

A person's hand is shown holding a smartwatch. The watch screen displays a car status interface with text like 'Stato AUTONOMO', '14:08', '51 MI', and '87%'. A colorful, multi-colored circular graphic with a white location pin icon is centered over the watch. The background is dark and blurry, showing a person's face and glasses.

Gearheads

**Gesture recognition of doorknobs by a Smartwatch
and its application towards smart home**



PROBLEM OF EXISTING SYSTEM

OUR **IDEA**

FUTURE **FEASIBILITY**

CLASSIFIER **MACHINE LEARNING**

IMPLEMENTATION ON TIZEN

DEMO VIDEO

RESULT



P R O B L E M

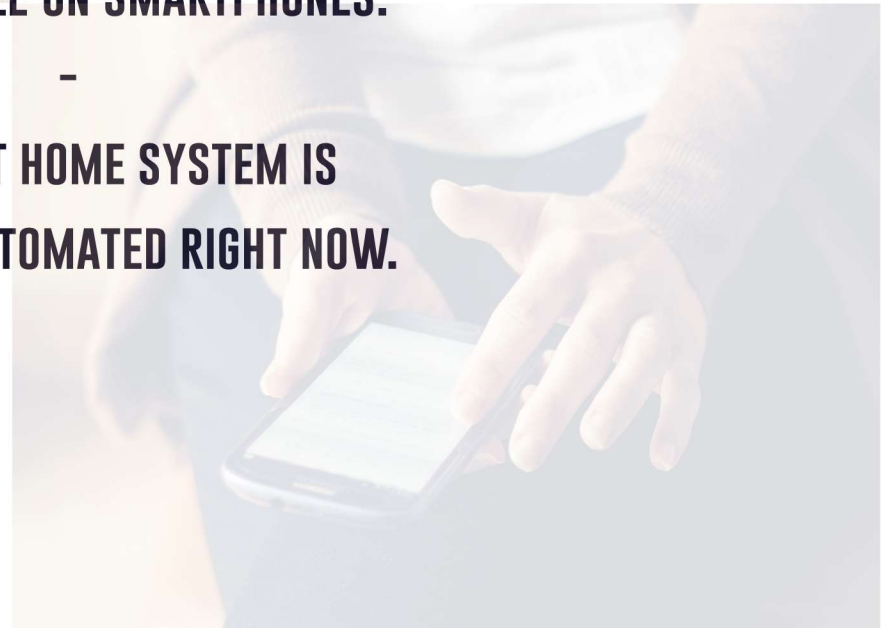
THE PROBLEM OF MODERN SMARTHOME SYSTEM



**MOST OF THE FUNCTIONS OF SMART WATCHES
ARE AVAILABLE ON SMARTPHONES.**

-

**THE SMART HOME SYSTEM IS
NOT FULLY AUTOMATED RIGHT NOW.**





P R O B L E M

THE PROBLEM OF MODERN SMARTHOME SYSTEM



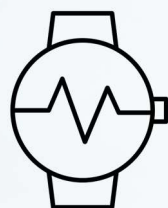
LIMITATION OF GEARS

THE NUMBER OF HOME APPLIANCES THAT ARE DIRECTLY CONNECTED



LIMITATION OF SMARTPHONE

LIMITATIONS TO RECOGNIZING USER BEHAVIOR



**MOTION
RECOGNIZE**



**CONTROL
ON/OFF**



**WITHOUT LIFTING A FINGER
TRIGGERED BY ONLY YOUR MOTION**

LET BE FREE



F E A S I B I L T Y

BE AVAILABLE FOR APPLICATION



LIGHT



AIR CONDITIONER



MUSIC

**IT CAN BE APPLIED IN ANY OTHER APPLIANCE
NOT ONLY LIGHT BULBS**

MACHINE LEARNING

HOW TO RECOGNIZE MOTION

Accel:[X] -0.883 Gyro:[X] -21.280
 Accel:[Y] 1.381 Gyro:[Y] 51.730
 Accel:[Z] 10.026 Gyro:[Z] -8.330
 Freq: 100.00
 Running...

