ENGR 301
Engineering Applications of Data Science

**Laboratory 6**
Topic: Electrical Engineering; Subtopic: Robot Movement

**Introduction:** By themselves, robots have no idea how to do anything. Data science is extensively used in training robots to do a wide variety of tasks, including navigation. Typically, sensors are attached to the robot, which feeds in information about its location, speed, surrounding heat signals, etc. This information is then used to train models which inform the robot about what to expect given certain conditions. A similar procedure is used in such applications like self-driving cars. The quality of the model determines how well the robot performs in its applications. In this lab, you will use navigation data from a robot to determine what kind of surface it is travelling on.

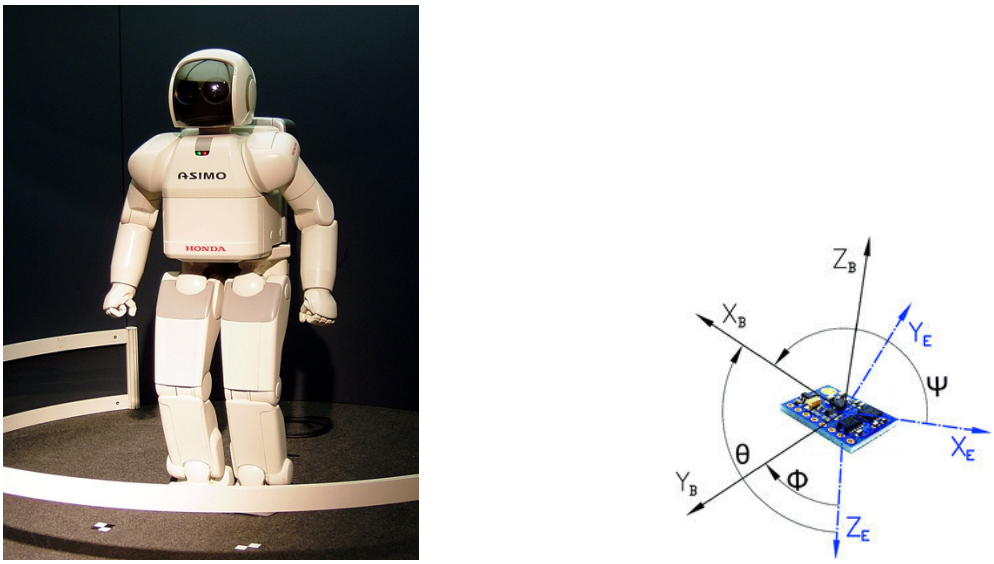


Figure 1. Left: ASIMO the robot. Right: an inertial movement unit sensor. (Images curtesy: https://en.wikipedia.org/wiki/Robot, Human Motion Characterization Using Wireless Inertial Sensors)

**Discipline Specific Information:** For this laboratory, you will receive movement data collected using inertial movement unit (IMU) sensors on a robot navigating on different types of surfaces. Specifically, for every point in time, you will have the robot's angular velocity (it's speed of motion), linear acceleration (how its speed is changing; speeding up or slowing down), and orientation (direction it is facing relative to the fixed coordinate system set by the surface; z is down while x and y are along the ground). These will constantly change as the robot moves around the surface.

The goal here is to predict what surface the robot is moving on given this movement data. By doing this, you can put this model in your robot to help it automatically determine what surface it is moving on next time.

There are nine different types of surfaces. Alphabetically, they are: (1) carpet, (2) concrete, (3) fine concrete, (4) hard tiles, (5) hard tiles with a large space, (6) soft PVC, (7) soft tiled, (8) tiled, (9) wood.

There were 3,610 experiments run, with each experiment containing 128 measurements (i.e., at 128 time steps). The movement data set therefore contains 482,080 data points. The surface data set contains only the surface each experiment was run on, and therefore contains 3,610 data points.

## Tasks

**This laboratory assignment and the next one will be slightly different than the previous ones.** You will have a lot more freedom in these assignments in terms of feature engineering, hyperparameter tuning, etc. Also, you will be given unlabeled test data to predict upon, the accuracy of which will be checked against the actual results (which are withheld from you). This will be described more in the final part of the lab.

### Part 1 – Exploratory Data Analysis

1. The movement data for this part is provided on Canvas in a CSV file called:

robots_movement.csv

while the surface data is provided in a CSV file called:

robots_surface.csv

Import both into their own data frame.

