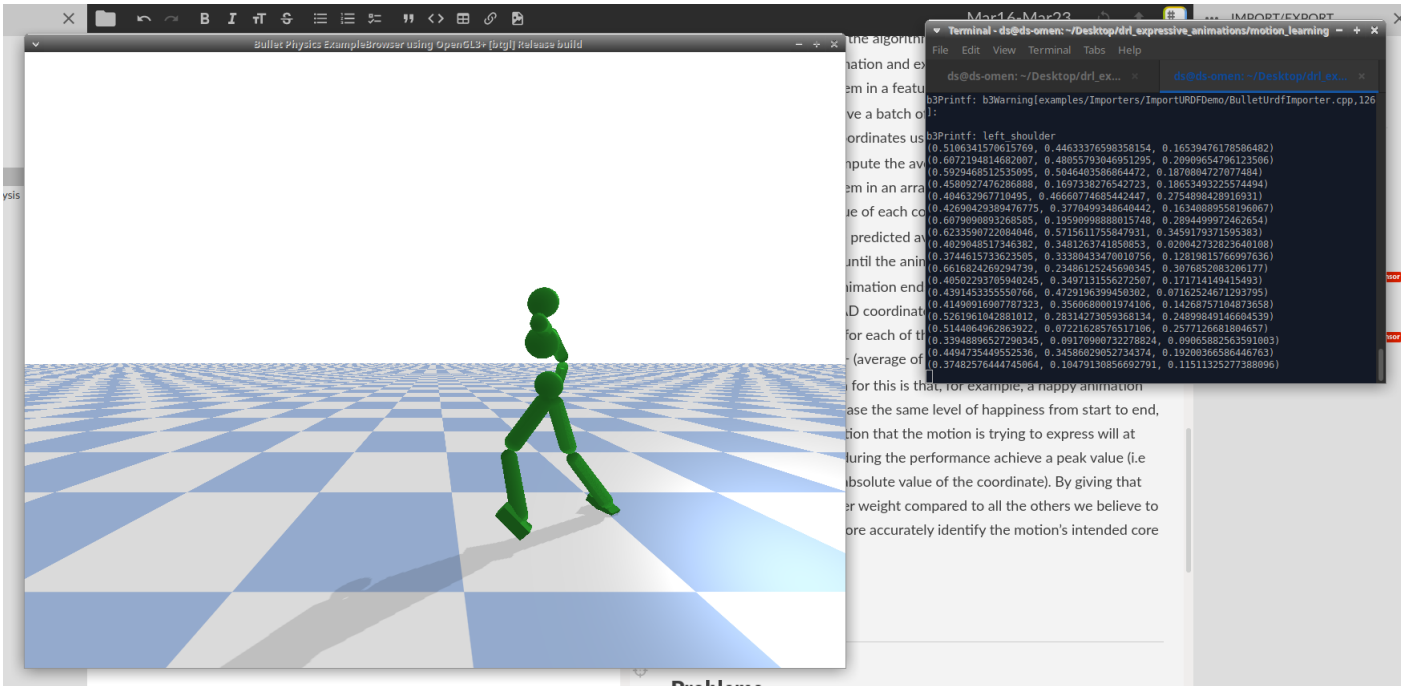
In [851]:
```
pred_y_p = model_p.predict(test_X)
mse = mean_squared_error(test_y_p, pred_y_p)
mae = mean_absolute_error(test_y_p, pred_y_p)
print("Pleasure")
print("MSE: %.2f" % mse)
print("MAE: %.2f" % mae)


pred_y_a = model_a.predict(test_X)
mse = mean_squared_error(test_y_a, pred_y_a)
mae = mean_absolute_error(test_y_a, pred_y_a)
print("\nArousal")
print("MSE: %.2f" % mse)
print("MAE: %.2f" % mae)


pred_y_d = model_d.predict(test_X)
mse = mean_squared_error(test_y_d, pred_y_d)
mae = mean_absolute_error(test_y_d, pred_y_d)
print("\nDominance")
print("MSE: %.2f" % mse)
print("MAE: %.2f" % mae)
```

```
Pleasure
MSE: 0.09
MAE: 0.19

Arousal
MSE: 0.06
MAE: 0.16

Dominance
MSE: 0.08
MAE: 0.19
```



- Improved integration algorithm of Emotion Classification with the Motion Learning submodule
  - Identified a bug in the code related to the normalization of extracted features
  - Added Dominance model and prediction
  - Changed the way the algorithm works. Right now, we:
    - Run the animation and extract LMA features every 60 frames and store them in a features array.
    - When we have a batch of 5 LMA features we predict each of their PAD coordinates using our models
    - We then compute the average of our PAD coordinate values and store them in an array. We also keep track of the highest absolute value of each coordinate so far
    - We print the predicted average PAD coordinates and repeat the process until the animation ends or loops
    - When the animation ends or loops we compute the final predicted PAD coordinates by doing a weighted average using the formula for each of the coordinates: (biggest absolute value) * 0.5 + (average of all other stored values) * 0.5
    - The intuition for this is that, for example, a happy animation won't showcase the same level of happiness from start to end, but the emotion that the motion is trying to

express will at some point during the performance achieve a peak value (i.e the highest absolute value of the coordinate). By giving that value a higher weight compared to all the others we believe to be able to more accurately identify the motion's intended core emotion



