ANTICIPATING HUMAN ACTIVITIES FOR REACTIVE ROBOTIC RESPONSES

by

ATHANASIOS ANDRIANIS

Applied Informatics and Multimedia Engineer, Technological Educational Institute of Crete.

A THESIS

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

DEPARTMENT OF INFORMATICS ENGINEERING

SCHOOL OF APPLIED TECHNOLOGY

TECHNOLOGICAL EDUCATIONAL INSTITUTE OF CRETE

2016

Approved by:

Major Professor
Papadourakis Georgios

ABSTRACT

In this Master thesis the objective was to implement an activity recognition application in order to have the first step for activity anticipation, using MATLAB. The CAD-60 dataset. We implement four different ways with K-means used to divide in clusters the activities separately or the united dataset. Then each one of the training data was compared either by Euclidean distance of a testing activity or by matching the series of the characteristic poses. Each of the four ways was tested with multiple parameters and results was taken for all of them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

There are many people that helped me to accomplish, this thesis. First I would like to thank my professors Dr Dimitrios Kosmopoulos and Dr Georgios Papadourakis for their guidance, their valuable advice and their unlimited patience they had with me during this thesis. Mr. Kostas Papoutsakis for his help at the skeletal configuration part. My friend and colleague Manos Choustoulakis for his help and encouragement in many tasks during all the Master of Science program.

Last but not the least, I would like to thank people that even they had, absolutely, no idea what am I doing, they supported me with all their heart. My friends Filippos, Sifis, Areti, Nikos, Konstantina, Stella, Eugenia, Anna, Harris, Athena, Panagiotis, Michaela and Katerina.

DEDICATION

This thesis is dedicated to my parents Giorgos and Anna, my brother Manolis for supporting me all of my life in general.

To my alter ego, Maria.

Special dedicated to my beloved grandmother Vasileia. She knows why.

Αυτή η εργασία είναι αφιερωμένη στους γονείς μου Γιώργο και Άννα και στον αδερφό μου τον Μανώλη για την στήριξη τους σε όλη μου την ζωή.

Στην Μαρία μου.

Ειδικά αφιερωμένο στην αγαπημένη μου γιαγιά Βασιλεία. Αυτή ξέρει γιατί.

CHAPTER 1: INTRODUCTION

Activity recognition aims to recognize common human activities, actions and goals, of one or more persons, from a series of observations on them, depending also of the environmental conditions in real life settings. It is an important and promising technology in computer science because it can be applied to many sections of our everyday life, such as ambulation, transportation, daily activities, exercise and military activities and these can be separated to single person activities, interactions of several people and group activities.

It also has connections to many fields of study such as human and computer interaction, medicine and sociology. Applications, for all the above sections and problems, includes sign language recognition, healthcare, eldercare, surveillance systems, video analysis, robotics and a variety of systems that interactions between persons and electronic devices is needed, for example human-computer interfaces.

The anticipation of behaviors of others, for humans in everyday life, is taken for granted. However, this ability seems more easy that it is. We realize the importance of this nature-granted ability, when we want to develop robots that need to control their own movement and must predict the movement and action of humans, or other robots, that are around them in order to prevent unpleasant circumstances. So for changing their "social" behavior, it is crucial to anticipating the behavior of the other participants.

Robots can potentially be effective coworkers to humans in a large amount of applications. Some of them are industrial manufacturing, logistics, and personal healthcare services. For a personal robot to be capable for human assistance, it is important for it to be able to detect what a human is currently doing as well as anticipate what he is going to do next and how. The difficult questions revolve around how to generalize an appropriate robot action plan from an observed human action or determining what type of robot interaction is most effective in terms of assisting the human agent.

Prior to the release of Microsoft Kinect research has mainly focused on learning and recognizing actions from conventional two dimensional (2D) video. The introduction of low-cost integrated depth sensors, such as Microsoft Kinect, that can capture both RGB (Red, Green and Blue) video and depth (D) information has significantly advanced the research of human action recognition.[1]

Despite significant effort, recognizing actions is still remaining as a challenging task. With the recent advent of the Kinect, depth cameras have received great attention and excited interest, within the computer vision and robotics researchers for its applications.[2]

It was recently shown that conventional approaches based upon color sequences could not perform well on depth maps due to a large amount of false point detections fired on the spatio-temporally discontinuous regions. On the other hand, color sequences and depth maps, both have quite different properties. The descriptors are based on gradient, brightness and optical flow in traditional color sequences might be unsuitable for representing depth maps. It is easy to understand, to design action features according to the specific characteristics of depth sequences, e.g., cloud points.

The advantages of depth sensor over visible light camera are many. Firstly, the 3D structural information of the scene, provided from range sensor, offers more discerning, body shape and structure information, which has been successfully recovered skeleton joints from a single depth map, for recognizing actions and recovering postures. The low-level difficulties in RGB imagery are alleviated.

Secondly, color and texture are precluded in depth maps, which eases the problems of human detection and segmentation. Thirdly, the depth camera is insensitive to lighting change, which brings great benefits to the system monitoring in the dark environment

These advantages lead to interesting research such as estimating human skeletons from a single depth image and these are quite accurate under experimental settings and

[1] (Zhanga, et al., 2015)
[2] (Goles, 2010)

bring benefits to many applications including activity recognition. However, the algorithms are limited at the same time. It does not work properly when the human body is partly in view, and the guess is not reliable or when the person touches the background or when the person is not in an upright position there is a possibility of failure. In spatial surveillance, camera is usually elevated and the subjects are not in front the camera. These issues are causing difficulties for skeletal estimation. [3]

[3] (Xia, et al., 2013)

CHAPTER 2: FUNDAMENTALS

1. Computer Vision

Computer vision is a field in computer science that describes methods for acquiring, processing, analyzing, and understanding images and multi-dimensional data taken from the real world for producing numerical or symbolic information.[4]

The development of this field aims to enforce the human vision with discerning and understanding an image through computer algorithms. This means the transformation of visual images into descriptions of world that can interface with other processes and get information for the appropriate action. This image understanding can be considered as the disengagement of symbolic information from image data using models, which are constructed with the aid of other sciences, as geometry, statistics, physics, and learning theory. Computer vision has additionally been described as the initiative of automating and integrating many processes and representations for vision perception.

Also, computer vision is the theory, that lies behind artificial intelligence systems that takes information out from images. Computer vision seeks to apply its theories and models to the construction of computer vision systems for each one of the many forms of image data, such as video sequences, multiple cameras views and multi-dimensional data from a medical scanner.

Sub-domains of computer vision include reconstructing scenes, event detection, video tracking, object recognition, object pose estimation, learning, indexing, motion estimation, and image restoration.[5]

[4] (Klette, 2014)
[5] (Forsyth, et al., 2003) (Ballard, et al., 1982) (Sonka, et al., 2008)

2. Pattern Recognition

Pattern recognition is a subtopic of machine learning and is the study of how machines can observe the environment, learn to see the difference between, patterns of interest, make sound and reasonable decisions about the categories of the patterns. To this theory pattern is considered as a description of an object, so its recognition is the classification of a specific object to a pattern class.

Pattern recognition systems are in many cases trained from labeled "training" known as supervised learning, but when no labeled data are available other algorithms can be used to discover previously unknown patterns, which are best known as unsupervised learning.

Pattern recognition algorithms generally aim to provide a reasonable answer for all possible inputs by finding "most likely" matching outputs for the inputs, taking into account their statistical variation. [6]

3. RGBD-Kinect

Kinect is an input device, which uses a line of motion sensing produced by Microsoft for Xbox and Windows PCs. It was codenamed in development as Project Natal. Based around a webcam-style add-on peripheral, it enables users to interact with their console or computer without a game controller, through a natural user interface using gestures and spoken commands for control. Kinect competes with many motion controllers and its primary innovation, includes advanced gesture, facial and voice recognition.

Kinect sensor is a horizontal bar, which is connected to a small base with a motorized pivot and is designed to have a lengthwise position above or below the video display. The device features an RGB camera, depth sensor and microphone software,

[6] (Bishop, 2006) (Papadourakis)

which provides full-body 3D motion capture, facial recognition and voice recognition potential.

Kinect sensor's multi-array microphone, enables to localize the acoustic source and ambient noise suppression.

The depth sensor combines an infrared laser projector with a monochrome CMOS sensor, that can capture video data in 3D despite of ambient light conditions. The depth sensor can adjust its sensing range, and Kinect software is automatically calibrating the sensor based on gameplay and the physical environment, adapting to the presence of furniture or other obstacles, of the player. Kinect is capable of tracking up to six people, including motion analysis for two active players, with a feature extraction of 20 joints per player.[7]

4. Machine Learning

Machine learning is the field of study that provides to computers the ability for learning without being fully programmed. Machine learning techniques, studying and constructing algorithms that can learn from data and make predictions on them. The operation of those algorithms is to build a model from example inputs in order to make predictions or decisions driven from data, instead of strictly following static program instructions.

Machine learning is closely related to and often overlaps with computational statistics and often a discipline which also focuses in prediction-making through the use of computers. It is attached to mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning is employed for much computing tasks, where designing and programming explicit algorithms is infeasible. Can be used for applications like spam filtering, optical character recognition, search engines and computer vision.

[7] (Wilson, et al., 2009) (Pham, 2009) (Microsoft, 2009)

Machine learning tasks are typically classified into three broad categories, separated by, the nature of the input signal for learning or feedback available to a learning system. These are supervised learning, unsupervised learning and reinforcement learning.

The first category of machine learning task is supervised learning is. In supervised learning, each example is a pair consisting of an input object and an output value that is expected. A supervised learning algorithm's first step is to analyze the training data and then produces a justified function, which can be used for mapping new examples. The best and optimal scenario will allow the algorithm to determine correctly the class labels for unseen circumstances. This requires the learning algorithm to generalize and make decisions from the training data to situations, that never faced before, in a reasonable way.

Unsupervised learning is the next category of machine learning task of using a function for describing hidden structures from unlabeled data, so there is no error or final signal to evaluate a possible solution. This is the main difference between unsupervised learning and the other two categories, supervised learning and reinforcement learning.

Unsupervised learning is related to the problem of density estimation in statistics science. However unsupervised learning also includes and other techniques, which are used for preprocessing data, that can summarize and give details of key features of the data.

An example of unsupervised learning is clustering which will be analyzed at the next section.

Semi-supervised learning is a mid-category that lies Between supervised and unsupervised learning. Each example is an incomplete training signal which consists a training set, but with a part of the target outputs remains unknown.

Last category is reinforcement learning. A computer program interacts with a dynamic environment in which a certain goal must be performed, without determination when this goal is close to the output.[8]

[8] (Heikki, 1996) (Mitchel, 1997) (Mohri, et al., 2012)
http://www.britannica.com/technology/machine-learning

5. Clustering

Cluster analysis or clustering is the grouping a set of objects in such a way, that objects in the same group, which usually called as a cluster, are more similar to each other than to those in other clusters. It is a main task of exploratory data mining, and one of the most common techniques for statistical data analysis, and it is used in many fields, such bioinformatics, machine learning, image analysis, pattern recognition and information retrieval.

Cluster analysis itself is the general task and not one specific algorithm. It achievement can be emerged by various algorithms, that are different in their concept of what constitutes a cluster and how to find them more efficiently. Popular concepts of clusters include groups with small distances among the cluster members, thick areas of the data space, breaks or particular statistical distributions. Clustering can therefore be defined as a multi-objective optimization problem. The appropriate clustering algorithm and its parameter settings, depends on the individual data set and the future use of the results. Cluster analysis is not an automatic task, but a repetitive process of knowledge discovery or interactive multi-objective optimization in which containing trial, success and failure. Modification of data preprocessing, processing and model parameters is often necessary, until the result achieves the desired properties.[9]

[9] (Everitt, 2011)

CHAPTER 3: RELATED WORK

In robotics and computer vision field of studies there is much effort in the activity recognition and anticipation problem. The three basic steps which are common in every one of the efforts are, the feature extraction, the training of dataset and in the end, testing and results.

Feature extraction is the procedure in which, the arbitrary data, like text or images, are being transformed into formatted features, usually numerical. It can be used for machine learning algorithms and there are many techniques for extracting the features of an image or a video dataset.

One of the most popular of these techniques, is feature extraction from Kinect sensor. Data can be extracted from Microsoft Kinect, in the RGB and depth maps forms, meaning that features, are based on the estimated skeleton from Kinect Libraries. The 3D skeletal joint locations, due to the design of spherical coordinate system and the robust 3D skeleton estimation from Kinect, is invariant.[10]

The feature vectors for signal data such as postures read by a RGB-D sensor can be used for deriving descriptions of activity clusters learned using unsupervised methods. The machine can explain the trends in terms that humans comprehend. This may be used to develop more robust responsive behaviors during human-robot interaction.[11]

The RGB-D action datasets, are categorized into three categories.

- Single-view action in which each action is captured from a single specific view point,
- Multi-view action, where two or more viewpoints of each action are captured and
- Human-human interaction or multi-person activity, which consist of interactions between two people or activities performed by multiple persons.