Use Save Exit

SENSOR DATA ACCESS

no	x_mean	x_stddev	y_mean	y_stddev	z_mean	z_stddev	x_diff	y_diff	z_diff
1	-4.83073	0.58111	-5.61051	2.60353	5.75238	1.56259	3.72802	8.2361	5.62792
2	-4.65784	0.855771	-5.41122	2.56741	6.09231	1.55135	4.72583	10.7198	6.25962
3	-5.42003	0.725816	-4.97062	2.10173	6.07477	1.20334	3.75195	7.7671	4.85743
4	-5.46667	0.842872	-5.57786	2.02088	5.44852	1.39858	3.42892	7.43689	5.22353
5	-5.83317	0.67905	-4.45488	2.30496	5.9442	1.37338	2.80199	8.48256	5.202
6	-5.62692	0.759104	-5.35244	2.31011	5.41156	1.51391	2.90728	9.04248	5.22353
7	-5.79105	0.810506	-4.75942	2.35115	5.75356	1.35029	2.89053	7.17607	4.58943
8	-5.80271	0.93745	-4.89206	2.15215	5.72938	1.1824	3.19442	6.21416	4.87897
9	-5.90264	0.760777	-4.99385	2.04991	5.63987	1.09106	3.97687	7.85803	4.05823
10	-5.84036	0.860814	-5.14108	2.70795	5.23732	1.29696	3.30927	7.78624	4.29272
11	-5.58292	0.806653	-5.15403	2.66049	5.51262	1.33364	3.09392	8.40838	4.54397
12	-5.64827	0.744618	-5.7377	2.2806	4.97102	1.72674	5.63031	8.83909	6.6999
13	-5.48668	0.982986	-5.10589	2.73155	5.60389	1.28203	3.18245	7.85564	4.46979
14	-5.63705	0.781236	-5.77111	2.41639	4.81825	1.84567	3.1657	9.0664	6.43191
15	-5.69071	0.926935	-5.3493	2.69162	5.06645	1.80043	3.59402	9.40379	6.62333
16	-5.89022	0.869711	-4.84841	3.09238	5.17177	1.5334	5.97249	10.6552	5.98445
17	-5.82232	0.724589	-5.44775	2.48496	5.03683	1.53397	2.84507	9.2339	5.11585
18	-5.69429	0.83076	-5.11165	2.96764	5.143	1.79492	3.10588	9.41576	5.69492

DATA COLLECTING

```
In [29]: %matplotlib inline
import pandas as pd
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt
import sklearn
from sklearn.cross_validation import train_test_split
from sklearn.metrics import classification_report

train, test = train_test_split(dataset, train_size=0.7, random_state=0)
X_train, X_test = train.iloc[:, 1:13].values, test.iloc[:, 1:13].values
Y_train, Y_test = train.iloc[:, 13].values, test.iloc[:, 13].values
```

MACHINE LEARNING

```
In [31]: train, test = train_test_split(dataset, train_size=0.7, random_state=0)
X_train, X_test = train.iloc[:, 1:13].values, test.iloc[:, 1:13].values
Y_train, Y_test = train.iloc[:, 13].values, test.iloc[:, 13].values

In [32]: print (Y_train.shape, X_train.shape)
print (Y_test.shape, X_test.shape)

(152,) (152, 12)
(66,) (66, 12)

In [33]: model = GaussianNB()
model.fit(X_train, Y_train)

Out[33]: GaussianNB(priors=None)

In [34]: predicted= model.predict(X_test)
accuracy_score(predicted, Y_test)

Out[34]: 1.0
```

MACHINE LEARNING

HOW TO RECOGNIZE MOTION

```
In [8]: porter = Porter(model, language='java')
output = porter.export()
print(output)
```

```
2142857147, 0.5627115285714287, 3.4134760714285717, 5.690
825078571428572, 5.5277749999999992, 7.404720357142855));
double[] likelihoods = new double[11];

for (i = 0; i < 11; i++) {
    double sum = 0.;
    for (j = 0; j < 12; j++) {
        sum += Math.log(2. * Math.PI * sigmas[i][j]);
    }
    double nij = -0.5 * sum;
    sum = 0.;
    for (j = 0; j < 12; j++) {
        sum += Math.pow(atts[j] - thetas[i][j], 2.);
    }
    nij -= 0.5 * sum;
    likelihoods[i] = Math.log(priors[i]) + nij;
}

double highestLikeli = Double.NEGATIVE_INFINITY;
```

MACHINE LEARNING

MODEL EXTRACTION

CLASSIFIER IN TIZEN

```
atts[5] = stod(x, n);
atts[6] = minMax(x, n);
atts[7] = minMax(y, n);
atts[8] = minMax(z, n);
atts[9] = diff(x, y, n);
atts[10] = diff(y, z, n);
atts[11] = diff(z, x, n);
```

```
//2. Gaussian Naive Bayes Model.
```

```
// 1) Compute likelihood for each class
```

```
double nij = -0.5 * sum;
sum = 0.0;
for (j = 0; j < 12; j++) {
    sum += pow(atts[j] - thetas[i][j], 2.0) / sigmas[i][j];
}
nij -= 0.5 * sum;
likelihoods[i] = log(priors[i]) + nij;
}
```


