Datenanalyse Kompensiert und Nicht Kompensiert

1. DeepMotion Nicht Kompensiert

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.spatial import distance
from scipy.interpolate import interp1d
from matplotlib.offsetbox import OffsetImage, AnnotationBbox
```

In [ ]:
```
df_deepM_03 = pd.read_csv('../Data/KeypointsBereinigtNichtKompensiert.csv
df_deepM_03.head(5)
```

Out[ ]:

	compensation	frame	path	x_0	
0	0	1	/Users/salomekoller/Library/CloudStorage/OneDr...	0.0000	-9
1	0	2	/Users/salomekoller/Library/CloudStorage/OneDr...	-0.2305	-9
2	0	3	/Users/salomekoller/Library/CloudStorage/OneDr...	-0.5451	-9
3	0	4	/Users/salomekoller/Library/CloudStorage/OneDr...	-0.8911	-9
4	0	5	/Users/salomekoller/Library/CloudStorage/OneDr...	-1.2253	-9

5 rows × 102 columns

1.2 Analyse Form

In [ ]:
```
print('Dimension:', df_deepM_03.shape)
print('Number of rows:', df_deepM_03.shape[0])
print('Number of columns:', df_deepM_03.shape[1])
```

```
Dimension: (937, 102)
Number of rows: 937
Number of columns: 102
```

1.2 Splitting Frames

In [ ]:
```
# Variablen initiieren
frame_count = 0
frames_until_reset = []

# Iterieren über Dataframe, um Frames mit '1' zu finden
for index, row in df_deepM_03.iterrows():
    frame_count += 1

    if row["frame"] == 1 and frame_count > 1:
        frames_until_reset.append(frame_count -1)
```

```python
if frame_count > 0:
    frames_until_reset.append(frame_count)

print("Number of frames until reset for each cycle:", frames_until_reset)
```

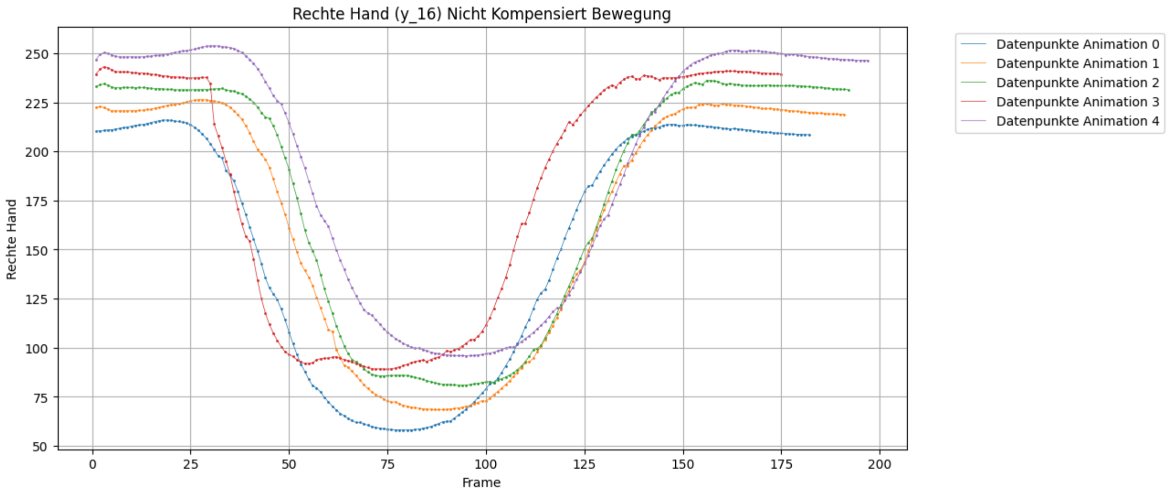
Number of frames until reset for each cycle: [182, 373, 565, 740, 937]

In [ ]:
```python
# Einzelne Bewegungen werden in verschiedene Dataframes gepackt
dfs = []
start = 0
for end in frames_until_reset:
    dfs.append(df_deepM_03.iloc[start:end])
    start = end

print("Number of splits:", len(dfs))
```

Number of splits: 5

1.3 Bewegungsanalyse anhand rechter Handbewegung (Nicht Kompensiert)

In [ ]:
```python
plt.figure(figsize=(12, 6))
for i, df in enumerate(dfs):
    plt.scatter(df["frame"], df["y_16"], marker='o', color=f'C{i}', s=1)
    plt.plot(df["frame"], df["y_16"], color=f'C{i}', linestyle='-', linew

plt.title("Rechte Hand (y_16) Nicht Kompensiert Bewegung")
plt.xlabel("Frame")
plt.ylabel("Rechte Hand")
plt.grid(True)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



1.3.1 Berechnung des Mean

In [ ]:
```python
means03 = []
mean_values_03 = df_deepM_03[['frame', 'x_0', 'y_0', 'z_0', 'x_1', 'y_1',
                              'x_4', 'y_4', 'z_4', 'x_5', 'y_5', 'z_5', 'x_6', 'y_6'
                              'x_8', 'y_8', 'z_8', 'x_9', 'y_9', 'z_9', 'x_10', 'y_1
                              'x_12', 'y_12', 'z_12', 'x_13', 'y_13', 'z_13', 'x_14'
                              'x_16', 'y_16', 'z_16', 'x_17', 'y_17', 'z_17', 'x_18'
                              'x_20', 'y_20', 'z_20', 'x_21', 'y_21', 'z_21', 'x_22'
                              'x_24', 'y_24', 'z_24', 'x_25', 'y_25', 'z_25', 'x_26'
```

```
                        'x_28', 'y_28', 'z_28', 'x_29', 'y_29', 'z_29', 'x_30'
                        'x_32', 'y_32', 'z_32']].groupby('frame').mean()
means03.append(mean_values_03)

mean_values_03.head(5)
```

Out[ ]:

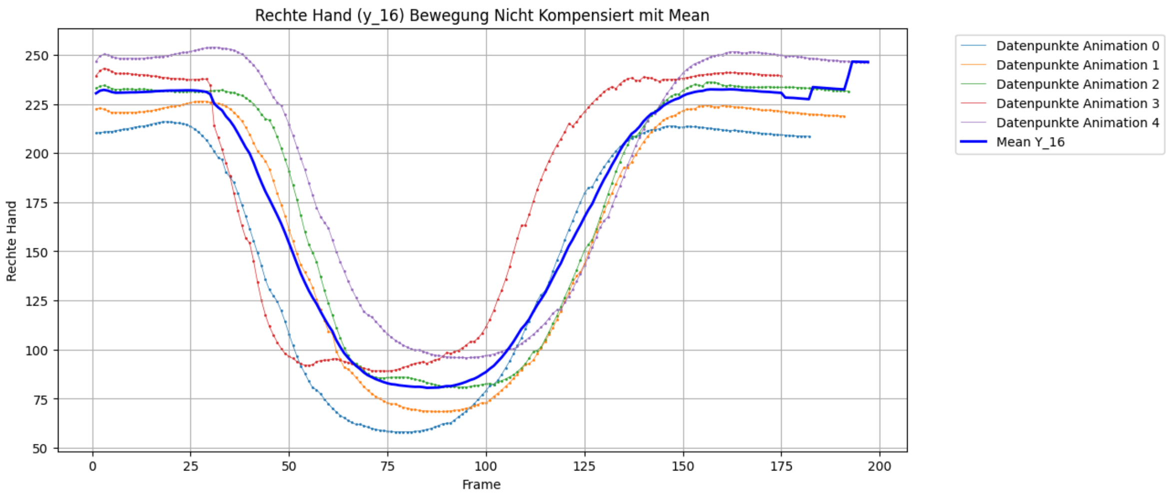
	x_0	y_0	z_0	x_1	y_1	z_1	x_
frame							
1	65.99356	18.27286	-171.467506	-993.7168	-312.4525	-60.97508	-993.716
2	66.85646	19.80700	-171.254204	-993.7168	-312.4525	-60.97508	-993.716
3	67.98854	20.18948	-170.908414	-993.7168	-312.4525	-60.97508	-993.716
4	68.69194	19.89282	-170.505374	-993.7168	-312.4525	-60.97508	-993.716
5	69.62770	19.35010	-170.065084	-993.7168	-312.4525	-60.97508	-993.716