[10] (Xia, et al., 2014) (Khosrowpour, et al., 2014) (Sung, et al., 2011) (Xia, et al., 2013)
[11] (Freedman, et al., 2015)

Note that in both single-view and multi-view datasets, each action/activity is performed by one actor at a time.[12]

The features that can capture meaningful characteristics of the person can also be extracted by using Histogram of Oriented Gradient(HOG). It can be combined with the Prime Sense skeleton tracking system[13] or it can be based on evaluating well-normalized local histograms in a dense grid. Distribution of local intensity gradients or edge directions can be characterizing local object appearance and shape, even without precise knowledge of the corresponding gradient or edge positions. In practice this can be implemented by the division the image window into small spatial regions for each cell gathering a local 1-D histogram of edge orientations or gradient directions, all over the pixels of the cell. The form of the final representation, come from the combined histogram entries.

It is also useful to contrast or normalize the local responses before using them, in order to have better invariance to illumination or shadowing. This can be done by gather the local histogram energy measurement, over larger spatial regions and after the results can be used for the normalization of all the cells in the block[14].

Another approach for feature representation is that a local cluster of extended surface normal, called as the polynormal, based on extended surface normals jointly encodes local motion and shape cues. The movements of skeleton joints can model human actions as have been indicated by biological observations. Using the configuration, speed, and acceleration of these joints can be calculated the moving pose descriptor. A complementary feature can portray the motion information, are the relative positions of pairwise joints. Instead of skeleton joints, the robustness of cloud points is bigger to noise and occlusion. In local occupancy patterns, cloud points, subdivided with the local 3D sub volumes and associated with skeleton joints, turn into a set of spatial grids. These grids are counting the number of cloud points falling, into each grid in order to sum, the low-level polynormals, in each adaptive spatio-temporal grid. So the final depth video

[12] (Zhanga, et al., 2015)
[13] (Sung, et al., 2011)
[14] (Dalal, et al., 2005)

representation formed by the connection of feature vectors, extracted from all space-time grids.[15]

Radon transform(RT) can be also used as a feature extractor due to its effectiveness in representing the shape characteristics. Its compact prominent shape and motion features into radon coefficients from 2-D binary images. One definition for the RT is the linear integration of an image over all parallel lines, which corresponds to the computation of the sum, of the matrix of image intensity, with the specified direction. The RT is used to represent the view structure of 2-D binary image and get directional features in the range of 0°–179°. RT for binary image, can be defined in discrete form as comprising of summing up the pixel intensities, taken along the lines at specified directions.[16]

One interesting approach is the feature extraction comes from Neural Networks. To effectively incorporate the motion information in video analysis, the preformation of 3D convolution in the convolutional layers of Convolutional Neural Networks so that distinguished features, from both dimensions, spatial and temporal, can be captured. Multiple distinct convolutional operations can be applied at the same location on the input, multiple types of features can be extracted.[17]

Finally features can be gathered as motion data from a user wearing specificated equipment such as, CyberGlove and for the measures of location and orientation of the hand is equipped with a Polhemus sensor. The start and end of a motion can be indicated by the user via a button on the CyberGlove's wrist.[18]

The training of the data is the second step of activity recognition. There are used many machine learning algorithms and ways for training the data, either supervised or unsupervised. Conditional Random Fields, Neural Networks, Markov Models, Support Vector Machines and clustering methods like Kmeans are some of these techniques.

[15] (Yang, et al., 2014)
[16] (Khan, et al., 2011)
[17] (Ji, et al., 2015)
[18] (Vamplew, et al.)

The labeling and segmentation of structured data, as sequences, trees and lattices can be described by a probabilistic framework known as Conditional random fields (CRFs). The idea behind is that of the definition of a conditional probability distribution, which lies, over label sequences given a specific observation sequence, rather than a joint distribution over both label and observation sequences.

CRFs are useful technique of machine learning and have been applied to a variety of applications. Combined with RGB-D data they have been applied to scene labeling and activity detection by grouping the frames into temporal segments after the discretizing of time to the frames of the video, and, where each one of them, spans a set of contiguous frames, which corresponds to a single sub-activity. For the past frames, the structure of the CRF is known despite of the labels of the nodes in the CRF.

There have been efforts by modeling the human activities as above and object affordances in the past using a CRF, and its extension to include future possible scenarios. Each possibility was represented as a potential labeled graph structure, which includes discrete labels as well as human and object trajectories, which is called anticipatory temporal conditional random field (ATCRF). These sampling techniques are used for constructing possible future activities, and estimate the most likely future scenarios[19]. This technique expanded by using anticipation of future activities, so that a robot could perform look ahead planning, of its reactive responses showing that anticipation can improve performance of detection of even past activities and affordances.[20]

Furthermore, a method proposed to first acquire potential graph structures that are similar to the ground-truth ones, by approximating the graph with only additive features. The space of likely graph structures is explored by designed moves, which also could, to anticipate the future activities where considering multiple possible graphs is critical [21]. By adding an extra layer of latent variables, the model becomes more flexible and therefore, it can be used for modeling more complex data. So a novel Hidden-state CRF model for human activity recognition. Latent variables were used, to exploit the

[19] (Koppula, et al., 2013)
[20] (Koppula, et al., 2014)
[21] (Koppula , et al., 2013)

underlying structures of the target states. By making the observation and state nodes fully connected, the model does not require any conditional independent assumption between latent variables and the observations. The model is very efficient in that the inference algorithm is applied to a linear-chain structure[22]. All of the above models, who are using CRFs, are very general and they can be easily extended for other prediction tasks on sequential data.

Another way of training data is the Markov models. Markov model is a stochastic model used to describe randomly changing systems where the assumption that the future states depend only on the present state and not on the sequence of events that preceded it. A hidden Markov model models the state which is only partially observable of a system with a random variable that changes through time. In activity recognition algorithms can be based on hierarchical maximum entropy Markov model (MEMM), in which the person's activity considering, as composed of a set of sub-activities. An RGBD sensor used as the input sensor, then a two-layered maximum entropy Markov model which modeled different properties of the human activities. In this model are included its hierarchical nature, the transitions between sub-activities over time, and the relation between sub-activities and different types of features [23]. A similar approach uses a Kinect sensor that detects body postures in real-time. Furthermore, the Hidden Markov Model (HMM) labels are sequences of actions divided, into temporal segments and classifies the observed actions automatically. Experimental results showing that HMMs, promising applicability for real-world activity analysis[24].

A Support Vector Machine (SVM) is a distinction classifier, formally defined by a separating hyperplane. In other words, given labeled training data, the output of the algorithm is an optimal hyperplane which new examples are categorized. It is an another supervised learning technique for activity recognition and anticipation. Multiclass SVM's are used for training the system for the classification of the data. A matrix was generated for the entire training data set e.g. a video sequence. Each video is separated in frames

[22] (Hu, et al., 2014)
[23] (Sung, et al., 2011)
[24] (Khosrowpour, et al., 2014)

and each frame, can be represented by a row of the matrix and the dimension of SVM training matrix was adjustable to the needs of each training data set [25]. The SVM can be re-trained using another dataset of specified data. The set of retraining data can be subsampled, so that the final training set fit to the SVM training. This retraining process can improve the performance of the system [26].

A much sophisticated approach of supervised training is the artificial neural networks. The simplest definition of an artificial neural network, is a computing system composed by simple, highly interconnected processing elements, which process information, response to external inputs[27]. ANNs are processing devices, algorithms or hardware, that are modeled based on the neuronal structure of the mammalian cerebral cortex but on much smaller scales. These can be separated in many different types that each one has its own advantages and disadvantages.

One of them is Convolutional neural networks (CNNs) are one of the types of deep models that can act directly on the raw inputs, so the process of feature construction is automated. However, such models are currently limited to handle 2D inputs but a novel 3D CNN model was developed for action recognition and trained using supervised algorithm in this work, and it required a large number of labeled samples. Multiple channels, representing the information from the input frames, are generated by this model, and the final feature representation is obtained by combining information from all channels [28]. Recurrent networks and feedforward networks used also for training data in activity recognition applications. The training of these networks was done by the backpropagation through time algorithm.[29]

Neural Networks can also combine with other techniques for training the dataset. Artificial Neural Networks (ANN) is used to recognize different human activities in real

[25] (Biswas, et al., 2011)
[26] (Dalal, et al., 2005)
[27] (Caudill, 1989)
[28] (Ji, et al., 2015)
[29] (Vamplew, et al.)

time. After the feature extraction the training and recognition process is performed by using Naive Bayes, AdaBoost and k-NN and neural network classifiers.[30]

Training of data in an activity recognition system can be done by unsupervised algorithms as well. K-means clustering is the simplest and most-used clustering algorithm. K-means can cluster the set of all sensor readings into k clusters and that consists the training data. The signal input at the center of each cluster is used to represent all sensor readings that belongs to that specific cluster. This means that any novel signal input is aliased with one from the training data so that a misclassified action is certain [31]. Also the extraction of 3D skeletal joint locations from Kinect depth maps can be mapped and clustered into k posture visual words, which represent the prototypical poses of actions. The temporal evolutions of those poses are modeled by discrete hidden Markov models [32] or in each cluster can be trained a linear support vector machine and it can be efficiently solved by stochastic gradient descent. [33]

Finally, each method is tested and result are taken from comparison and discuss for improvement. Starting from Neural Networks techniques, the 3D CNN model achieves an overall accuracy of 90.2% [34] and the radon transform system model using ANN classifier resulted in an average recognition rate of 78.0% in real time data[35]. About the recurrent neural network it was necessary to choose a threshold for more suitable performance and in both of these categories the result was, around 96-97% accuracy[36].

About the support vector machine models, it is shown that, it's possible to classify hand gestures, using simply the depth images. For illustration the 8 gestures which were picked up, where all recognized [37], so it is possible a similar implementation for activity recognition and anticipation. About the HOG cell which tested four times with different

[30] (Khan, et al., 2011)
[31] (Freedman, et al., 2015) (Tatiraju, et al., 2011)
[32] (Xia, et al., 2014)
[33] (Gu, et al., 2012)
[34] (Ji, et al., 2015)
[35] (Khan, et al., 2011)
[36] (Vamplew, et al.)
[37] (Biswas, et al., 2011)

normalizations and including repeated information, the improved performance was from 84% to 89%[38].

At the CRF models the accuracy was 85.4% for affordance, 70.3% for sub-activity labeling and 83.1% for high-level activities respectively for detection. Furthermore, the accuracy which obtained was 67.2% and 49.6% for anticipating affordances and sub-activities respectively in future time-frames[39], at Learning Latent Structure for Activity Recognition the best performance get 89,2% on the average precision and 83,1% on the average recall [40], and finally in ATCRF model, an activity anticipation accuracy of 75.4%, 69.2% and 58.1% for different anticipation times of 1, 3 and 10 seconds respectively, obtained [41].

In Hidden Markov models the experimental results for the HMM along with the KDE was performed of 81% and 70% for atomic action classification[42] and for the MEMM algorithm the achievement was an average performance of 84.3% when the person was previously seen in the training set, and 64.2% when the person was not seen before in the training set [43].

Finally, the results with Kmeans usage while the initialization of the cluster centroids was random and combined with the HMM algorithm, by experiments with multiple Ks, the overall mean accuracy is 90.92%, the best accuracy is 95.0%[44]. Furthermore when K-means is combined with a fine-tuned kernel SVM in terms of prediction performance, the best results was 96.44% [45].

[38] (Dalal, et al., 2005)
[39] (Koppula , et al., 2013)
[40] (Hu, et al., 2014)
[41] (Koppula, et al., 2013)
[42] (Khosrowpour, et al., 2014)
[43] (Sung, et al., 2011)
[44] (Xia, et al., 2014)
[45] (Gu, et al., 2012)

CHAPTER 4: OUR APPROACH

Our approach for the recognition of human activities has been separated in 4 steps:

1. Choosing dataset and import into matlab
2. Configure skeleton Features into 3d distances
3. Train dataset with K-means
4. Evaluate with testing data

We have chosen MATLAB for this project because it allows large data input, easy matrix manipulations, plotting of functions and data, ready implementation of algorithms like K-means and pdist and has the ability of the creation of user interfaces.

At the particular chapter we will analyze each one of the above steps as well as the algorithms and the metric techniques that used for the training of the dataset and the evaluation.

Algorithms and metric techniques

1. K-means

K-means clustering is a very popular method for cluster analysis. This clustering algorithm aims to separate a number of observations into a number equals of k clusters, in which everyone observation, belonging to the cluster with the nearest mean, is considered as a prototype of the specific cluster.

K-means is one of the simplest unsupervised learning algorithms. The procedure follows a simple way to sort a given dataset through a certain number of clusters fixed by deduction. The main idea is the definition of k centroids, one for each cluster, placed in a cunning way because the cause of different result due to of different location. The next step is to take each point which belongs to the given dataset and match it to the nearest centroid. When all points are not pending, the first step is completed and the first groupage is completed. Then we re-calculate the k new centroids as barycenter of the clusters we took as result, from the previous step. After we have these k new centroids, a new sort has to be done between the same dataset points and the nearest new centroid

so this process is generating a loop. As a result of this loop, the k centroids change their location step by step, until no more cluster changes are needed.

Finally, the aim of this algorithm is the minimize of an objective function, in this case a squared error function. The objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left(x_i^{(j)} - c_j \right)^2,$$

where $\left(x_i^{(j)} - c_j \right)^2$ is a chosen distance measure between a data point $x_i^{(j)}$ from the dataset, and the cluster center c_j , is an sign of the distance of the *n* data points from their each cluster centers.