2. What is the distribution of the different surface types? What do you notice about the distribution?

3. Look at the input features. What is the correlation between the input features? What do you notice about the correlation between them?

4. For the first experiment (the one with a series ID of 0), plot each of the features versus time (i.e., measurement number). What do you notice, if anything?

### Part 2 – Feature Engineering I and Baseline Models

1. Notice that because the input features data set is much larger than the output data set, we have to collapse the input features one in order to make predictions.

As we now know, the easiest way to do this is using the *pandas* groupby function. The question, however, is what metric do we use to collapse the time dependent data (measurement number)?

Let's start with the most straight forward one, the mean. Create a new data frame in which you use the groupby function on the series ID using the mean. What is the shape of this new data frame? (Hint: it should be 3,610, exactly the number of experiments performed).

2. Assign all of the input features to x, making sure to drop the measurement_number feature, which is a so-called useless predictor.

3. Label encode your surface types and assign them to y. Make sure only this information is assigned to y.

4. Train a logistic regression model on this data (use a max_iter of 500 to ensure convergence). Make sure to split your data into appropriate training and test sets. How does this model perform? Use accuracy and a confusion matrix to justify your thoughts.

5. Train a k-nearest neighbors classifier using 2 nearest neighbors. How does this model perform?

The KNeighborsClassifier model is part of the sklearn.neighbors package.

6. Use a GridSearchCV procedure to find the optimum number of nearest neighbors and the best distance metric (Euclidean or Manhattan) for the k-nearest neighbors. What are the best parameters? Does this improve the performance of the model?

GridSearchCV is part of the sklearn.model_selection package.

7. Train a random forest classifier on this data, using 800 decision trees. How does this model perform?

The RandomForestClassifier is part of the sklearn.ensemble package.

Part 3 – Feature Engineering II and Improving the Models

In this part, we will look at how more advanced feature engineering can improve the quality of data science models.

1. First, we will make new features based on the features we have. This is an incredibly important part of improving models.

Create new features in your data set containing the input features based on the following relationships:

$$Total\ direction = \sqrt{(direction\ x)^2 + (direction\ y)^2 + (direction\ z)^2}$$

$$Total\ angular\ velocity = \sqrt{(angular\ velocity\ x)^2 + (angular\ velocity\ y)^2 + (angular\ velocity\ z)^2}$$

$$Total\ linear\ acceleration = \sqrt{(linear\ acceleration\ x)^2 + (linear\ acceleration\ y)^2 + (linear\ acceleration\ z)^2}$$

Why did we choose to engineer/create new features in this way?

2. Next, create a blank data frame. (set new_df = pd.DataFrame() ).

Write a Python *for* loop which loops over every column in your input features dataframe, computes the mean, minimum, maximum, and standard deviation of each series, and assigns it to the new data frame.

Some starting code is provided here:

```
for col in df_old.columns:
    df_new[col + '_mean'] = df_old.groupby(['series_id'])[col].mean()
    ....
```

Fill in the rest by yourself.

What is the shape of this new data frame? (Hint: it should be 3,610, exactly the number of experiments performed).

3. Train the same logistic regression, k-nearest neighbors, and random forest models as before on this new data frame. Remember to scale and train/test split your data as appropriate.

Compare the performance of each of these models with each other and with the models from the previous part.

How did this feature engineering affect the performance of these models?

Part 4 – The Ultimate Showdown of Ultimate Destiny

In the last cell of your notebook, create one cell in which you import the unlabeled movement data (movement_test.csv), perform a prediction of what surface each robot was on (including feature engineering, your best model with optimized hyperparameters, etc.), and write your predictions to a CSV file.

Make sure you use only your best model with the appropriate hyperparameters here, which you should have trained in part 3. There should be no GridSearchCV here. Just use the model.predict() function on the new engineering x data and generate new predicted y values.

Change the y values to a data frame and write it to a CSV file. The appropriate command to write to a CSV file (assuming you imported *pandas* as pd and called your data frame df) is:

df.to_csv(“file_name.csv”)

It will write it to the directory where you launched the notebook. If you have trouble finding it, search in the finder for it.

The output of this CSV file will be compared against the actual values to determine the accuracy of your predictions.