5 rows × 99 columns

1.3.2 Bewegungsmuster mit Mean

In [ ]:

```
plt.figure(figsize=(12, 6))

for i, df in enumerate(dfs):
    plt.scatter(df["frame"], df["y_16"], marker='o', color=f'C{i}', s=1)
    plt.plot(df["frame"], df["y_16"], color=f'C{i}', linestyle='-', linew

plt.plot(mean_values_03.index, mean_values_03["y_16"], color='b', linesty
plt.title("Rechte Hand (y_16) Bewegung Nicht Kompensiert mit Mean")
plt.xlabel("Frame")
plt.ylabel("Rechte Hand")
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(True)
plt.show()
```

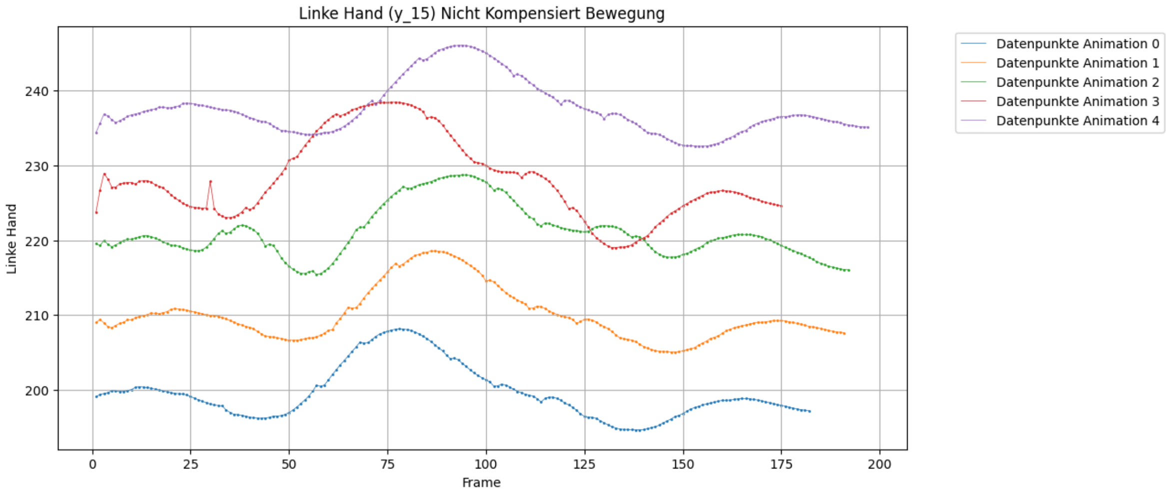


1.4 Linke Handbewegung Nicht Kompensiert

In [ ]:

```
plt.figure(figsize=(12, 6))
for i, df in enumerate(dfs):
```

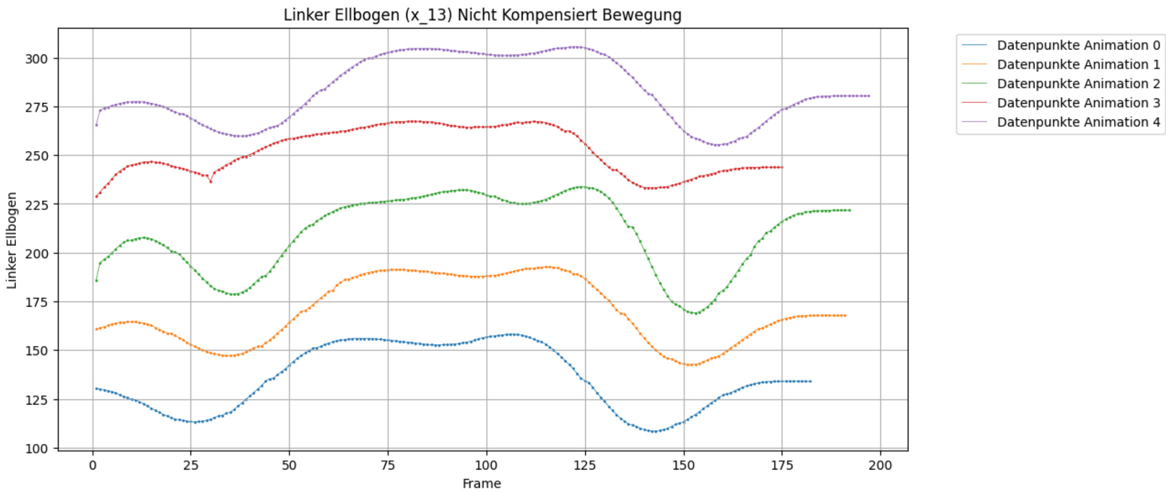
```
    plt.scatter(df["frame"], df["y_15"], marker='o', color=f'C{i}', s=1)
    plt.plot(df["frame"], df["y_15"], color=f'C{i}', linestyle='-', linew

plt.title("Linke Hand (y_15) Nicht Kompensiert Bewegung")
plt.xlabel("Frame")
plt.ylabel("Linke Hand")
plt.grid(True)
# Move the legend outside of the plot
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```



1.5 Linker Ellbogen Nicht Kompensiert

In [ ]:

```
plt.figure(figsize=(12, 6))
for i, df in enumerate(dfs):

    plt.scatter(df["frame"], df["x_13"], marker='o', color=f'C{i}', s=1)
    plt.plot(df["frame"], df["x_13"], color=f'C{i}', linestyle='-', linew


plt.title("Linker Ellbogen (x_13) Nicht Kompensiert Bewegung")
plt.xlabel("Frame")
plt.ylabel("Linker Ellbogen")
plt.grid(True)
# Move the legend outside of the plot
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```



2. DeepMotion Kompensiert

In [ ]:
```
df_deepM01 = pd.read_csv('../Data/KeypointsBereinigtKompensiert.csv', sep
df_deepM01.head(5)
```

Out[ ]:

	compensation	frame	path	x_0	
0	1	1	/Users/salomekoller/Library/CloudStorage/OneDr...	0.000	-100
1	1	2	/Users/salomekoller/Library/CloudStorage/OneDr...	1.909	-100
2	1	3	/Users/salomekoller/Library/CloudStorage/OneDr...	2.832	-100
3	1	4	/Users/salomekoller/Library/CloudStorage/OneDr...	3.748	-100
4	1	5	/Users/salomekoller/Library/CloudStorage/OneDr...	5.316	-100

5 rows × 102 columns

2.1 Analyse Form

In [ ]:
```
print('Dimension:', df_deepM01.shape)
print('Number of rows:', df_deepM01.shape[0])
print('Number of columns:', df_deepM01.shape[1])
```

```
Dimension: (1096, 102)
Number of rows: 1096
Number of columns: 102
```

2.2 Splitting Frames

In [ ]:
```
# Variablen initiieren
frame_count = 0
frames_until_reset = []

# Iterieren über Dataframe, um Frames mit '1' zu finden
for index, row in df_deepM01.iterrows():
    frame_count += 1

    if row["frame"] == 1 and frame_count > 1:
```

```
        frames_until_reset.append(frame_count -1)

if frame_count > 0:
    frames_until_reset.append(frame_count)

print("Number of frames until reset for each cycle:", frames_until_reset)
```

```
Number of frames until reset for each cycle: [210, 431, 652, 873, 1096]
```

In [ ]:

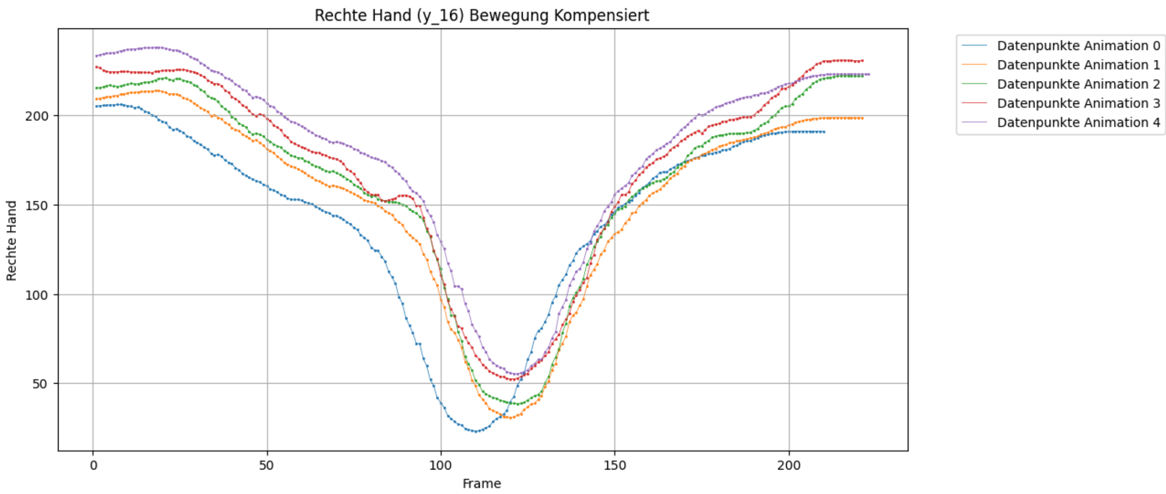
```
# Einzelne Bewegungen werden in verschiedene Dataframes gepackt
dfsKomp01 = []
start = 0
for end in frames_until_reset:
    dfsKomp01.append(df_deepM01.iloc[start:end])
    start = end

print("Number of splits:", len(dfsKomp01))
```

```
Number of splits: 5
```

2.3 Bewegungsanalyse anhand rechte Handbewegung (Kompensiert)

In [ ]:

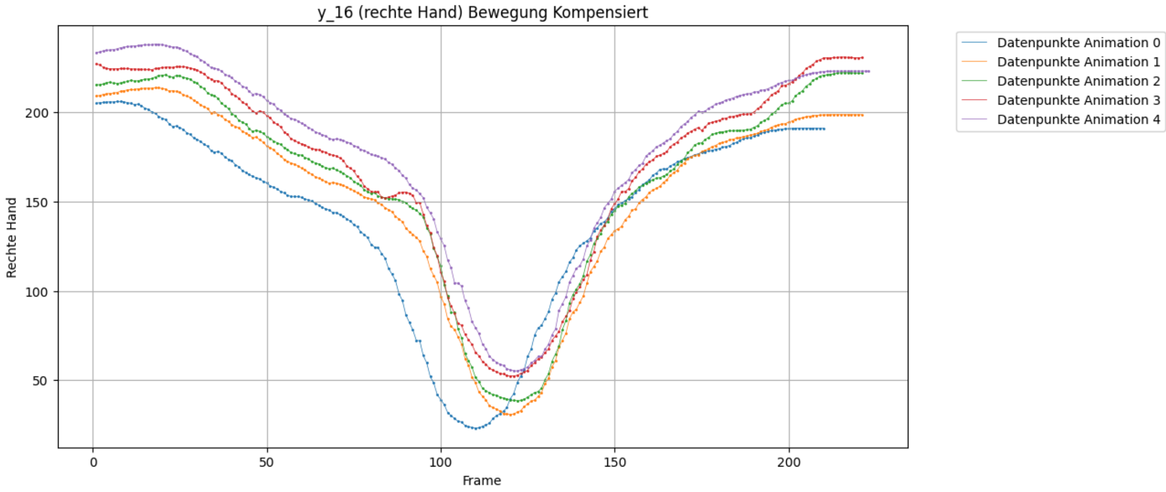
```
plt.figure(figsize=(12, 6))
for i, df in enumerate(dfsKomp01[:5]):
    plt.scatter(df["frame"], df["y_16"], marker='o', color=f'C{i}', s=1)
    plt.plot(df["frame"], df["y_16"], color=f'C{i}', linestyle='-', linew

plt.title("Rechte Hand (y_16) Bewegung Kompensiert")
plt.xlabel("Frame")
plt.ylabel("Rechte Hand")
plt.grid(True)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



In [ ]:

```
plt.figure(figsize=(12, 6))
for i, df in enumerate(dfsKomp01):
    plt.scatter(df["frame"], df["y_16"], marker='o', color=f'C{i}', s=1)
    plt.plot(df["frame"], df["y_16"], color=f'C{i}', linestyle='-', linew

plt.title("y_16 (rechte Hand) Bewegung Kompensiert")
plt.xlabel("Frame")
plt.ylabel("Rechte Hand")
plt.grid(True)
```

```
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



2.3.1 Berechnung des Mean

In [ ]:
```
means01 = []
mean_values_01 = df_deepM01[['frame', 'x_0', 'y_0', 'z_0', 'x_1', 'y_1',
                    'x_4', 'y_4', 'z_4', 'x_5', 'y_5', 'z_5', 'x_6', 'y_6'
                    'x_8', 'y_8', 'z_8', 'x_9', 'y_9', 'z_9', 'x_10', 'y_1
                    'x_12', 'y_12', 'z_12', 'x_13', 'y_13', 'z_13', 'x_14'
                    'x_16', 'y_16', 'z_16', 'x_17', 'y_17', 'z_17', 'x_18'
                    'x_20', 'y_20', 'z_20', 'x_21', 'y_21', 'z_21', 'x_22'
                    'x_24', 'y_24', 'z_24', 'x_25', 'y_25', 'z_25', 'x_26'
                    'x_28', 'y_28', 'z_28', 'x_29', 'y_29', 'z_29', 'x_30'
                    'x_32', 'y_32', 'z_32']].groupby('frame').mean()
means01.append(mean_values_01)
mean_values_01.head(5)
```

Out[ ]:

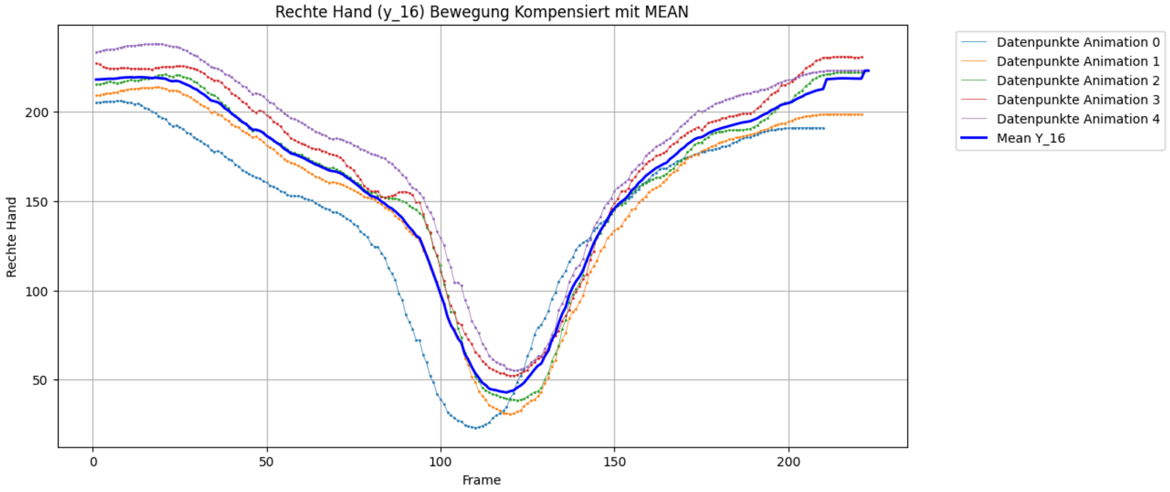
	x_0	y_0	z_0	x_1	y_1	z_1	x_2
frame							
1	42.9238	16.61496	-200.431752	-1007.957	-310.477	-48.49379	-1007.957
2	44.8634	16.47286	-202.449376	-1007.957	-310.477	-48.49379	-1007.957
3	46.3038	16.14494	-204.223192	-1007.957	-310.477	-48.49379	-1007.957
4	47.5600	15.77780	-206.149102	-1007.957	-310.477	-48.49379	-1007.957
5	49.5454	15.59410	-208.155064	-1007.957	-310.477	-48.49379	-1007.957