The algorithm is made up of the following steps:

1. *Place K points, which are representing initial group centroids, into the space taken by the objects that will be clustered.*
2. *Assign each object to the group that has the closest centroid.*
3. *When all objects have been assigned, recalculate the positions of the K centroids.*
4. *Repeat Steps 2 and 3 until the centroids change no more. This produces a separation of the objects into groups from which the metric can be calculated when is minimized.*

Although the procedure ends always, the k-means algorithm does not necessarily find the most optimal configuration, because of the global objective function minimum. The algorithm is also sensitive to the initial cluster centers which are selected randomly. For the reduction of this effect the k-means algorithm could be run multiple times.[46]

K-means clustering has been used as a feature learning step, in both supervised learning or unsupervised learning. The first step of the procedure is to train a k-means clustering representation, using the input training data which don't need to be labelled. Then, to protrude any new data into the new feature space, we can use the matrix-product, which has the threshold, of the data with the centroid locations, the distance

[46] http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/kmeans.html

from the data to each centroid, or simply a function for the nearest centroid or some transformation of the distance.

Calculating the number of clusters in a data set, is an often problem in data clustering, and is a different issue from the whole process of solving the clustering problem.

The correct choice of the number of the clusters is often uncertain, with acts which depends on the shape and scale of the distribution of points in a dataset and the desired clustering resolution of the user. In addition, increasing the number of the clusters, without penalty, will reduce the presentence of error in the resulting clustering. The optimal choice of clusters, will give the impression of the harmony between maximum compression of the data using one cluster, and maximum accuracy of having each data point to its own cluster.

If an appropriate value of number of clusters is not obvious from previous knowledge of the dataset, it must be chosen somehow. There are several methods for making this decision.[47]

The Elbow Method is a method that takes a look at the percentage of variance clarified as a function of the number of clusters. Percentage of variance is the ratio of the between-group variance to the total variance. One should choose a number of clusters so that adding another one cluster does not give better data modeling. More precisely, if the plot of the percentage of variance caused by the clusters, divided by the number of clusters, the first set of clusters will add much information, but at an unspecified moment, the marginal gain will drop, forming an angle in the graph. The number of clusters is chosen at this point, therefore the elbow criterion. This elbow cannot always have been identified.

The average silhouette of the data is another criterion for evaluating the natural number of clusters. The silhouette of a datum is a measure of how closely it is matched within its cluster and how loosely it is matched of the neighboring cluster. A silhouette close to 1 insinuates that, the datum is in a suitable cluster. Unlike a silhouette close to -

[47] (Coates, et al., 2012)

1 implies the datum is in the wrong cluster. Optimization techniques, like genetic algorithms, can determine the number of clusters that increases to the largest silhouette. It is also possible to re-scale the data, in a way that the silhouette can be maximized at the optimal number of clusters.[48]

2. Euclidean Distance

In mathematics, the Euclidean distance or Euclidean metric is the ordinary distance between two points in Euclidean space. With this distance, Euclidean space becomes a metric space.

The Euclidean distance between points p and q is the length of the line segment connecting them ($\overline{pq}$). In Cartesian coordinates, if p = (p_1, p_2..., p_n) and q = (q_1, q_2..., q_n) are two points in Euclidean *n*-space, then the distance (d) from p to q, or from q to p is given by the Pythagorean formula which its general type is:

$$d(p,q) = d(q,p) = \sqrt{(q_1 - p_1) + (q_2 - p_2) + \cdots + (q_n - p_n)} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

In general, for an *n*-dimensional space, the Euclidean distance is :

$$d(p,q) = \sqrt{(q_1 - p_1) + (q_2 - p_2) + \cdots + (q_n - p_n)}$$

[49]

3. Levenshtein Distance

The Levenshtein distance is a string metric for measuring the difference between two sequences. It is named after Vladimir Levenshtein, who considered this distance in 60s. Informally, the Levenshtein distance between two words is the minimum number of single-character edits like insertions, deletions or substitutions, which are required to change one word into the other

For example,

[48] (Cordeiro de Amorim, et al., 2015)
[49] (Deza, et al., 2009)

- If the first string is "test" and the second is "test", then Levenshtein distance is 0, because no transformations are needed. The strings are already identical.
- If the first string is "test" and the second is "tent", then Levenshtein distance is 0, because one substitution (change "s" to "n") is sufficient to transform the first string into the second.

The greater the Levenshtein distance, the more different the strings are. The Levenshtein distance algorithm can be found in multiple applications like spell checking, speech recognition, DNA analysis and plagiarism detection.

Levenshtein distance may also be referred to as edit distance, although that may also denote a larger family of distance metrics.[50]

4. Wagner-Fischer algorithm

The Wagner–Fischer algorithm computes edit distance based on the observation that if we reserve a matrix to hold the edit distances between all prefixes of the first string and all prefixes of the second, then we can compute the values in the matrix by flood filling the matrix, and thus find the distance between the two full strings as the last value computed [51].

At the following lines we will analyze the steps we did to implement this application.

Step 1. Choosing dataset and import into MATLAB

For this project, the chosen dataset was Cornell Activity Dataset 60(CAD 60). The CAD 60 data set comprise of RGB-D video sequences of humans performing activities

[50]http://people.cs.pitt.edu/~kirk/cs1501/Pruhs/Fall2006/Assignments/editdistance/Levenshtein%20Distance.htm
[51] (Woltzenlogel Paleo, 2007)

which are recording using the Microsoft Kinect sensor. It produced from Jaeyong Sung, Colin Ponce, Bar Selman, Ashutosh Saxena from Cornell University[52].

RGBD data has resolution of 240 by 320. RGB is saved as three-channel 8-bit PNG file. And, Depth is saved as single-channel 16-bit PNG file. Due to alignment of Depth and RGB data, some pixels on the edges will have value of 0. Refer to "Feature Extraction" in Sung et al. code below to look at how to parse these PNG files.

Figure 1:Sample of CAD 60 Data Set

CAD-60 dataset features are 60 RGB-D videos with 4 subjects-persons. Two male, two females, one left-handed.

Figure 2:Person 1. Male, right handed

[52] (Sung, et al., 2011)

Figure 3:Person 2. Female, right handed

Figure 4:Person 3. Female, left handed

Figure 5:Person 4. Male, right handed

It was taken to 5 different environments: office, kitchen, bedroom, bathroom, and living room, for 14 activities: rinsing mouth, brushing teeth, wearing contact lens, talking on the phone, drinking water, opening pill container, cooking (chopping), cooking (stirring), talking on couch, relaxing on couch, writing on whiteboard, working on computer, staying still, and random moves.

It can be downloaded from http://pr.cs.cornell.edu/humanactivities /data.php#data in the form of zipped files. Each zipped directory contains following four types of files.

- activityLabel.txt: A file that maps each # to corresponding activity. # is a ten-digit integer
- #.txt: Skeleton data files and skeleton data format.

Skeleton data consists of 15 joints. There are 11 joints that have both joint orientation and joint position. And, 4 joints that only have joint position. Each row follows the following format:

Frame#,ORI(1),P(1),ORI(2),P(2),...,P(11),J(11),P(12),...,P(15)

Frame# => integer starting from 1

ORI(i) => orientation of ith joint, is a 3x3 matrix and followed by CONF (0,1,2,3,4,5,6,7,8, CONF)

P(i) => position of ith joint followed by CONF(x,y,z,CONF)

All the above values are in millimeters.

CONF, is a confidence value who is returning for the individual joints but it is fairly basic at the moment. If it is 1, tracking seems to work, 0 if tracking fails and the output is uncertain. Note that if the confidence is 0 there is no guarantee the corresponding position/orientation is meaningful. Sometimes it will be kept at its old value, but other times it might be meaningless.

Joint Number	Joint name
1	HEAD
2	NECK
3	TORSO
4	LEFT_SHOULDER

5	LEFT_ELBOW
6	RIGHT_SHOULDER
7	RIGHT_ELBOW
8	LEFT_HIP
9	LEFT_KNEE
10	RIGHT_HIP
11	RIGHT_KNEE
12	LEFT_HAND
13	RIGHT_HAND
14	LEFT_FOOT
15	RIGHT_FOOT

Table 1:Joint number -> Joint Name

Joint positions and orientations are given in the real world coordinate system. The origin of the system is at the sensor. +X points to the right of the, +Y points up, and +Z points in the direction of increasing depth.

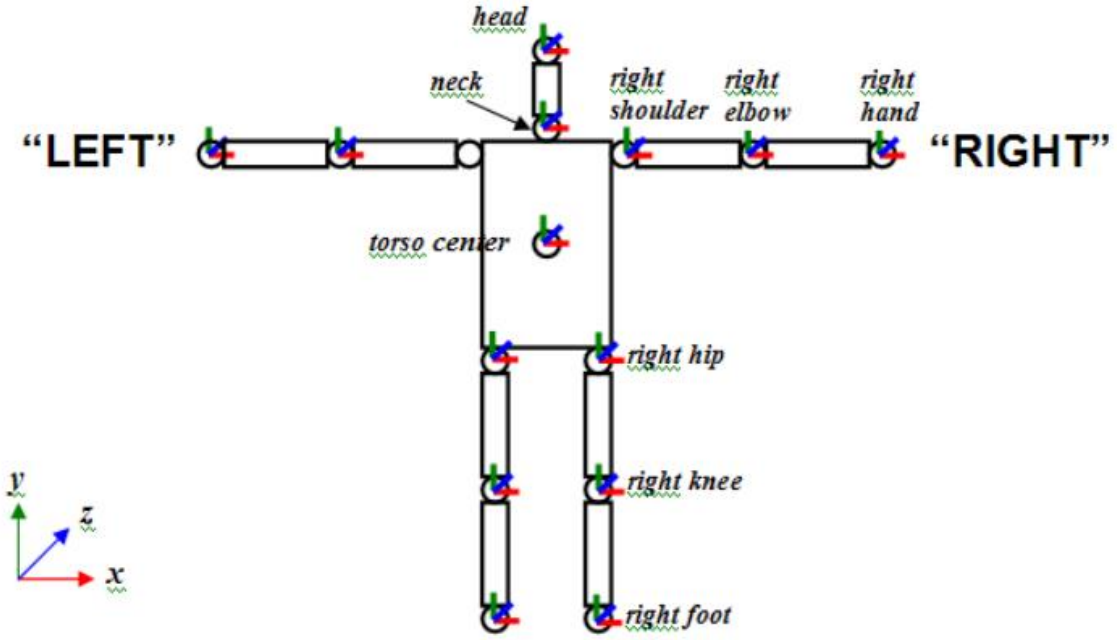

Figure 6: Joint definitions

Joint orientations are given as a 3x3 rotation orthonormal matrix. This represents a rotation between the joint's local coordinates and the world coordinates. The first column is the direction of the joint's +X axis given as a 3-vector in the world coordinate system. The second column is the +Y axis direction, and the third column is the +Z axis

direction. Our “neutral pose” is the T-pose shown in the figure above. In this pose, each joint’s orientation is aligned with the world coordinate system[53].

- #/Depth_X.png: Depth data file corresponding to frame number X of skeleton data which is a single channel 16-bit PNG
- #/RGB_X.png: RGB data file corresponding to frame number X of skeleton data which is a three channel 8-bit PNG[54].

After the selection of the dataset we import the positions of the joints into MATLAB environment. We have import, 45 joint positions, plus the first column that refer to the frame, for 14 sets of activities for 4 persons. That means we import 2520 columns for our whole dataset.

In our MATLAB dataset, each activity’s size is, a N Rows X 46 Columns dimensional vector, where N is the number of frames of each video.

Step 2. Configure dataset into 3d distances

The 3D distances of the following joint pairs were computed by using Euclidean distance. Each of these joint pairs, refers to a body pose.

Id-col	Joint pairs
1	'Torso' - 'Head'
2	'Torso' - 'ElbowLeft'
3	' Torso ' - 'HandLeft'
4	' Torso' - ‘ElbowRight’
5	' Torso ' - 'HandRight'
6	'Head' - ElbowRight’
7	'Head' - 'HandRight'
8	'Head' -'ElbowLeft'
9	'Head' - 'HandLeft'
10	'ShoulderCenter' - 'HandLeft'
11	'ShoulderCenter' - 'HandRight'

[53] (Pri)
[54] http://pr.cs.cornell.edu/humanactivities/data/README_CAD60.txt

12	'HandLeft' - 'HandRight'

Table 2:3D distances of the joint pairs

But using the estimated 3D body pose of the persons, we have to ensure that features are invariant to the size and position of the person in RGBD images. To achieve this, we had to normalize the poses based on the sum of skeleton 3D distances of Head-Torso, Left Hand to Left Elbow, Left Elbow to Left Shoulder, Right Shoulder to Left Shoulder, Right shoulder to Right Elbow and Right Elbow to Right Hand joint pairs for each subject individually. The result is a N row X 12 column dimensional vector where N is the number of frames of each activity[55].

After we normalize the data for each one of the four persons which used as subjects for the dataset, we decided to use the data of three of them for training and then the testing will take place with the data of the fourth person.

There are fourteen activities that we described through this process and they are referring to the Table 3 below.

Activity Number	Activity
1	Brushing Teeth
2	Cooking(Chopping)
3	Cooking(Stirring)
4	Drink Water
5	Open Pills Container
6	Random
7	Relax On Couch
8	Rinsing Water on Mouth
9	Still
10	Talk on Couch
11	Talk on Phone
12	Wear Contact Lenses
13	Work on PC
14	Write on Whiteboard

Table 3:Activities

The first option was to separate our training data in two ways. One way was the training by each activity separately and the other one is training of the whole dataset including all activities together.

