ISLAMIC UNIVERSITY OF TECHNOLOGY

Human Action Recognition using 3D Skeletal Joint Positions

Authors
Ferdous Ahmed (104406)
Abdullah-Al-Tariq (104435)

Supervisor
Md. Hasanul Kabir, Ph.D.
Associate Professor
Department of Computer Science and Engineering

A thesis submitted to the department of CSE in partial fulfillment of the requirements for the degree of B.Sc. Engineering in CSE
Academic year: 2013-2014

October 2014

Declaration of Authorship

This is to certify that the work presented in this thesis is the outcome of the analysis and investigation carried out by Abdullah-Al-Tariq and Ferdous Ahmed under the supervision of Dr. Md. Hasanul Kabir in the Department of Computer Science and Engineering (CSE), IUT, Gazipur, Bangladesh. It is also declared that neither of the thesis nor any part of this thesis has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Authors:

_____

Ferdous Ahmed

Student ID: 104406

_____

Abdullah-Al-Tariq

Student ID: 104435

Supervisor:

_____

Md. Hasanul Kabir, Ph.D.

Associate Professor

Department of Computer Science and Engineering

Islamic University of Technology (IUT)

Abstract

Human Action Recognition is one of the intriguing research area of modern Artificial Intelligence and Computer Vision. Different researchers have proposed different methods to enable machines with the capability of recognizing human actions. One of the most traversed approaches is to use 3D depth image to acknowledge human actions. Another approach is to consider human silhouettes to predict the human actions. In this thesis we introduce a novel method to extract key frames for recognizing human actions where we use the human actions using the help of 3D skeletal joint locations. The key frames are selected depending on the distance from one frame to its neighbours and selecting a fixed number of frames out of any arbitrary number of frames. We use Microsoft Kinect to extract the joint locations where any human's twenty joint locations are provided in 3D Cartesian coordinate system. Thought there are some errors in Microsoft Kinect's joint location extraction, we consider the locations to be accurate and our research starts from that assumption. Here we introduce a new feature representation by combining histogram of joint 3D (HOJ3D) and static posture feature of 3D skeletal joint locations. By combining two representation we try to overcome their corresponding disadvantages. HOJ3D fails to represent how individual joints changes their corresponding locations with respect to other joints. Static posture feature fails to represent how these joints are distributed. Then we used Hidden Markov Model (HMM) to recognize human actions. We perform an extensive set of experiments and compare our method with some of the existing method in the field by using publicly available datasets. The evaluation method followed is n-fold validation and the results show that our method is more accurate and robust consuming less time while generating key frames. Performances generated by different number of key frames and hidden states for Hidden Markov Models are compared to show the output measure of our proposed system. The method can be used in real time to recognize human actions and can be deployed for security, augmented reality and other computer vision oriented purposes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