5 rows × 99 columns

2.3.2 Bewegungsmuster mit Mean

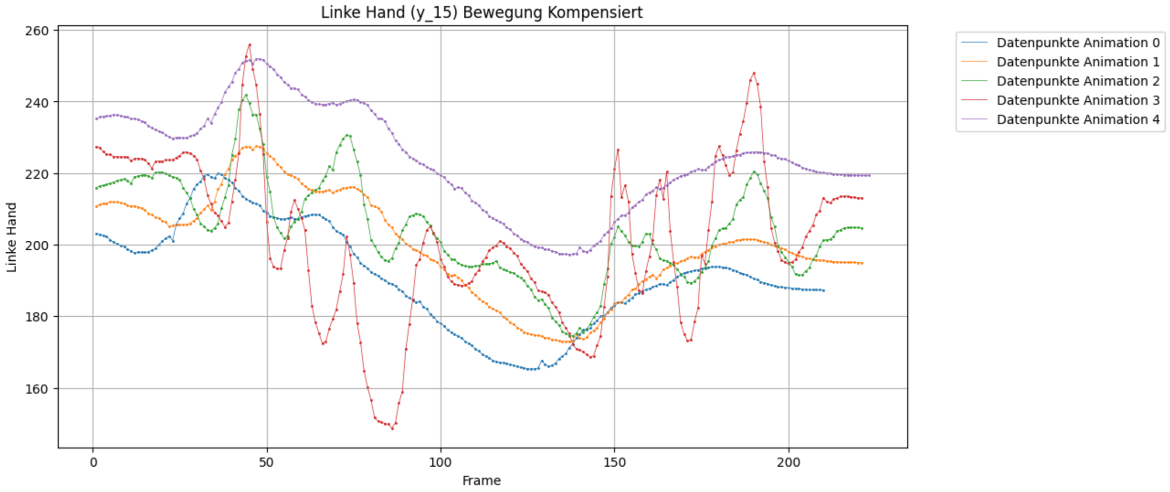
In [ ]:
```
plt.figure(figsize=(12, 6))
for i, df in enumerate(dfsKomp01):
    plt.scatter(df["frame"], df["y_16"], marker='o', color=f'C{i}', s=1)
    plt.plot(df["frame"], df["y_16"], color=f'C{i}', linestyle='-', linew

plt.plot(mean_values_01.index, mean_values_01["y_16"], color='b', linesty
```

```
plt.title("Rechte Hand (y_16) Bewegung Kompensiert mit MEAN")
plt.xlabel("Frame")
plt.ylabel("Rechte Hand")
plt.grid(True)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



2.4 Linke Handbewegung Kompensiert

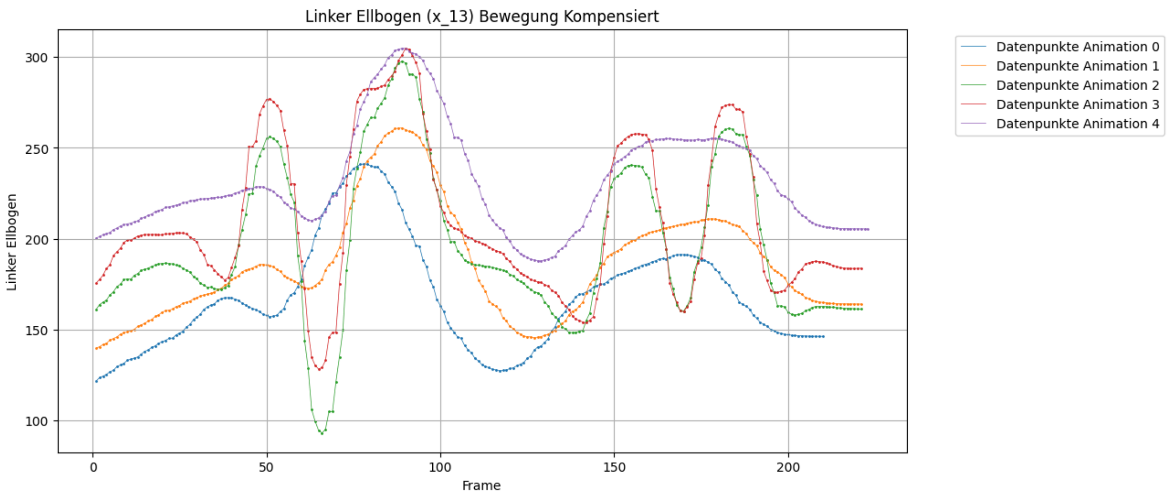
In [ ]:
```
plt.figure(figsize=(12, 6))
for i, df in enumerate(dfsKomp01):
    plt.scatter(df["frame"], df["y_15"], marker='o', color=f'C{i}', s=1)
    plt.plot(df["frame"], df["y_15"], color=f'C{i}', linestyle='-', linew

plt.title("Linke Hand (y_15) Bewegung Kompensiert")
plt.xlabel("Frame")
plt.ylabel("Linke Hand")
plt.grid(True)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



2.5 Linker Ellbogen Kompensiert

In [ ]:
```
plt.figure(figsize=(12, 6))
for i, df in enumerate(dfsKomp01):
    plt.scatter(df["frame"], df["x_13"], marker='o', color=f'C{i}', s=1)
```

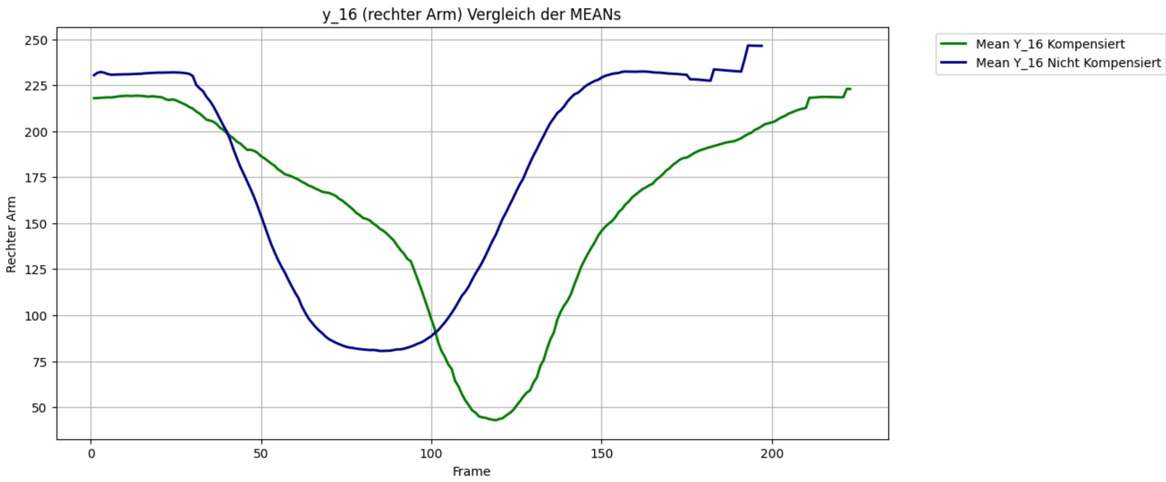
```
    plt.plot(df["frame"], df["x_13"], color=f'C{i}', linestyle='-', linew

plt.title("Linker Ellbogen (x_13) Bewegung Kompensiert")
plt.xlabel("Frame")
plt.ylabel("Linker Ellbogen")
plt.grid(True)
# Move the legend outside of the plot
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```



Berechnung Euclidean Distances

1. Analyse der Kompensierten und nicht Komptensierten Mean Werte

In [ ]:
```
plt.figure(figsize=(12, 6))

plt.plot(mean_values_01.index, mean_values_01["y_16"], color='green', lin
plt.plot(mean_values_03.index, mean_values_03["y_16"], color='darkblue',

plt.title("y_16 (rechter Arm) Vergleich der MEANs")
plt.xlabel("Frame")
plt.ylabel("Rechter Arm")
plt.grid(True)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