[55] http://gesture.chalearn.org/2013-multi-modal-challenge

For the first way we have separated the dataset into activities and we unite the three person who do the same activity one after another. We have done this process for all of the activities, and for all the possible datasets (if person 1 was the testing one, the training set involves persons 2,3,4, if person 2 was for testing, the training would be done with persons 1,3,4 etc.). So dataset1 is used if person 1 is the testing one, dataset2 for person 2, dataset3 for person 3 and dataset4 for person 4.

The training data were stored in a cell array where the index of the cell was the training activity described on Table 3. For example, for the first training dataset the first cell is a vector which its size is 5038 X 12. In simple words at the persons 2,3 and 4 had sampled 5038 frames together, of their brushing teeth activity and each one of these frames is described through a set of 12 joints.

Nearly the same process has been done for the second way, but with the critical difference that we united all the videos, one after another, into one, really big, dataset. The unity wasn't random but with a logical sequence. First the three persons perform the first activity, then the second, the third etc. We had kept the initial frame numbers, knowing when each activity starts and ends.

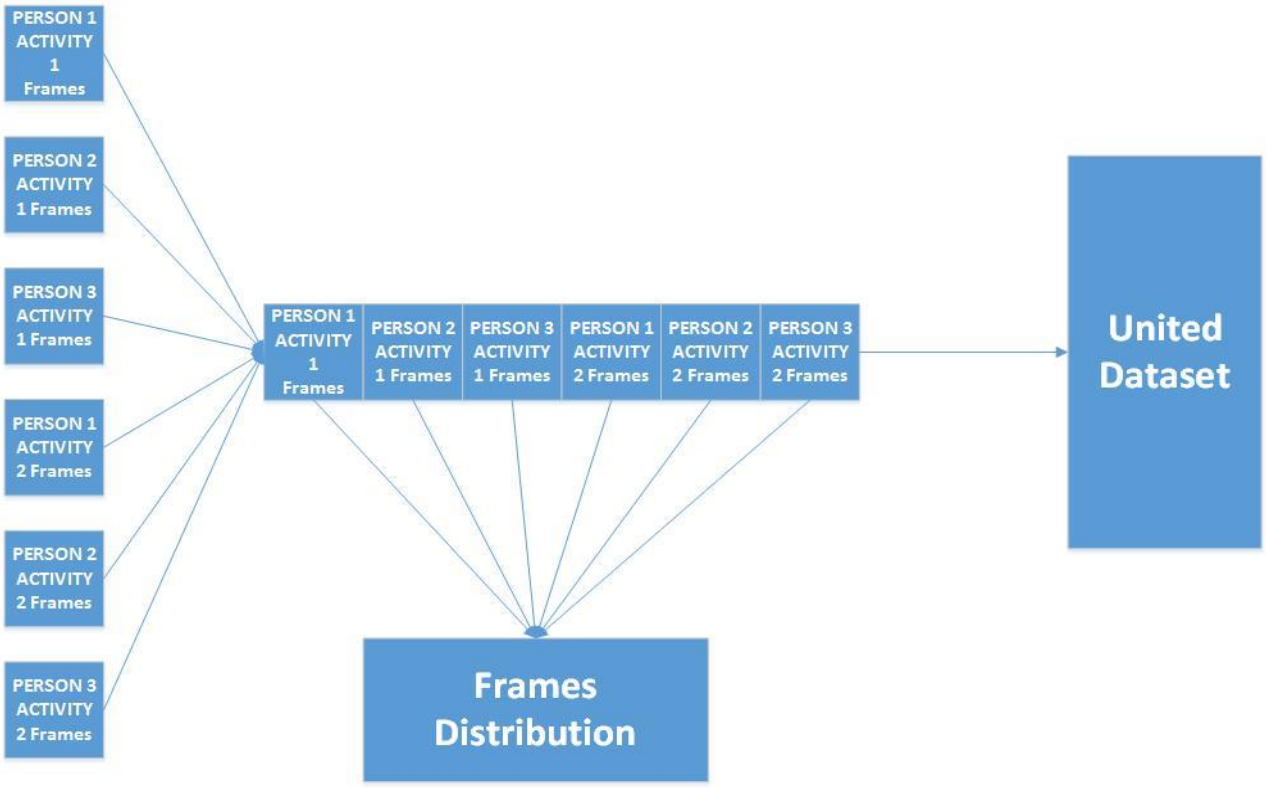

Figure 7:United Dataset

For example, at the first dataset which is an array with size 61348 X 12 (61348 frames) we know that the first activity ends at 5038th frame the second at 10366th etc. With this method we are able to detect which activity is at the specific frame and continue our process with success.

Step 3. Train dataset with K-means

The main idea of clustering the dataset with K means, is that each cluster centroid, could represent a characteristic pose of each activity. The next step of the initialization of the training with K-means was to find out how many clusters will be the optimal for our datasets.

We have used three ways to find out how many clusters. The elbow method was the first one, the silhouette method was the second, and the third was the observation of the videos in order to figure out how many poses are there in this activity.

The elbow method is implemented at MATLAB as gap criterion[56] at both, the separated activities dataset and the united dataset was very time-consuming process. Due to the large number of datasets the process take about two hours to find out that the optimal number of clusters for separated dataset was about 10 (different for each one, of course, because none of the activities had the same number of frames of each other) and 70 for the united dataset.

With the silhouette value method there wasn't so much need of time for the process itself but we had to change the k value all the time to find out a value of 20 for separated dataset and 120 for the united one.

The last method was our observation for each video in order to count how many poses the person change in every activity. For example, if the person holds still only one pose is done. But in case of wearing contact lenses there are three poses. One that person's hands are down, two where his hands are in the middle of his body, and third where his hands are up to his face and the head is stretched up. We have added 2 or three more clusters in every activity for better separation and to minimize any of our misunderstanding at the moves of the persons.

Activity	Number of clusters
Brushing Teeth	3
Cooking(Chopping)	3
Cooking(Stirring)	3
Drink Water	5
Open Pills Container	5
Random	15
Relax On Couch	4
Rinsing Water on Mouth	6
Still	3
Talk on Couch	3
Talk on Phone	3
Wear Contact Lenses	7
Work on PC	3
Write on Whiteboard	3

Table 4:Number of clusters for each activity

[56] http://www.mathworks.com/help/stats/clustering.evaluation.gapevaluation-class.html

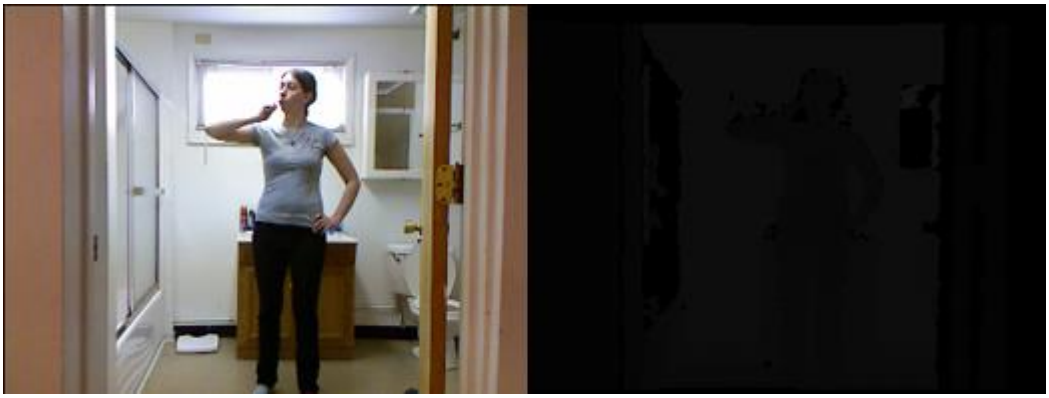

Figure 8:One characteristic pose(RGB and DEPTH) from Brushing Teeth Activity performed by Person 2

Figure 9:One characteristic pose(RGB and DEPTH) from Chopping Activity performed by Persons 3 and 4

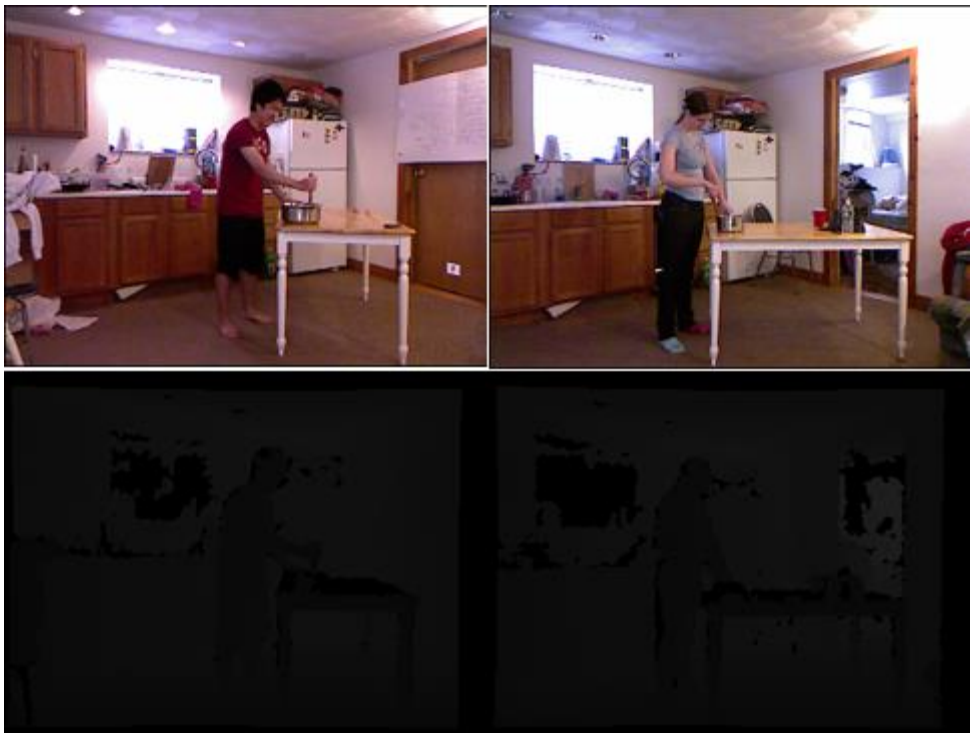

Figure 10:One characteristic pose(RGB and DEPTH) from Stirring Activity performed by Persons 1 and 2

Figure 11:Three characteristic poses(RGB and DEPTH) from Drink Water Activity performed by Person 1

Figure 12:Three characteristic poses(RGB) from Open Pills Activity performed by Person 1

Figure 13:Poses(RGB) from Random Moves Activity performed by Person 1

Figure 14:Poses(RGB) from Random Moves Activity performed by Person 2

Figure 15:Poses(RGB) from Random Moves Activity performed by Person 3

Figure 16:Poses(RGB) from Random Moves Activity performed by Person 4

Figure 17:The four persons performing Relax on Couch

Figure 18:Three characteristic poses(RGB) from Rinsing Mouth with Water Activity performed by Person 3

Figure 19:Poses(RGB and DEPTH) from Staying Still Activity performed by Person 1 and 2

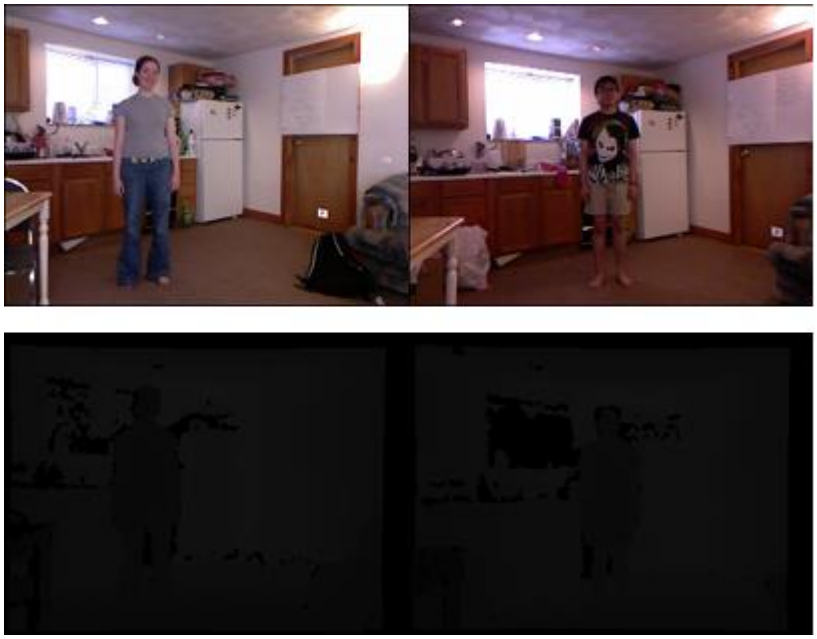

Figure 20:Poses(RGB and DEPTH) from Staying Still Activity performed by Person 3 and 4