- Started working on Motion Synthesis module (i.e PAD to LMA regression)
- Using (once again) XGBoostRegression
- Using the same dataset we used to train our LMA-PAD regressions, we took the PAD coordinates as features (i.e the X), and created an array of target variables (i.e an array of y's), one for each LMA feature (excluding distance traveled)
- We then trained one regression model for each LMA feature (of which we had 32)
- We then compared the generated LMA features to the real values

```
== LMA Feature - avg_hand_distance ==
Real: 0.4796902511846765
Predicted: 0.4366816

== LMA Feature - avg_l_hand_hip_distance ==
Real: 0.280837420638497
Predicted: 0.28031227

== LMA Feature - avg_r_hand_hip_distance ==
Real: 0.255294820375872
Predicted: 0.2847639

== LMA Feature - avg_feet_distance ==
Real: 0.1817902988913359
Predicted: 0.2483332

== LMA Feature - avg_l_hand_chest_distance ==
Real: 0.4778779897628815
Predicted: 0.42441794

== LMA Feature - avg_r_hand_chest_distance ==
Real: 0.4649615450562844
Predicted: 0.42679408

== LMA Feature - avg_l_elbow_hip_distance ==
Real: 0.3572558475164683
Predicted: 0.32761985

== LMA Feature - avg_r_elbow_hip_distance ==
Real: 0.3456088495616469
Predicted: 0.33056936

== LMA Feature - avg_chest_pelvis_distance ==
Real: 0.2861510000107813
Predicted: 0.28615132

== LMA Feature - avg_neck_chest_distance ==
Real: 0.2788742009852752
Predicted: 0.27867973

== LMA Feature - avg_neck_rotation_w ==
Real: 0.0055258431771388
Predicted: 0.01010515
```

- We also used our trained LMA-PAD models to predict the PAD coordinates of our generated LMA features

Run predictors

```
In [156]: y_p = model_p.predict(df)
          y_a = model_a.predict(df)
          y_d = model_d.predict(df)
```

Print Results

```
In [157]: print("Real PAD Coordinates: " + str((x[0][0], x[0][1], x[0][2])))
          print("Predicted PAD Coordinates: " + str((y_p[0], y_a[0], y_d[0])))

          Real PAD Coordinates: (-0.3, -0.66, -0.7)
          Predicted PAD Coordinates: (-0.17166759, -0.47006065, -0.34888157)
```

- Results were ok but could be improved. Didn't have time to perform much hyperparameter tweaking or play around with settings too much

Left Undone

Problems

- **[SOLVED]** On the Emotion Identification - Motion Learning integration, there's **a brief pause** in the animation in PyBullet while we're doing the prediction. I believe this is caused due to the fact that we're running everything without parallelization. So before we draw the next frame, we're actively waiting for our models to finish predicting the PAD coordinates.
    - I believe a solution to this might be to parallelize the Emotion Classification Algorithm so that our frame drawing doesn't have to wait for it to be done. Alternatively, there might be ways to optimize the speed of our model's predictions by either reducing the number of LMA feature rows we pass it, or by using a different data structure rather than the Pandas Dataframe.

Notes

Thoughts

Didn't get to dedicate as many hours to the thesis as I wanted this week. I'm part of GameDevTecnico and each month we scout some game jams and try to participate in a few. Since I hadn't participated in any jams for a while I felt the need to join in with the rest of the group on this one.

Regardless of this, I still managed to dedicate a fair amount of hours to the thesis and did make some significant progress (in my opinion). So far we got fairly decent models for LMA-PAD and PAD-LMA mapping. Whilst they can definitely be improved, either by tweaking hyperparameters/datasets or by trying out something else like an AutoEncoder (which I will be

trying out during the next week), they're quite presentable as is.

The idea for this first phase of the thesis (running from February to March) was to have the Motion Learning (i.e spacetime bounds) system working, learning how it works, and creating the LMA-PAD and PAD-LMA mappers. For the next phase (April-May) I have to create the motion synthesis algorithm (i.e taking the currently running animation and the generated PAD features according to the user input, and using them to tweak the joint positions of the skeleton so that it expresses the intended emotion), and the GUI (I will probably be using the PyBullet interface and just adding text fields to edit and outputting PAD coordinates, so as to not necessarily overcomplicate). Afterwards all that's left is user testing and writing the thesis, which will be done in June-July.

This being said, working on two projects at once burnt me out a lot this week as I had little to no free time. As such, I will probably be taking a 1 week break after next week (i.e at the end of Phase 1). This will allow me to rest for a bit, and also work on other projects (Portfolio 2 and creating my own CV to start applying for companies). Taking into account progress has been going pretty smoothly, it should also not delay me by any significant amount.

Work Hours

- Worked Thursday and Friday from 13pm to 3pm
- Worked Monday and Tuesday from 13pm to 18pm Couldn't work too many hours this week due to my participation in a Game Jam with the GameDevTecnico Group - https://antunes10.itch.io/felicita