2. Vergleich Werte Kompensiert und Nicht Kompensiert anhand Euclidean Distances

In [ ]:

```
# Definition von Kolonnen
columns = ['frame', 'x_0', 'y_0', 'z_0', 'x_11', 'y_11', 'z_11',
           'x_12', 'y_12', 'z_12', 'x_13', 'y_13', 'z_13', 'x_14', 'y_14'
           'x_16', 'y_16', 'z_16', 'x_17', 'y_17', 'z_17', 'x_18', 'y_18'
           'x_20', 'y_20', 'z_20', 'x_21', 'y_21', 'z_21', 'x_22', 'y_22'
           'x_24', 'y_24', 'z_24', 'x_25', 'y_25', 'z_25', 'x_26', 'y_26'
           'x_28', 'y_28', 'z_28', 'x_31', 'y_31', 'z_31', 'x_32', 'y_32'

# Kalkulieren von euclidean Distance
euclidean_distances_03 = {}

common_frames = mean_values_03.index.intersection(mean_values_01.index)

for frame in common_frames:
    euclidean_distances_03[frame] = []
    for col in columns[1:]:
        point_1 = (frame, mean_values_03.loc[frame, col])
        point_2 = (frame, mean_values_01.loc[frame, col])
        euclidean_distance_03 = distance.euclidean(point_1, point_2)
        euclidean_distances_03[frame].append(euclidean_distance_03)

euclidean_df_03 = pd.DataFrame.from_dict(euclidean_distances_03, orient='
euclidean_df_03.head(5)
```

Out[ ]:

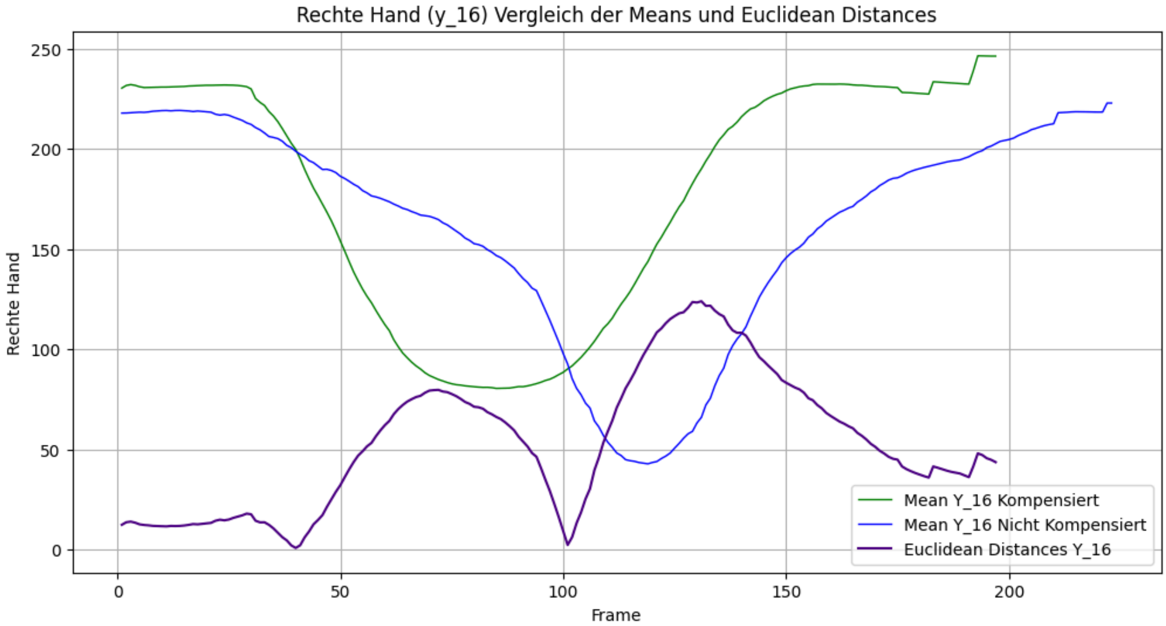
	x_0	y_0	z_0	x_11	y_11	z_11	x_12	y_12
1	23.06976	1.65790	28.964246	29.9736	0.00700	28.227944	29.95742	1.09204
2	21.99306	3.33414	31.195172	28.9674	1.57052	30.352582	28.95784	2.76372
3	21.68474	4.04454	33.314778	28.5382	2.19484	32.289940	28.52836	3.40398
4	21.13194	4.11502	35.643728	27.8812	2.23538	34.584982	27.87176	3.44928
5	20.08230	3.75600	38.089980	26.7930	1.78524	36.955010	26.78522	3.05044

5 rows × 63 columns

2.1 Vergleich Rechte Hand

In [ ]:

```
plt.figure(figsize=(12, 6))

# Plot the mean values for DeepMotion 01
plt.plot(mean_values_03.index, mean_values_03["y_16"], color='green', lin

# Plot the mean values for Mediapipe 01
plt.plot(mean_values_01.index, mean_values_01["y_16"], color='blue', line

# Plot the Euclidean distances for the last calculated column
plt.plot(euclidean_df_03.index, euclidean_df_03["y_16"], color='indigo',


plt.title("Rechte Hand (y_16) Vergleich der Means und Euclidean Distances
plt.xlabel("Frame")
plt.ylabel("Rechte Hand")
plt.grid(True)
plt.legend(loc='best')
plt.show()
```

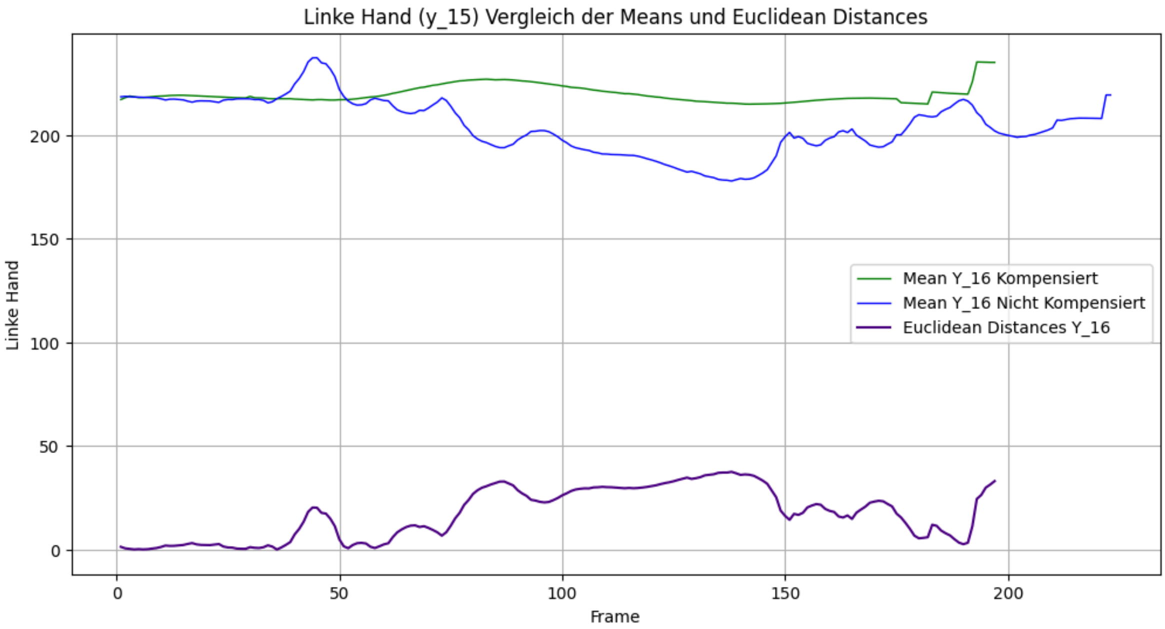


2.2 Vergleich Linke Hand

In [ ]:

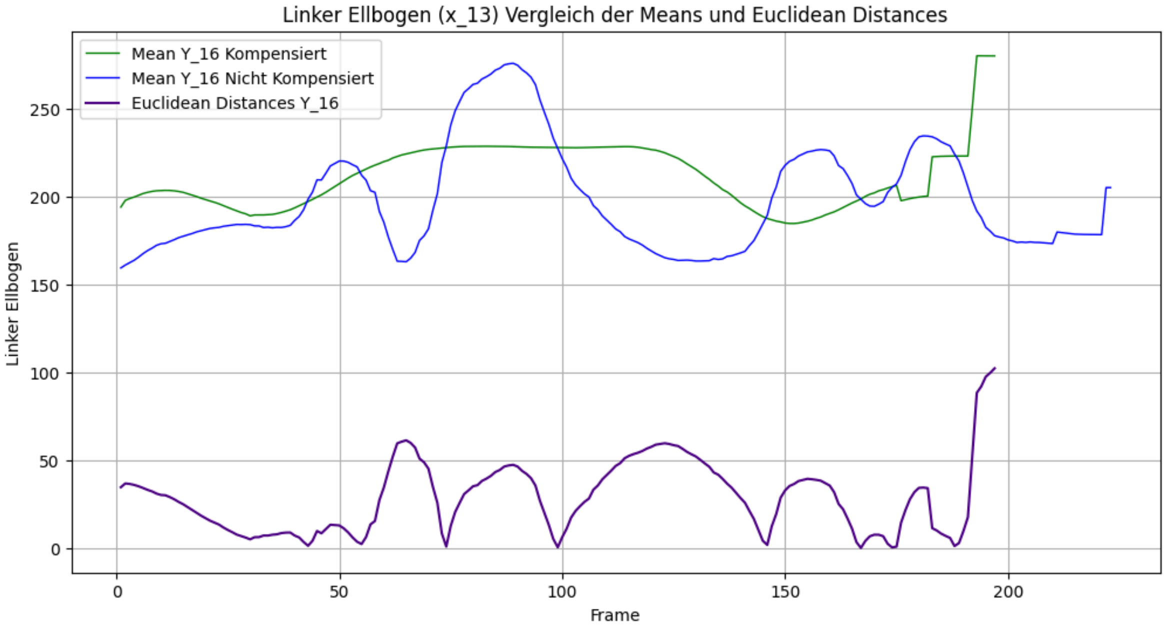
```
plt.figure(figsize=(12, 6))

# Plot the mean values for DeepMotion 01
plt.plot(mean_values_03.index, mean_values_03["y_15"], color='green', lin

# Plot the mean values for Mediapipe 01
plt.plot(mean_values_01.index, mean_values_01["y_15"], color='blue', line

# Plot the Euclidean distances for the last calculated column
plt.plot(euclidean_df_03.index, euclidean_df_03["y_15"], color='indigo',


plt.title("Linke Hand (y_15) Vergleich der Means und Euclidean Distances"
plt.xlabel("Frame")
plt.ylabel("Linke Hand")
plt.grid(True)
```

```
plt.legend(loc='best')
plt.show()
```



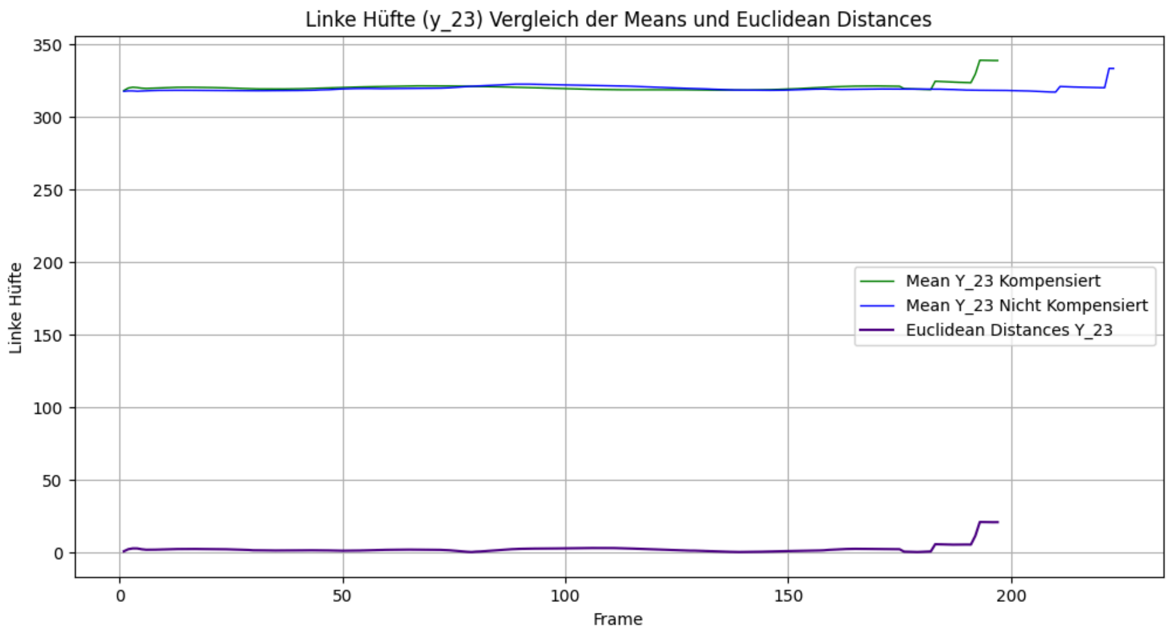
2.3 Vergleich linker Ellbogen

In [ ]:
```
plt.figure(figsize=(12, 6))

# Plot the mean values for DeepMotion 01
plt.plot(mean_values_03.index, mean_values_03["x_13"], color='green', lin

# Plot the mean values for Mediapipe 01
plt.plot(mean_values_01.index, mean_values_01["x_13"], color='blue', line

# Plot the Euclidean distances for the last calculated column
plt.plot(euclidean_df_03.index, euclidean_df_03["x_13"], color='indigo',


plt.title("Linker Ellbogen (x_13) Vergleich der Means und Euclidean Dista
plt.xlabel("Frame")
plt.ylabel("Linker Ellbogen")
plt.grid(True)
plt.legend(loc='best')
plt.show()
```



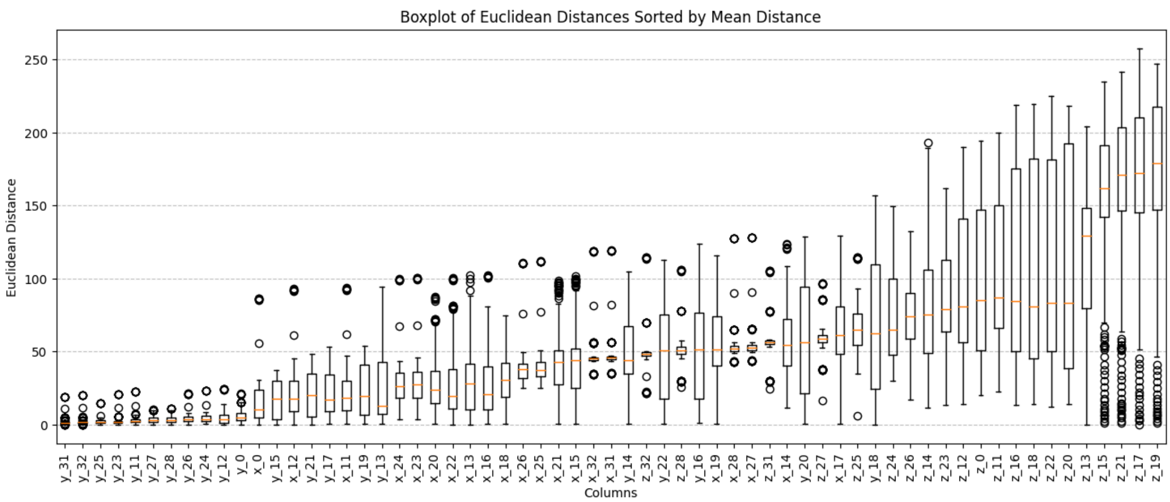
2.4 Vergleich linke Hüfte

In [ ]:
```
plt.figure(figsize=(12, 6))

# Plot the mean values for DeepMotion 01
plt.plot(mean_values_03.index, mean_values_03["y_23"], color='green', lin

# Plot the mean values for Mediapipe 01
plt.plot(mean_values_01.index, mean_values_01["y_23"], color='blue', line

# Plot the Euclidean distances for the last calculated column
plt.plot(euclidean_df_03.index, euclidean_df_03["y_23"], color='indigo',


plt.title("Linke Hüfte (y_23) Vergleich der Means und Euclidean Distances
plt.xlabel("Frame")
plt.ylabel("Linke Hüfte")
plt.grid(True)
plt.legend(loc='best')
plt.show()
```