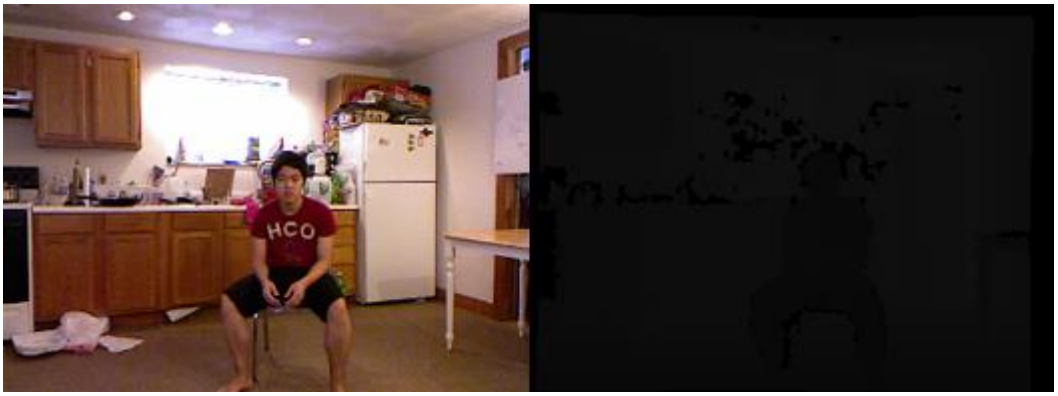

Figure 21:One characteristic pose(RGB and DEPTH) from Talk on Couch Activity performed by Person 1

Figure 22:Six characteristic poses(RGB) from Wear Contact Lenses Activity performed by Person 4

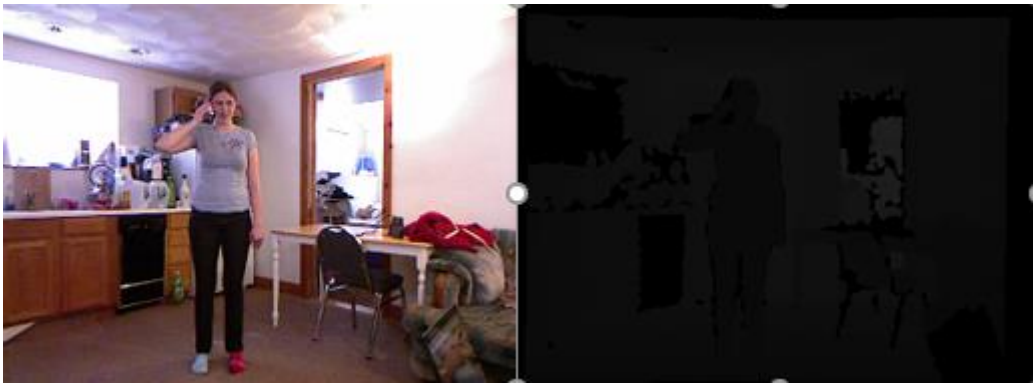

Figure 23:One characteristic pose(RGB and DEPTH) from Talk on Phone Activity performed by Person 2

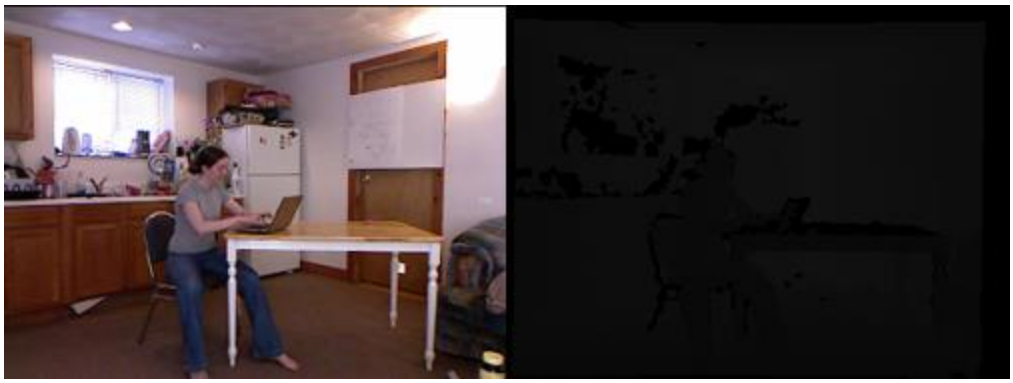

Figure 24:One characteristic pose(RGB and DEPTH) from Work on Computer Activity performed by Person 3

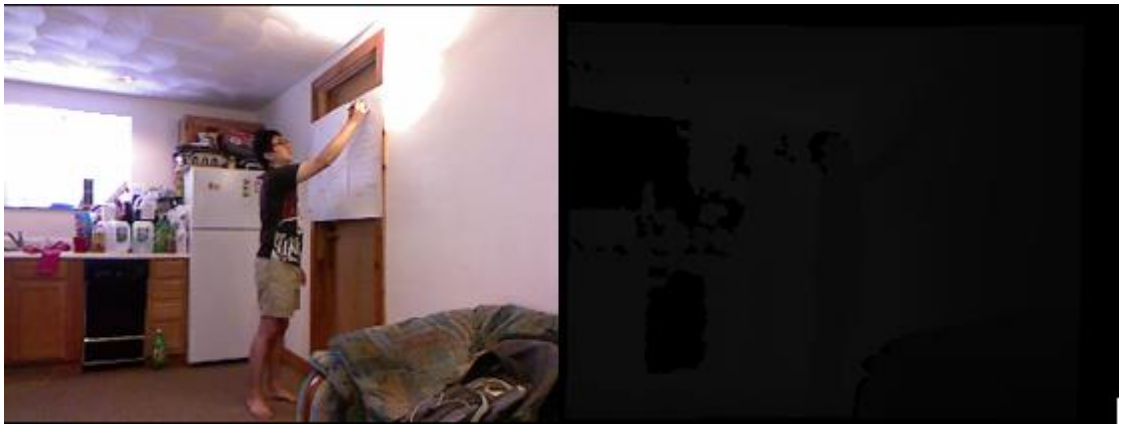

Figure 25:One characteristic pose(RGB and DEPTH) from Write on Whiteboard Activity performed by Person 4

We took results for more than these number of clusters to check our training in multiple values. In separate training we have the multiple values for each activity, 5 clusters, 10 clusters, 20 clusters and 30 clusters. In united training we have 50 clusters, 70 clusters, 120 clusters and 200 clusters.

In the case of the united dataset training we had to make sure that every activity should have at least one cluster, so we were supposed to run the k-means multiple times for making sure that all of the activities have its part to the training results.

Step 4. Evaluate with testing data

After the training of the dataset, next step is the testing and evaluation with one of the activities done by the fourth person. So a set of frames, normalized by the procedure in step 2 is selected for recognition.

The evaluation of the data can be done with two ways for both dataset training, separated of united. Both of them using the Euclidean distance between the centroids and the frames of the testing data. In mathematics, the Euclidean distance or Euclidean metric is the distance between two or more points in Euclidean space. With this distance, Euclidean space becomes a metric space.

Each of the trained activities have its own clusters which is a vector with 12 components, who represents a pose. The testing activity is also a vector with 12 components, who represents a pose. The calculation of how similar is a pose to another is the Euclidean distance between these two vectors.

Let's have a closer look at the first way of evaluating the data.

- **Evaluation with separated trained data using smallest Euclidean distance**

The process for this evaluation was to calculate the Euclidean distances between each cluster and the frames of the given activity. Then the minimum distances from each activity comparison was imported into another array and then counted how many times was each activity found based on the row index of the above array.

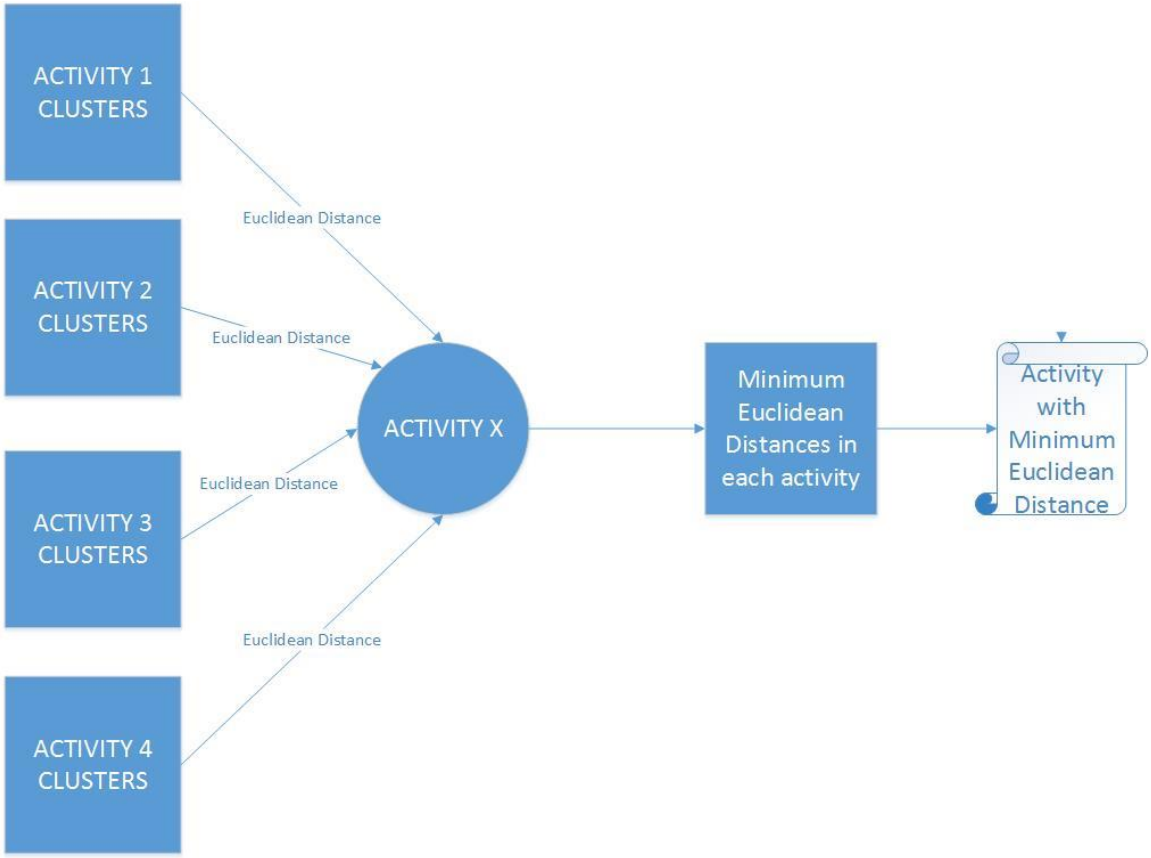

Figure 26: Evaluation with separated trained data using smallest Euclidean distance

We implemented a similar process, with the difference that we didn't just took the minimum of each activity but we took each distance was under a threshold and then we calculated the most matching ones by counting, how many similar poses was found on each activity.

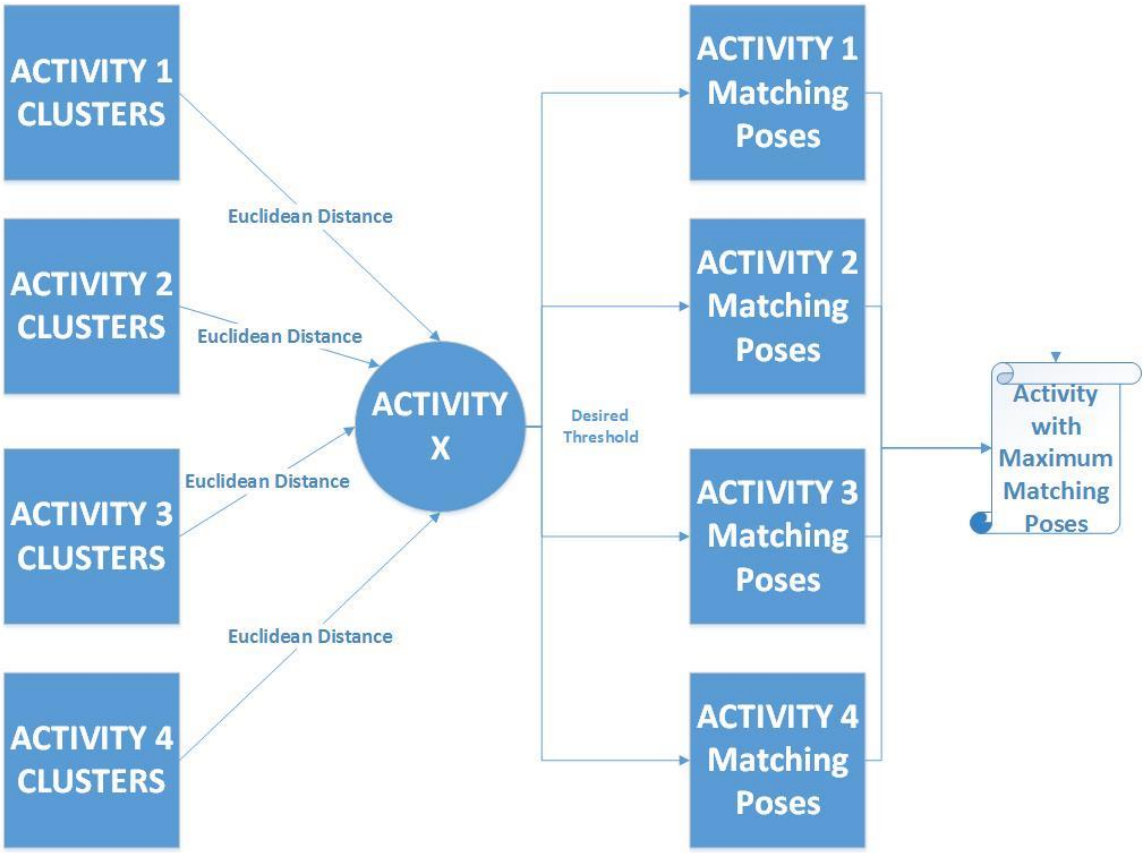

Figure 27:Evaluation with separated trained data using smallest Euclidean distance with threshold

- **Evaluation with united trained data using smallest Euclidean distance**

The evaluation of the united dataset was using the Frame Distribution array for the separation between the activities.