IMPLEMENT

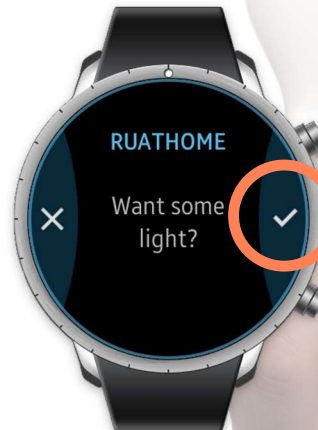
IN SMARTWATCH AND SMARTPHONE



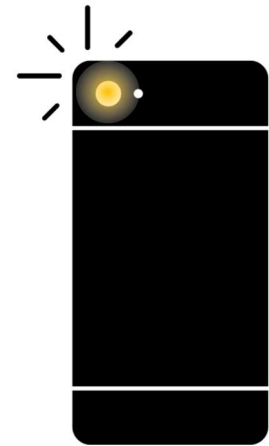
WAIT STATE



MOTION RECOGNITION



AMBIGUOUS SITUATION



LIGHT ON



DEMO VIDEO

SEE HOW IT WORKS

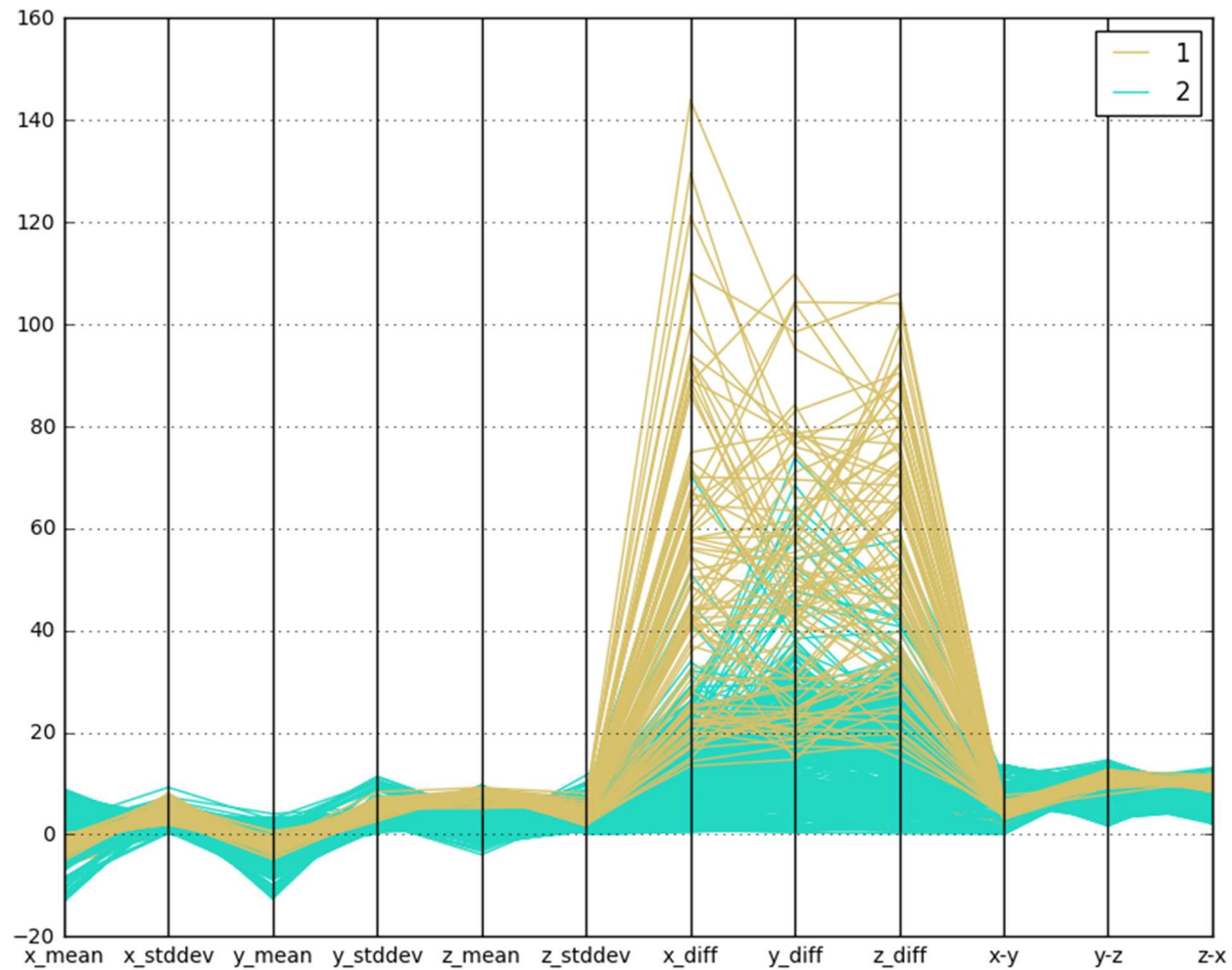


TEAM
GEAR
HEADS



R E S U L T

CORRECTNESS AND POSSIBILITY

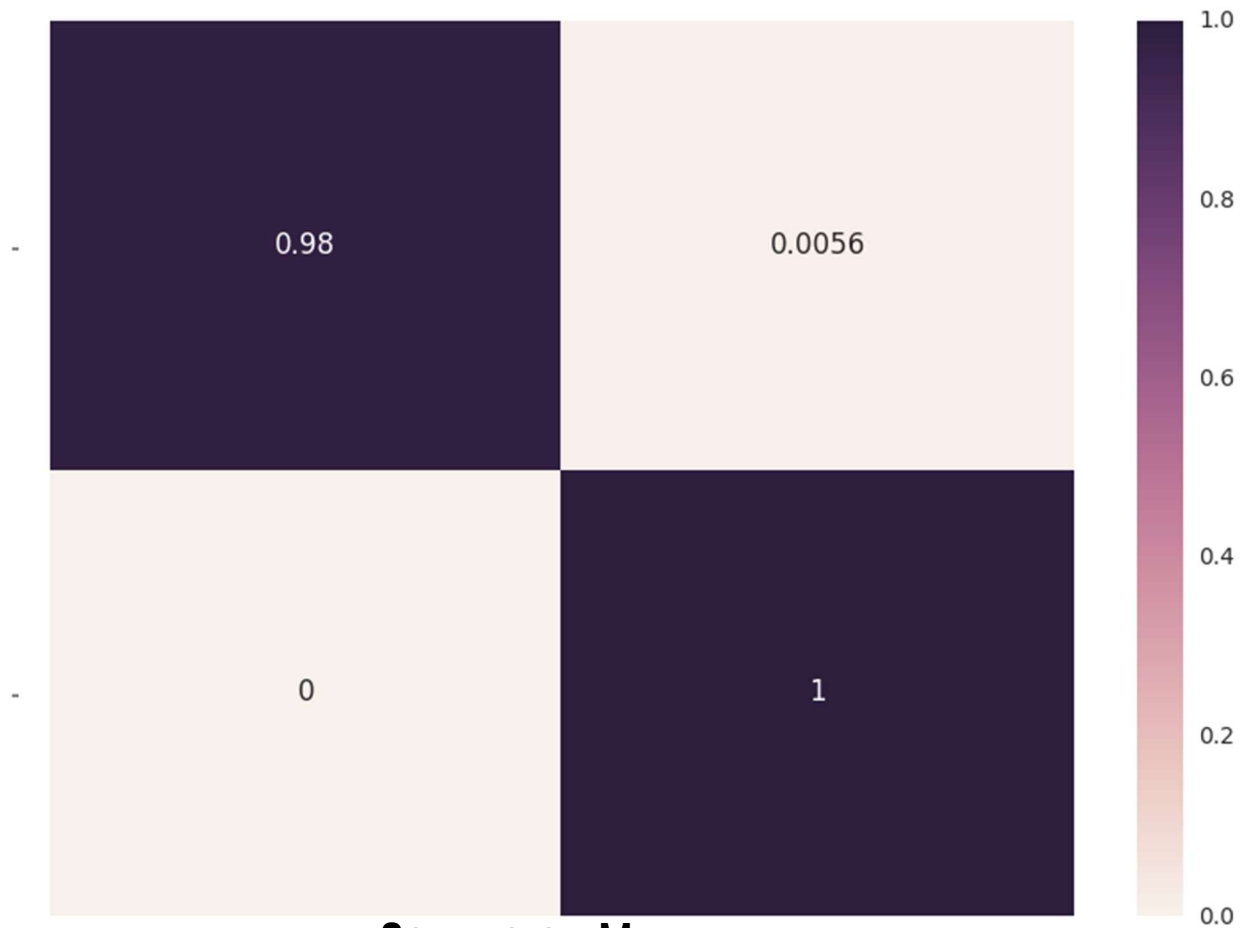


PARALLEL COORDINATES PLOT
(YELLOW IS OPEN MOTION, MINT IS OTHERS)



R E S U L T

CORRECTNESS AND POSSIBILITY



CONFUSION MATRIX
(TP IS 0.98, TN IS 1.0)

T E A M
G E A R
H E A D S





R E S U L T

CORRECTNESS AND POSSIBILITY

FREEDOM FROM INCONVENIENCE

FULLY AUTOMATED SMARTHOME

NOT ANYMORE SMARTWATCH AS WHITE ELEPHANT

T E A M
G E A R
H E A D S