3. Analyse Euclidean Distances

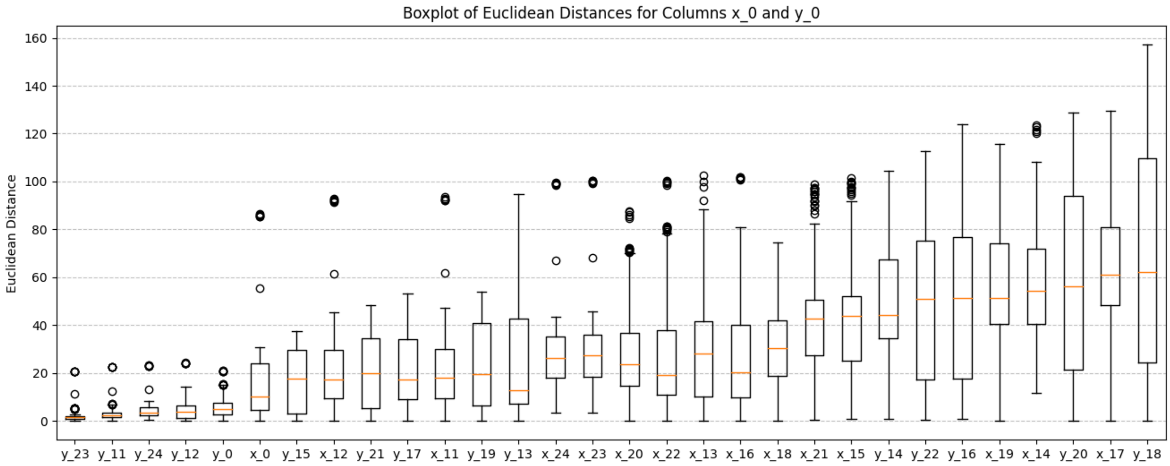
In [ ]:
```
mean_distances = euclidean_df_03.mean()
sorted_df = euclidean_df_03[mean_distances.sort_values().index]

plt.figure(figsize=(16, 6))
plt.boxplot(sorted_df.values)
plt.xticks(range(1, len(sorted_df.columns) + 1), sorted_df.columns, rotat
plt.title('Boxplot of Euclidean Distances Sorted by Mean Distance')
plt.xlabel('Columns')
plt.ylabel('Euclidean Distance')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



3.1 Analyse Euclidean Distances ohne Z Werte

In [ ]:
```
# Select only the 'x_0' and 'y_0' columns from the sorted DataFrame
selected_columns = ['x_0', 'x_11', 'x_12', 'x_13',
                    'x_14', 'x_15', 'x_16', 'x_17',
                    'x_18', 'x_19', 'x_20', 'x_21',
                    'x_22', 'x_23', 'x_24', 'y_0',
                    'y_11', 'y_12', 'y_13',
```

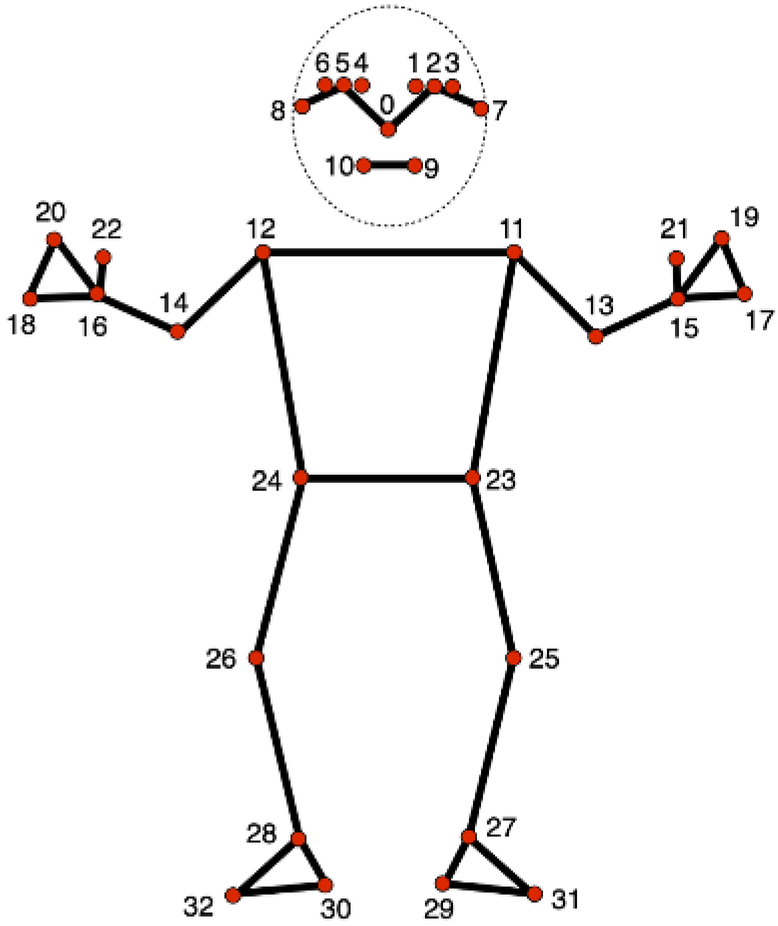
```
                    'y_14', 'y_15', 'y_16', 'y_17',
                    'y_18', 'y_19', 'y_20', 'y_21',
                    'y_22', 'y_23', 'y_24']

# Calculate the mean of Euclidean distances for each column
mean_distances = euclidean_df_03[selected_columns].mean()

# Sort the DataFrame by the mean distance in ascending order
sorted_df = euclidean_df_03[mean_distances.sort_values().index]

# Plotting the box plot
plt.figure(figsize=(16, 6))
plt.boxplot(sorted_df.values)
plt.xticks(range(1, len(sorted_df.columns) + 1), sorted_df.columns)
plt.title('Boxplot of Euclidean Distances for Columns x_0 and y_0')
plt.ylabel('Euclidean Distance')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



4. Vergleich ganzer Oberkörper

In [ ]:

```
# Inserting an image
img = plt.imread('Pictures/landmark.png')  # Replace 'landmark.png' with
im = OffsetImage(img, zoom=0.2)  # Adjust the zoom level as needed
ab = AnnotationBbox(im, (0.5, 0.5), frameon=False)
plt.gca().add_artist(ab)

# Remove axis
plt.axis('off')

plt.show()
```



In [ ]:

```
selected_columns = ['x_0', 'y_0', 'z_0', 'x_11', 'y_11', 'z_11',
                    'x_12', 'y_12', 'z_12', 'x_13', 'y_13', 'z_13',
                    'x_14', 'y_14', 'z_14', 'x_15', 'y_15', 'z_15',
                    'x_16', 'y_16', 'z_16', 'x_17', 'y_17', 'z_17',
                    'x_18', 'y_18', 'z_18', 'x_19', 'y_19', 'z_19',
                    'x_20', 'y_20', 'z_20', 'x_21', 'y_21', 'z_21',
                    'x_22', 'y_22', 'z_22', 'x_23', 'y_23', 'z_23',
                    'x_24', 'y_24', 'z_24']

difference_in_mean_upper_body_01 = mean_values_01[selected_columns].mean(
print(f"Mean Value: {difference_in_mean_upper_body_01}")
```