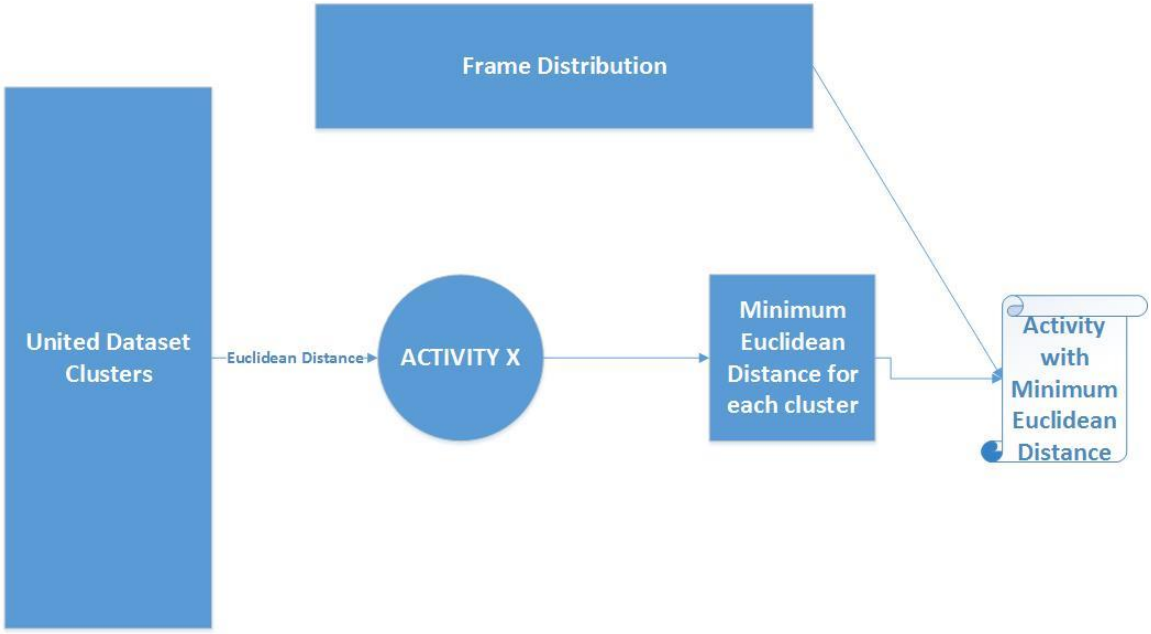

Figure 28:Evaluation with united trained data using smallest Euclidean distance

After the distance calculation and the pose matching like the above way, with (Figure 29) or without (Figure 28) distance threshold, the final results were calculated by matching the row of the distance results array combined with the data inside the Frame Distribution array.

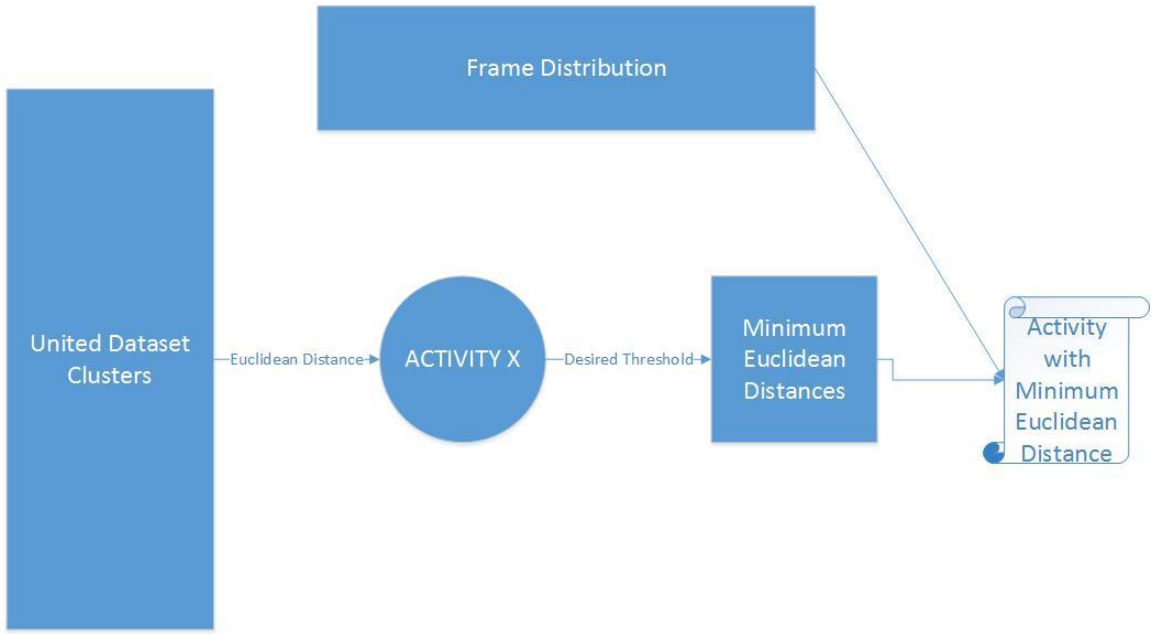

Figure 29:Evaluation with United trained data using smallest Euclidean distance with threshold

For the second way we had to find out in what sequence the matching poses appears. So if the sequence of the poses of an activity was the same or similar with the testing one that means that this activity is the right one.

For this implementation we used again the Euclidean distance to find the most matching frames both for training dataset and for testing dataset but this time we kept the sequence of the appearance of every matching. After this process we transform the numeric arrays into strings. Then we proceed to compare the two strings with Levenshtein or edit distance with use of Wagner-Fisher algorithm. This implementation was taken from Mathworks.[57]

- **Evaluation with separated trained data using matched frame sequence**

The problem here was to give labels to clusters, because if they kept separated the first of the activity number 1 had label 1, the second will use number 2 etc. But for the first cluster for activity number 2 will be labeled with number 1 as well. This is false because the poses that representing each cluster are not the same not even similar at each other. So, due to the labeling of the clusters, we had to unite the clusters of all the activities to a single array, with a procedure similar of the unity of the dataset we saw 2 steps earlier. We unite the clusters but we kept in an another array, the activity that each cluster represents (Figure 30).

[57] http://www.mathworks.com/matlabcentral/fileexchange/17585-calculation-of-distance-between-strings

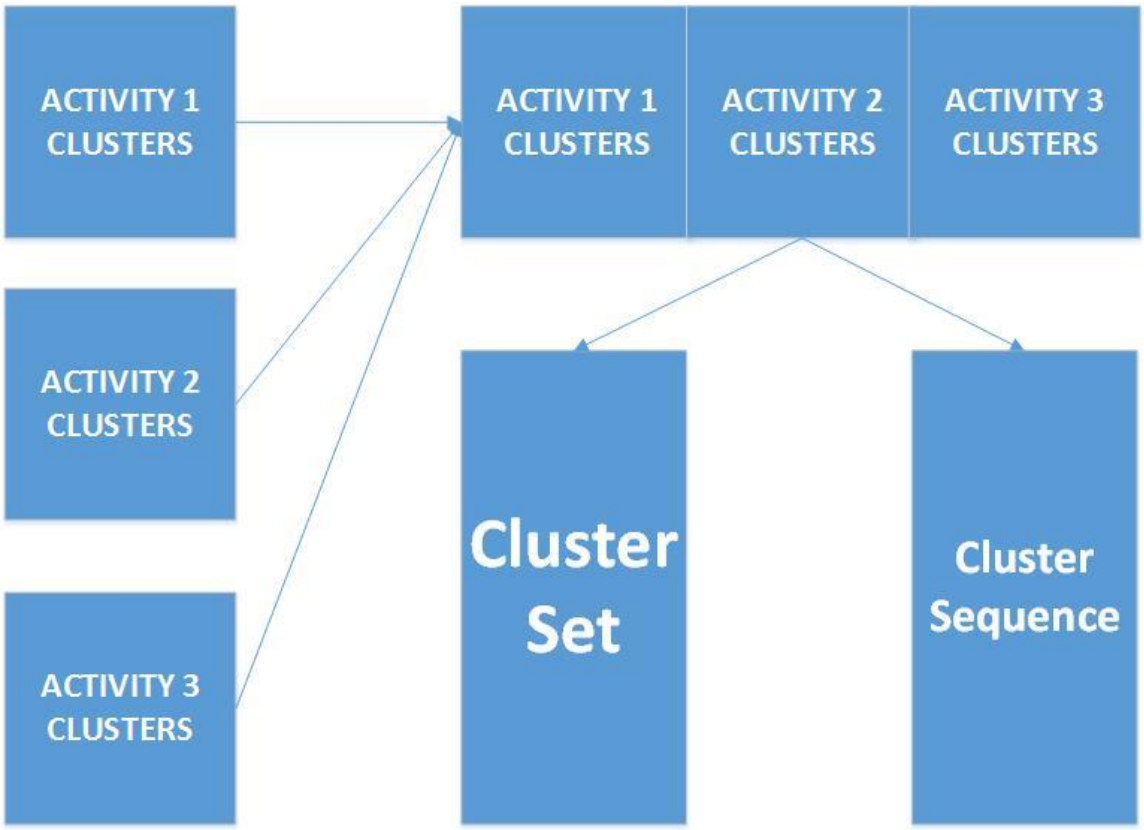

Figure 30:Unite cluster set for sequence

After this procedure we found the key frames for each activity, the key frames for the testing activity with Euclidean distance and saved on an array with the sequence that been found. After that the duplicate numbers were extinguished because we need only the sequence of the poses and not how many of the same pose are in on an activity. Then each sequence, including the testing one, were transformed at strings, and all compared with the testing with the Levenshtein and editor distances procedure, which described previously. Finally, the one with the smallest distance is the one that is recognized as the activity (Figure 31).

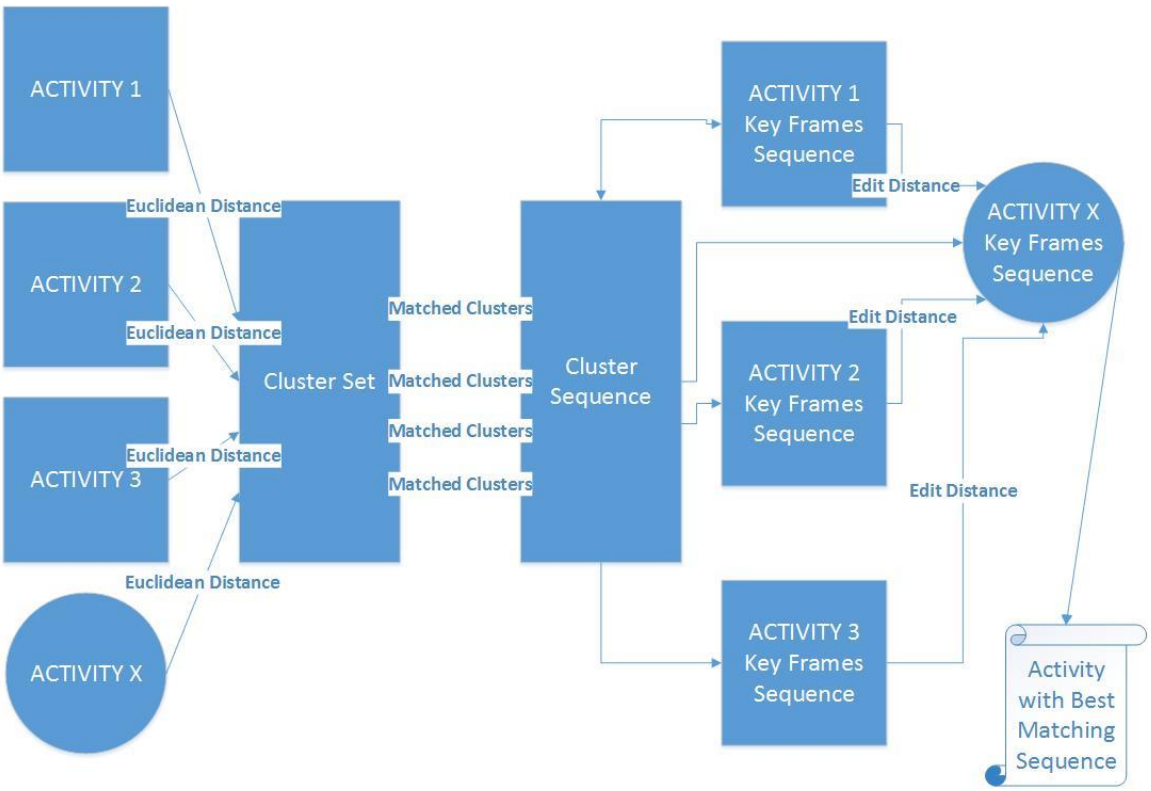

Figure 31:Evaluation with separated trained data using matched frame sequence

- **Evaluation with united trained data using matched frame sequence**

For the united dataset training did not had to do nothing more from the previous evaluation, than change the separated activities training dataset with the united one and the array with the cluster sequence with the one for Frame distribution. Then the procedure was the same as above (Figure 32).