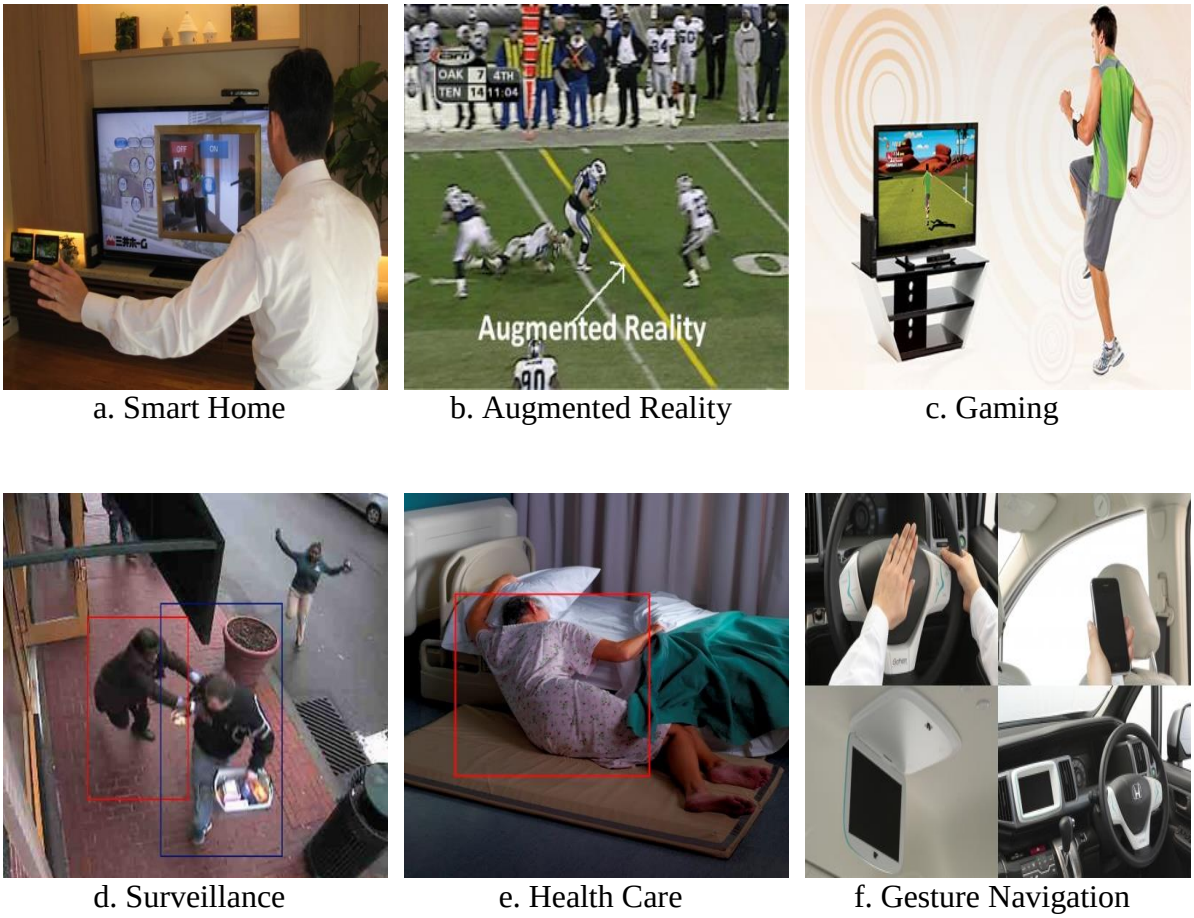
1.1 Overview

Human action recognition is a basic example of computer vision and machine learning. In today's world, automated systems with the capability to make decisions and to act accordingly of their own are highly sought-after ones for various reasons starting from office-automation to automated security systems. In order to build such systems that interact with humans and make decisions based on the activities performed by the human subjects, machines should have the ability to recognize those activities. If machines are capable of recognizing the activities of humans, they can be used for automation in different sectors. When it comes to recognize human activities, the first step should be to classify different activities of humans. Activities of humans can be divided into four classes.

- **Gestures:** Gestures deal with the motion of atomic body parts of humans. These activities run for a limited amount of time and may be used for communication between people. For example: waving a hand, shaking a leg, stretching an arm, etc.
- **Actions:** Actions are the compilation of multiple gestures of human body that requires several atomic parts. The example of action can be walking, sitting down, standing up, etc.
- **Interactions:** Interaction requires two persons and actions between them. There should be some point of contact between them. Otherwise this would be two individual actions. For example: a person stealing a suitcase from another, shaking hands, etc.
- **Group Interactions**: The interactions among a number of people is considered as a group interaction. A group meal, two groups of people fighting etc. are the examples of group interaction.

The main focus of our thesis is to identify and recognize human actions. The application of such identification and recognition can be many. The machine that can recognize human actions can be used to build a smart home where the intelligent system built in the rooms can detect the mood of the people inside and control the environment by monitoring the lighting, air flow, temperature, sound etc. Figure 1.1(a) depicts a smart home. The systems can also be applied to develop augmented reality. In augmented reality computer vision is applied to enhance the environmental features around the observer. In figure 1.1(b), we can see that augmented reality is applied on a rugby field to automatically detect the line of attack. Human action recognition system can also be applied in virtual gaming where the players can interact with the system only with the physical actions and gestures without needing any help of consoles. Figure 1.1(c) shows us a gamer playing a game only with the gestures and actions. Surveillance may be the most important use of computer vision and human action recognition system. The automated security systems should have the capability to detect intruders and suspicious behaviors. In figure 1.1(d), we can see that human action recognition is applied to detect suspicious behaviors. Another use of such systems can be in medical care where patients

need 24 hour attention of a nurse. Such effort can be minimized with the help of an automated system where a patient is always monitored to detect any unusual activities like the one in figure 1.1(e). Human action recognition system can also be applied in gesture navigation like the ones in figure 1.1(f).