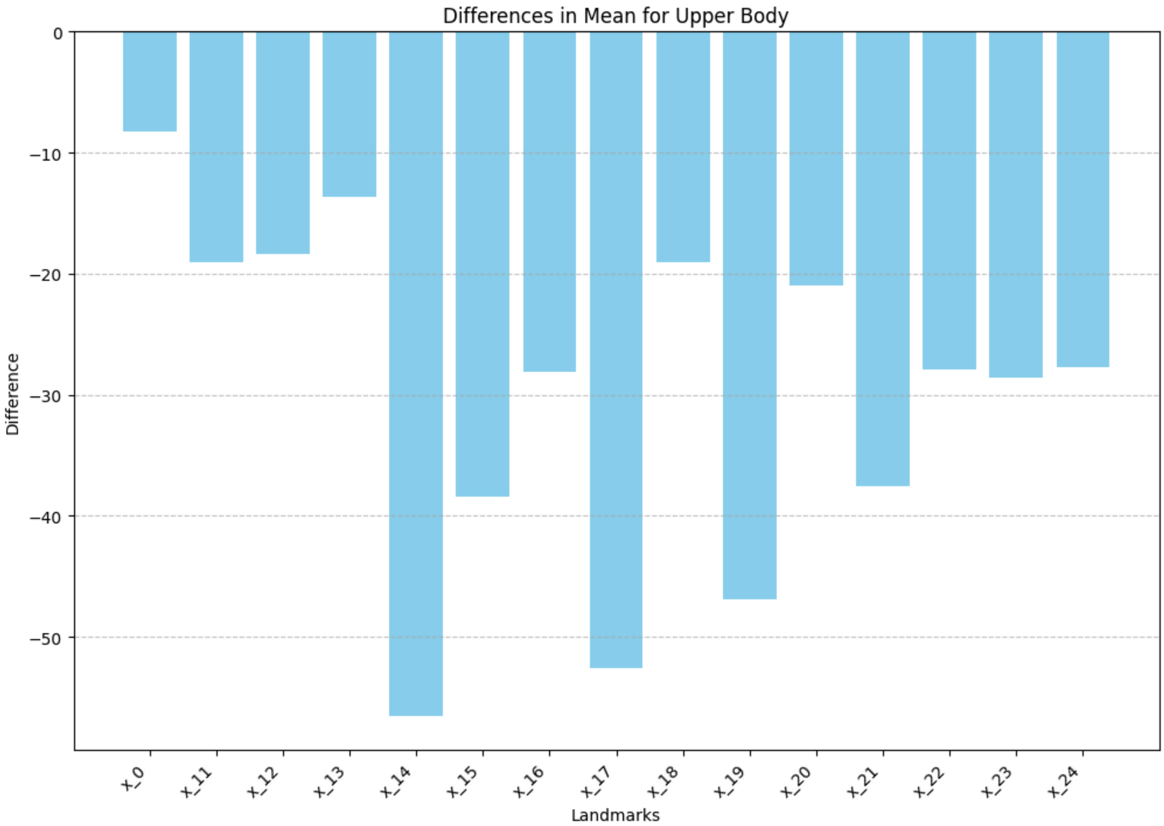
```
Mean Value: x_0      -8.223514
y_0        0.701673
z_0     -101.054790
x_11     -19.034095
y_11      -2.090782
z_11    -107.507648
x_12     -18.395724
y_12      -4.196192
z_12    -100.326219
x_13     -13.660687
y_13     -19.076607
z_13    -121.287961
x_14     -56.551723
y_14     -44.415245
z_14     -92.611904
x_15     -38.410071
y_15     -15.353725
z_15    -152.804474
x_16     -28.091003
y_16     -14.528532
z_16    -110.574064
x_17     -52.559885
y_17     -16.468250
z_17    -162.845611
x_18     -19.030388
y_18      -7.120281
z_18    -112.008680
x_19     -46.914996
y_19     -17.833418
z_19    -164.683528
x_20     -20.950228
y_20      -7.182865
z_20    -112.262405
x_21     -37.569543
y_21     -18.658648
z_21    -157.498633
x_22     -27.945124
y_22     -12.442283
z_22    -112.616766
x_23     -28.583976
y_23      -1.019616
z_23     -91.620590
x_24     -27.735380
y_24      -4.197384
z_24     -79.106428
dtype: float64
```

In [ ]:

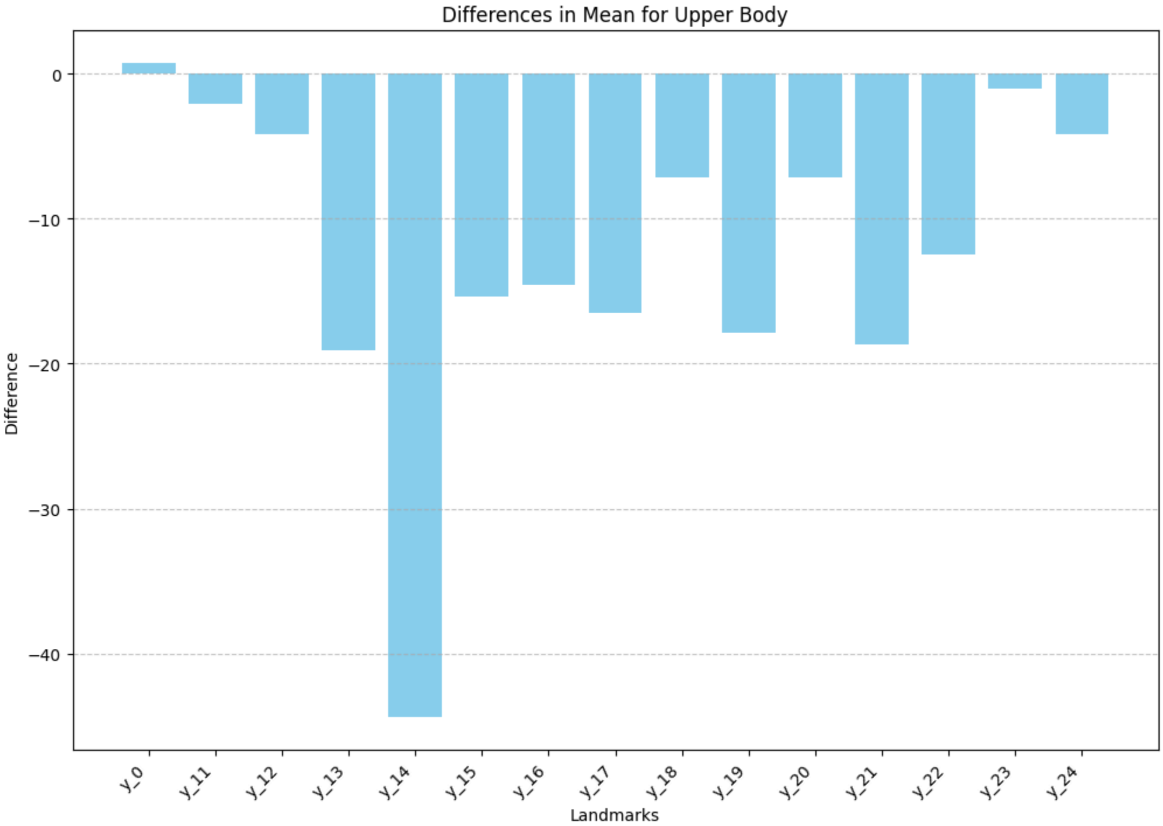
```
selected_columns_X = ['x_0', 'x_11', 'x_12', 'x_13',
                      'x_14', 'x_15', 'x_16', 'x_17',
                      'x_18', 'x_19', 'x_20', 'x_21',
                      'x_22', 'x_23', 'x_24']

selected_columns_Y = ['y_0', 'y_11', 'y_12', 'y_13',
                      'y_14', 'y_15', 'y_16', 'y_17',
                      'y_18', 'y_19', 'y_20', 'y_21',
                      'y_22', 'y_23', 'y_24']

selected_columns_Z = ['z_0', 'z_11', 'z_12', 'z_13',
                      'z_14', 'z_15', 'z_16', 'z_17',
                      'z_18', 'z_19', 'z_20', 'z_21',
```

```
                     'z_22', 'z_23', 'z_24']

difference_in_mean_upper_body_01_X = difference_in_mean_upper_body_01[sel
difference_in_mean_upper_body_01_Y = difference_in_mean_upper_body_01[sel
difference_in_mean_upper_body_01_Z = difference_in_mean_upper_body_01[sel
```

In [ ]:
```
plt.figure(figsize=(12, 8))
plt.bar(difference_in_mean_upper_body_01_X.index, difference_in_mean_uppe
plt.title('Differences in Mean for Upper Body')
plt.xlabel('Landmarks')
plt.ylabel('Difference')
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



In [ ]:
```
plt.figure(figsize=(12, 8))
plt.bar(difference_in_mean_upper_body_01_Y.index, difference_in_mean_uppe
plt.title('Differences in Mean for Upper Body')
plt.xlabel('Landmarks')
plt.ylabel('Difference')
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



In [ ]:
```
plt.figure(figsize=(12, 8))
plt.bar(difference_in_mean_upper_body_01_Z.index, difference_in_mean_uppe
plt.title('Differences in Mean for Upper Body')
plt.xlabel('Landmarks')
plt.ylabel('Difference')
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```