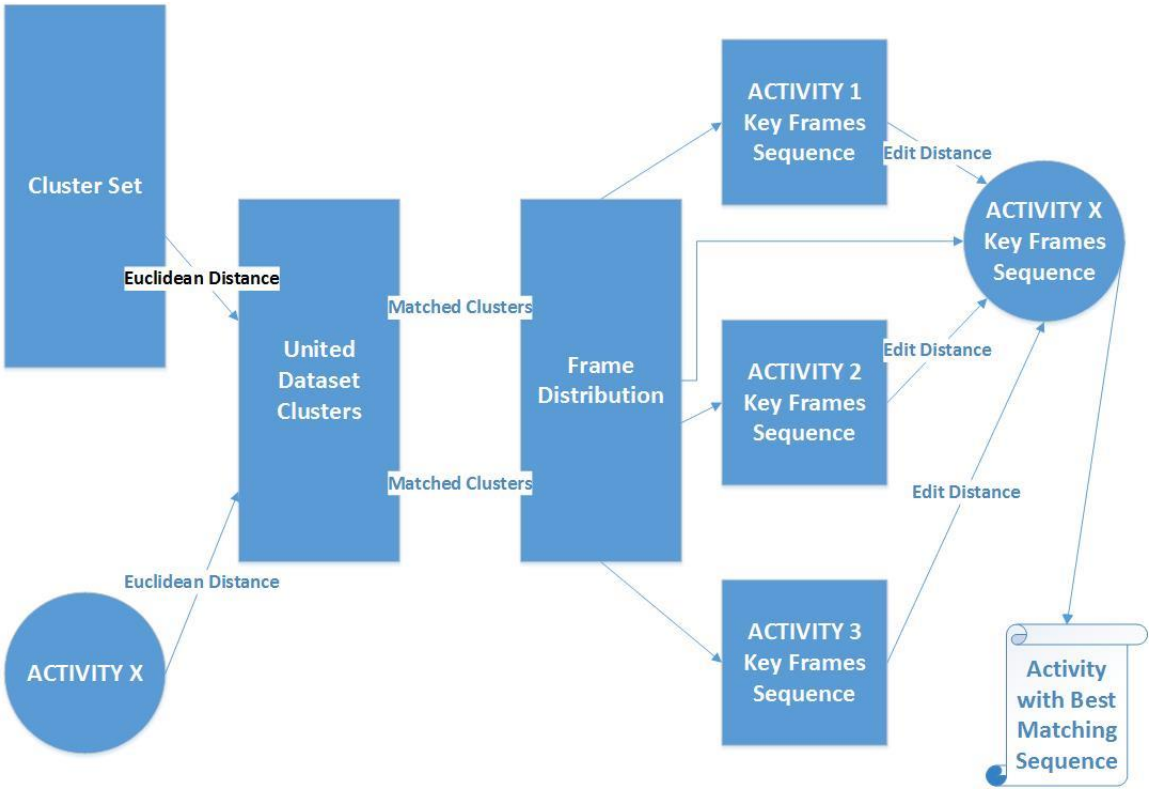

Figure 32:Evaluation with united trained data using matched frame sequence

CHAPTER 5: RESULTS

For each one of these methods we had tested for all the four persons, for all the activities and for various parameters as the number of K, in separate training, and the matching threshold. We have categorized the results by person, by activity and the overall of the performance.

- **Results for separated trained data using smallest Euclidean distance**

  The combination of the parameters for this way are written at the list below.

  1. Number of K was 5 and distance threshold 0,1
  2. Number of K was 5 and distance threshold 0,2
  3. Number of K was 5 and distance threshold 0,05
  4. Number of K was 5 and distance without threshold
  5. Number of K was 10 and distance threshold 0,1
  6. Number of K was 10 and distance threshold 0,2
  7. Number of K was 10 and distance threshold 0,05
  8. Number of K was 10 and distance without threshold
  9. Number of K was 20 and distance threshold 0,1
  10. Number of K was 20 and distance threshold 0,2
  11. Number of K was 20 and distance threshold 0,05
  12. Number of K was 20 and distance without threshold
  13. Number of K was 30 and distance threshold 0,1
  14. Number of K was 30 and distance threshold 0,2
  15. Number of K was 30 and distance threshold 0,05
  16. Number of K was 30 and distance without threshold
  17. Number of K was different for each activity and distance threshold 0,1
  18. Number of K was different for each activity and distance threshold 0,2
  19. Number of K was different for each activity and distance threshold 0,05

For the three last the number of K for each activity was 6 for rinsing mouth,3 for brushing teeth,7 for wearing contact lens,3 for talking on the phone,5 for drinking water,5 for opening pill container,3 for cooking (chopping),3 for cooking (stirring),3 for talking on couch,4 for relaxing on couch,3 for writing on whiteboard,3 for working on computer,3 for staying still, and 15 for random moves.

In general, the performance for this way for all observations was 41,26%.

The analytic results for how many times each activity recognized, each person performance and overall performance for each observation are at the tables below.

Observation No	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	Success %
Person 1	6	5	4	6	6	5	5	5	7	5	5	4	6	5	5	4	6	4	4	36,47%
Person 2	8	7	5	6	6	6	6	5	6	5	5	3	6	7	7	3	7	5	6	40,98%
Person 3	5	4	4	5	7	6	3	6	7	6	5	5	7	6	6	5	6	4	2	37,22%
Person 4	8	6	7	7	8	7	7	9	8	5	7	7	8	6	6	7	8	6	7	50,38%

Table 5: Separated trained data using smallest Euclidean distance's Performance per Person

Observation No	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	Success %
Brushing Teeth	2	2	2	2	2	2	1	2	3	3	2	1	3	3	3	2	2	1	2	52,63%
Cooking(Chopping)	4	4	4	4	4	4	3	4	4	3	4	4	4	3	4	4	4	0	3	89,47%
Cooking(Stirring)	1	0	0	1	2	0	2	1	2	0	1	1	2	0	1	1	0	0	1	21,05%
Drink Water	3	1	3	2	2	2	2	2	3	1	3	3	3	3	3	3	3	2	3	61,84%
Open Pills Container	4	4	3	2	4	4	3	2	4	2	3	1	4	4	3	2	4	4	2	77,63%
Random	1	1	1	1	0	0	2	0	0	0	0	0	1	0	1	0	1	4	1	18,42%

Relax On Couch	1	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	2	0	0	7,89%
Rinsing Water on Mouth	1	0	0	1	1	1	0	2	0	0	0	0	0	0	0	0	1	1	0	10,53%
Still	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	0	4	94,74%
Talk on Couch	0	0	0	0	0	1	0	0	0	1	0	1	0	2	0	0	0	1	0	7,89%
Talk on Phone	1	1	0	1	1	1	0	1	1	1	0	0	0	0	0	0	0	1	0	11,84%
Wear Contact Lenses	2	3	2	2	3	3	3	1	2	3	3	1	2	3	3	0	3	3	2	57,89%
Work on PC	1	0	0	2	2	0	0	3	2	1	0	1	2	0	0	1	1	0	0	21,05%
Write on Whiteboard	2	2	1	2	2	2	2	3	2	2	1	2	2	2	1	3	2	2	1	47,37%

Table 6: Separated trained data using smallest Euclidean distance's Performance per Activity

Observation No	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
Success	27	22	20	24	27	24	21	25	28	21	22	19	27	24	24	19	27	19	19
Success %	48,21 %	39,29 %	35,71 %	42,86 %	48,21 %	42,86 %	37,50 %	44,64 %	50,00 %	37,50 %	39,29 %	33,93 %	48,21 %	42,86 %	42,86 %	33,93 %	48,21 %	33,93 %	33,93 %

Table 7: Separated trained data using smallest Euclidean distance's Overall Performance

- **Results for united trained data using smallest Euclidean distance**

The combination of the parameters for this way are written at the list below.

1. Number of K was 50 and distance threshold 0,1
2. Number of K was 50 and distance threshold 0,2
3. Number of K was 50 and distance threshold 0,05
4. Number of K was 70 and distance threshold 0,1
5. Number of K was 70 and distance threshold 0,2
6. Number of K was 70 and distance threshold 0,05
7. Number of K was 120 and distance threshold 0,1
8. Number of K was 120 and distance threshold 0,2
9. Number of K was 120 and distance threshold 0,05
10. Number of K was 200 and distance threshold 0,1
11. Number of K was 200 and distance threshold 0,2
12. Number of K was 200 and distance threshold 0,05

In general, the performance for this way for all observations was 46,88%.

The analytic results for how many times each activity recognized, each person performance and overall performance for each observation are at the tables below.

Observation No	1	2	3	4	5	6	7	8	9	10	11	12	Success %
Person 1	6	6	4	8	8	5	8	8	6	8	8	6	48,21%
Person 2	8	7	4	8	8	7	6	7	6	6	6	7	47,62%
Person 3	5	5	2	6	6	3	7	6	4	8	7	5	38,10%
Person 4	8	8	7	7	9	7	7	8	5	9	10	7	54,76%

Table 8:United trained data using smallest Euclidean distance's performance per Person

Observation No	1	2	3	4	5	6	7	8	9	10	11	12	Percent
Brushing Teeth	2	2	3	2	2	2	2	2	1	2	2	2	50,00%
Cooking(Chopping)	2	2	3	4	4	3	4	4	4	4	4	4	87,50%
Cooking(Stirring)	2	1	0	2	2	1	2	3	0	2	2	0	35,42%
Drink Water	3	3	2	2	2	2	3	3	2	3	3	3	64,58%
Open Pills Container	4	4	1	4	4	2	2	2	2	4	4	4	77,08%
Random	0	0	1	1	1	2	2	2	3	3	3	4	45,83%
Relax On Couch	2	2	0	2	2	0	2	2	0	1	1	0	29,17%
Rinsing Water on Mouth	0	0	1	0	1	1	0	1	0	0	1	0	10,42%
Still	4	4	4	4	4	4	4	4	4	4	4	4	100,00%
Talk on Couch	0	0	0	0	0	0	0	0	0	0	0	0	0,00%
Talk on Phone	0	0	0	1	1	0	0	0	0	0	0	0	4,17%
Wear Contact Lenses	4	4	2	4	4	4	4	4	4	4	4	4	95,83%
Work on PC	1	1	0	1	1	0	1	0	0	2	1	0	16,67%
Write on Whiteboard	3	3	1	2	3	1	2	2	1	2	2	1	47,92%

Table 9:United trained data using smallest Euclidean distance's performance per Activity

Observation No	1	2	3	4	5	6	7	8	9	10	11	12
Success	27	26	17	29	31	22	28	29	19	31	31	25
Success %	48,21%	46,43%	30,36%	51,79%	55,36%	39,29%	50,00%	51,79%	33,93%	55,36%	55,36%	44,64%

Table 10:United trained data using smallest Euclidean distance's overall Performance

- **Results for separated trained data using matched frame sequence**

In this way we observed the video and count how many characteristic poses has every activity. The number of clusters was 4 for rinsing mouth,1 for brushing teeth,6 for wearing contact lens,1 for talking on the phone,3 for drinking water,3 for opening pill container,1 for cooking (chopping),1 for cooking (stirring),1 for talking on couch,2 for relaxing on couch,1 for writing on whiteboard,1 for working on computer,1 for staying still, and 10 for random moves.

We chose two different thresholds for creating the matching data. The first was the matching distance between the clusters and the training data (Threshold 1) and the second was the matching distance between the clusters and the testing activity (Threshold 2). Both were calculated for every activity every time. It was the slowest implementation due to the large amount of calculation. The combinations were:

1. Threshold 1 = 0.1mm and Threshold 2 = 0.05mm
2. Threshold 1 = 0.1mm and Threshold 2 = 0.1mm
3. Threshold 1 = 0.2mm and Threshold 2 = 0.2mm
4. Threshold 1 = 0.05mm and Threshold 2 = 0.05mm

Observation No	1	2	3	4	Success %
Person 1	2	2	1	1	7,14%
Person 2	2	2	2	2	10,71%
Person 3	1	3	3	4	17,86%
Person 4	0	2	2	2	10,71%

Table 11:Separated trained data using matched frame sequence's performance per person

Observation No	1	2	3	4	Success %
Brushing Teeth	0	1	0	0	6,25%
Cooking(Chopping)	2	0	0	0	12,50%
Cooking(Stirring)	1	0	0	0	6,25%
Drink Water	1	0	0	0	6,25%
Open Pills Container	0	1	1	0	12,50%
Random	0	3	4	4	68,75%
Relax On Couch	0	0	0	0	0,00%
Rinsing Water on Mouth	0	1	0	3	25,00%

Still	1	0	0	0	6,25%
Talk on Couch	0	1	1	1	18,75%
Talk on Phone	0	0	0	0	0,00%
Wear Contact Lenses	0	0	0	0	0,00%
Work on PC	0	2	2	1	31,25%
Write on Whiteboard	0	0	0	0	0,00%

Table 12:Separate trained data using matched frame sequence's performance per activity

Observation No	**1**	**2**	**3**	**4**
Success	5	9	8	9
Success %	8,93%	16,07%	14,29%	16,07%

Table 13:Separate trained data using matched frame sequence's overall performance

- **Results for united trained data using matched frame sequence**

In this way we make sure that every activity has at least one representative cluster. So the number of clusters was chosen at 35 because the calculated number of poses was by watching the video and it was slightly increased to avoid the above situation.

We chose two different thresholds for creating the matching data. The first was the matching distance between the clusters and the training data (Threshold 1) and the second was the matching distance between the clusters and the testing activity (Threshold 2). The first was calculated once automatically, as a MATLAB output, when we applied K-means on the dataset. The second is calculated every time for each activity. The combinations were:

1. Threshold 1 = 0.01mm and Threshold 2 = 0.1mm
2. Threshold 1 = 0.01mm and Threshold 2 = 0.05mm
3. Threshold 1 = 0.02mm and Threshold 2 = 0.1mm
4. Threshold 1 = 0.02mm and Threshold 2 = 0.05mm
5. Threshold 1 = 0.05mm and Threshold 2 = 0.1mm
6. Threshold 1 = 0.05 mm and Threshold 2 = 0.05mm

In general, the performance for this way for all observations was 13,10%.

The analytic results for how many times each activity recognized, each person performance and overall performance for each observation are at the tables below.

Observation No	1	2	3	4	5	6	Success %
Person 1	1	1	2	2	1	3	11,90%
Person 2	2	3	1	2	1	3	14,29%
Person 3	3	2	1	2	1	2	13,10%
Person 4	2	2	2	1	2	2	13,10%

Table 14:United trained data using matched frame sequence's performance per person

Observation No	1	2	3	4	5	6	Success %
Brushing Teeth	0	0	0	0	0	0	0,00%
Cooking(Chopping)	2	0	0	1	0	1	16,67%
Cooking(Stirring)	0	1	1	0	1	1	16,67%
Drink Water	2	2	0	0	1	1	25,00%
Open Pills Container	1	0	1	0	1	0	12,50%
Random	0	1	0	1	0	1	12,50%
Relax On Couch	0	0	0	0	0	0	0,00%
Rinsing Water on Mouth	1	0	0	0	0	0	4,17%
Still	0	4	1	4	1	4	58,33%
Talk on Couch	0	0	1	0	0	0	4,17%
Talk on Phone	0	0	0	0	0	0	0,00%
Wear Contact Lenses	1	1	1	1	0	1	20,83%
Work on PC	0	0	0	0	0	0	0,00%
Write on Whiteboard	1	0	1	0	1	1	16,67%

Table 15:United trained data using matched frame sequence's performance per activity

Observation No	1	2	3	4	5	6
Success	8	8	7	5	10	6
Success %	14,29%	14,29%	12,50%	8,93%	17,86%	10,71%

Table 16:United trained data using matched frame sequence's overall performance

About the results the best way of the four to recognize an activity was the second one meaning the training of all the activities together and then calculate the minimum Euclidean distance. The best performance was with 70 clusters and distance threshold 0,2, 200 clusters and distance thresholds 0,1 and 0,2 with 31 recognized activities out of 56 (4 persons X 14 activities). The separated dataset training had also a 50% recognized activities but the algorithm was needed a bigger amount of time than the previous way.

The most recognized activity was the staying still will almost perfect performances (97%) for the two ways with Euclidean distance. This was because all the persons, was

performing with almost the same body position, as we can see at the figures 19 and 20, so all the poses where matched correctly. With lower, but good presentences, for the same reason, was chopping (88,5%), wear contact lenses (76.8%), drink water (63,21%) and open pills container activities (77.35%).

On the other side the most common mistake was the recognition of action 4(drink water) when the requested position was number 1 (brushing teeth). The figures 8 and the middle pose of the figure 11 show us why this conflict. These are almost identical to each other that's why the application counted much of the same pose for each other. So as for person 1 when its trying to recognize his random activity moves find much poses from standing still, because his moves where stopped and holding still many times so his poses where counted to staying still activity as well. This mistake was common for other sets of activities too, like chopping when the requested one was stirring and Drinking water instead of Talk on Phone.

On the other side, the rest two ways who used sequence of the frames had a poor performance of 13,84% and 13,10%. The interesting part here is one of the activities that had the biggest presentences was the random moves with 68,75% success. In this specific move there was many moves, not controlled or programmed like the others. So the levensthein algorithm made fewer changes than the others, to identify this move.

In this ways the major problem was that if given sequence of move was the clusters 5-6-3-9-7 and the testing sequence that is calculated for one activity was 4-6-3-9-7 the distance would be equal to 1. But for the sequence 6-3-9-7-5 the distance would be equal to 5 because all the numbers should change position. So even the sequence was right even for the four out of the five numbers the system could not identify them right.

CHAPTER 6: CONCLUSION AND FUTURE WORK

In this Master thesis the objective was to implement an activity recognition using K-means. We implement four different ways with K-means used to divide in clusters the activities separately or the united dataset. Then each one of the training data was compared either by Euclidean distance of a testing activity or by matching the series of the characteristic poses. The united training with distance calculation had the best performance with 55,36% recognizing 31 of the 56 activities.

This can be improved by many ways. First we have to perform testing to multiple cluster number to find the optimal. For the same dataset, after finding out the poses, a trained neural network can be applied. Because of the high performance presentences could be increase the performance of this too and make it really faster. Due to the many compares, the debug time is very big especially at the separated training ways.

The matching sequence ways, despite of their poor performance can be improved by testing more cluster numbers and thresholds. Their good performance at random activity can be improved and the sequence of the poses can be analyzed for anticipating the next one, maybe with the help with a Hidden Markov Model or a Support Vector Machine for to characterize each activity and help the anticipation of the next move.

REFERENCES

"Testing Project Natal: We Touched the Intangible" [Journal] / auth. Wilson Mark and Buchanan Matt . - [s.l.] : Gawker Media, 2009. - June 3, 2009.

3D Convolutional Neural Networks for Human Action Recognition [Report] / auth. Ji Shuiwang [et al.]. - Tempe, USA : Arizona State University, 2015.

An approximate gazetteer for GATE based on levenshtein distance [Report] / auth. Woltzenlogel Paleo Bruno. - [s.l.] : Student Section of the European Summer School in Logic, Language and Information (ESSLLI), 2007.

Anticipating Human Activities using Object Affordances for Reactive Robotic Response [Report] / auth. Koppula Hema S. and Saxena Ashutosh. - [s.l.] : Department of Computer Science, Cornell University, 2014.

Anticipating the Future by Constructing Human Activities using Object Affordances [Report] / auth. Koppula Hema S. and Saxena Ashutosh. - [s.l.] : Department of Computer Science, Cornell University, 2013.

Automated Interpretations of Unsupervised Learning-Derived Clusters for Activity Recognition. [Report] / auth. Freedman Richard and Zilberstein Shlomo. - Kobe, Japan. : [s.n.], 2015.

Automated Worker Activity Analysis in Indoor Environments for Direct-Work Rate Improvement from Long Sequences of RGB-D Images [Report] / auth. Khosrowpour Ardalan [et al.]. - Atlanta, Georgia : Construction Research Congress, 2014.

Cluster analysis [Book] / auth. Everitt Brian. - Chichester, West Sussex, U.K : Wiley, 2011.

Clustered Support Vector Machines [Report] / auth. Gu Quanquan and Han Jiawei. - [s.l.] : Department of Computer Science University of Illinois at Urbana-Champaign, 2012.

Computer Vision [Book] / auth. Ballard Dana and Christopher Brown. - [s.l.] : Prentice Hall, 1982.

Computer Vision, A Modern Approach [Book] / auth. Forsyth David and Ponce Jean . - [s.l.] : Prentice Hall, 2003.

Concise Computer Vision [Book] / auth. Klette Reinhard. - [s.l.] : Springer, 2014.

Data mining: machine learning, statistics, and databases. [Book] / auth. Heikki Mannila. - [s.l.] : IEEE Computer Society, 1996.

E3: Microsoft shows off gesture control technology for Xbox 360 [Journal] / auth. Pham Alex. - Los Angeles : Los Angeles Times, 2009. - June 1, 2009.

Encyclopedia of Distances [Report] / auth. Deza Elena and Deza Michel Marie. - [s.l.] : Springer, 2009.

Foundations of Machine Learning [Book] / auth. Mohri Mehryar , Rostamizadeh Afshin and Talwalka Ameet. - [s.l.] : The MIT Press, 2012.

Fundamentals of Neural Networks: Architectures, Algorithms, and Applications [Book] / auth. Fausett Laurene. - [s.l.] : Prentice Hall, 1994.

Gesture Recognition using Microsoft Kinect [Report] / auth. Biswas K. K. and Basu Saurav Kumar. - Wellington, New Zealand : 5th International Conference on Automation, Robotics and Applications (ICARA), , 2011.

Histograms of Oriented Gradients for Human Detection [Report] / auth. Dalal Navneet and Triggs Bill. - San Diego, CA, USA : IEEE Computer Society Conference on Computer Vision and Pattern Recognition, , 2005.

Human Activity Detection from RGBD Images [Report] / auth. Sung Jaeyong [et al.]. - San Francisco, CA : In AAAI workshop on Pattern, Activity and Intent Recognition (PAIR), 2011.

Human Activity Detection from RGBD Images [Report] / auth. Sung Jaeyong [et al.]. - [s.l.] : AAAI workshop on Pattern, Activity and Intent Recognition (PAIR), 2011.

Image Processing, Analysis, and Machine Vision [Book] / auth. Sonka Milan, Hlavac Vaclav and Roger Boyle. - [s.l.] : Thomson, 2008.

Image Segmentation using k-means clustering, EM and Normalized Cuts [Report] / auth. Tatiraju Suman and Mehta Avi. - Irvine, CA : Department of EECS, University Of California, 2011.

Inside the race to hack the kinect. [Article] / auth. Goles J.. - [s.l.] : New Scientist, 2010. - 208.

Learning feature representations with k-means [Book Section] / auth. Coates Adam and Ng Andrew Y. // Neural Networks: Tricks of the Trade / book auth. G. Montavon G. B. Orr, K.-R. Müller. - [s.l.] : Springer, 2012.

Learning Latent Structure for Activity Recognition [Report] / auth. Hu Ninghang [et al.]. - Amsterdam, Netherlands : Intell. Syst. Lab. Amsterdam, Univ. of Amsterdam, , 2014.

Learning Spatio-Temporal Structure from RGB-D Videos for Human Activity Detection and Anticipation [Report] / auth. Koppula Hema and Saxena Ashutosh. - Ithaca, NY : Computer Science Department, Cornell University, 2013.

Machine Learning [Book] / auth. Mitchel Tom. - [s.l.] : McGraw Hill, 1997.

Neural Network Primer: Part I [Book] / auth. Caudill Maureen. - [s.l.] : AI Expert, 1989.

Neural Networks for Pattern Recognition [Book] / auth. Bishop Christopher. - [s.l.] : Oxford University Press, 1995.

Pattern Recognition [Book] / auth. Papadourakis George. - Heraklion Crete : TEI of Crete.

Pattern Recognition and Machine Learning [Book] / auth. Bishop Christopher. - [s.l.] : Springer, 2006.

Prime Sensor™ NITE 1.3 Algorithms notes [Report]. - 2010 : PrimeSense Inc..

Project Natal [Report] / auth. Microsoft. - [s.l.] : Microsoft, 2009.

Real Time Human Activity Recognition System based on Radon Transform [Report] / auth. Khan Z.A. and W. Sohn. - Kyung Hee University, South Korea : IJCA Special Issue on “Artificial Intelligence Techniques - Novel Approaches & Practical Applications”, 2011.

Recognition and anticipation of hand motions using a recurrent neural network [Report] / auth. Vamplew Peter and Adams Anthony . - University of Tasmania : Artificial Neural Networks Research Group, Department of Computer Science.

Recovering the number of clusters in data sets with noise features using feature rescaling factors [Article] / auth. Cordeiro de Amorim Renato and Hennig Christian // Information Sciences: an International Journal. - New York : Elsevier Science Inc, 2015. - Issue C. - Volume 324.

RGB-D-based Action Recognition Datasets: A Survey [Report] / auth. Zhanga Jing [et al.]. - Wollongong, Australia : School of Computing and Information Technology, 2015.

Spatio-temporal Depth Cuboid Similarity Feature for Activity Recognition Using Depth Camera [Article] / auth. Xia Lu and Aggarwal J.K. // Computer Vision and Pattern Recognition (CVPR). - Portland, OR : IEEE, 2013.

Super Normal Vector for Activity Recognition Using Depth Sequences [Report] / auth. Yang Xiaodong and Tian YingLi. - New York : Department of Electrical Engineering, 2014.

View Invariant Human Action Recognition Using Histograms of 3D Joints [Report] / auth. Xia Lu, Aggarwal J.K. and Chen Chia-Chih. - Austin, Texas : Computer & Vision Research Center / Department of ECE, 2014.

http://www.britannica.com/technology/machine-learning

http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/kmeans.html

http://people.cs.pitt.edu/~kirk/cs1501/Pruhs/Fall2006/Assignments/editdistance/Levenshtein%20Distance.htm

http://pr.cs.cornell.edu/humanactivities /data.php#data

http://pr.cs.cornell.edu/humanactivities/data/README_CAD60.txt

http://gesture.chalearn.org/2013-multi-modal-challenge

http://www.mathworks.com/help/stats/clustering.evaluation.gapevaluation-class.html

http://www.mathworks.com/matlabcentral/fileexchange/17585-calculation-of-distance-between-strings