a. Smart Home b. Augmented Reality c. Gaming

d. Surveillance e. Health Care f. Gesture Navigation

Figure 1: Application Domain of Human Action Recognition System

1.2 Problem Statement

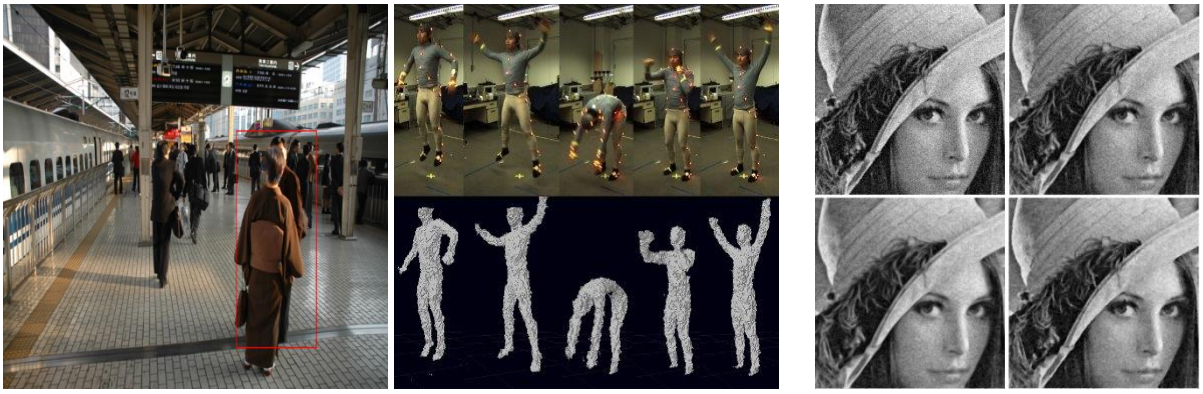
Human action recognition is a vast field. There are a lot of challenges regarding this and there are a lot of variations in human actions. Some actions are long in duration whereas some are short. Physical features and clothes are also considered while recognizing an action. But all the human action recognition systems can be divided into some basic steps. All of these parts are important to establish an efficient and effective system for human action recognition. They are Image Sequence Acquisition, Preprocessing, Feature Extraction, Feature Refinement, Action Representation and Action Recognition. Each of these steps has its own research area. For the image sequence acquisition and preprocessing steps we incorporated Microsoft Kinect. Microsoft Kinect produces 3D skeleton model of humans from single depth images without any temporal information that are used in the later steps in our thesis. The main focus of our thesis is the feature extraction step. We developed an algorithm that produces features as key

frames from a given video sequence of an action performed by human. As the feature dimension is not very huge, we skipped the feature refinement phase. The recognition of action is done using trained HMM.

1.3 Research Challenges

Recognizing actions can be very challenging due to some of the factors related to the real world scenarios. All of these challenges have not been overcome yet all at a time. Thus the main goal of our thesis was to minimize the effects of these obstacles to produce an acceptable result. The challenges faced by researchers in recognizing human actions are given below:

- **Overlapping Action:** The actions performed by humans are not necessarily segmented perfectly. There are actions that are overlapped on one another. Such cases can rise when the subject quickly perform more than one action like waving while walking or walking just after running.
- **Diversity in Action:** Human beings are not same in shape or size wise. Similarly one do not perform any action that is identical to the action performed by the person next to him. This diversity poses a great obstacle in the research of human action recognition.
- **Occlusion:** Occlusion occurs when some object blocks the view of some target object. Occlusion can happen in human action recognition due to various reasons like overlapping body parts, clothing, shadow etc. Figure 1.3 (a) is an example of occlusion.
- **Similar Actions:** Different actions are similar and some actions are even part of some other action. This results in wrong action recognition of human subjects. Figure 1.3 (b) shows some similar actions.
- **High Dimensional Representation of Temporal Sequence:** Temporal sequence of actions are represented in higher dimension. This results into false action recognition.
- **Arbitrary Viewing Angle:** There can be any angle between camera and subject. This makes the recognizing action more difficult.
- **Discriminative Features:** Extracting discriminative features from an image is very crucial. The discriminative features of a video sequence can be a lot of things. For example depth value, color, joint location histogram etc.
- **Noise:** Noise in video sequence to classify action can be very difficult. Most of the time this will result in lower than expected accuracy. Figure 1.3 (c) is an example of image with Gaussian Noise.
- **Subjective Interpretation of Action:** Same action performed by the same subject may seem different for different human observers. This is the result of subjective interpretation of actions. For example, when a subject wave his hand, it may seem as a stopping sign or a gesture of help to some culture. So the researchers may consider integrating cultural diversity in action recognition system.



a. Occlusion b. Similarity in Action c. Gaussian Noise

Figure 2: Research Challenges in Human Action Recognition

1.4 Thesis Objectives

The objective of this thesis is to use efficient algorithm for extracting key frames from a video action sequence and to classify the action offline using the Human Skeletal 3D Joint Locations provided by Microsoft Kinect. As human action recognition requires different steps mentioned in 1.2 and each of which individually is a research challenge, we used Microsoft Kinect to generate ourselves the 3D joint locations of a human subject. So our thesis is aimed at developing a method that efficiently extract features from a given video sequence. The algorithm uses data from Kinect generated joint locations and generate features that are used to determine the Key Frames of a particular action. As we mentioned above, one of the main research challenge is the dependence of action recognition on the angle of the camera. In our thesis, we proposed a method that will be view indifferent and will work offline. Many of the methods for recognizing human actions are costly and takes a lot of machine cycles or memory or both. We proposed an efficient method to generate key frames. As we are using 3D joint location of our subjects, we considered the data is devoid of occlusion. Our goal is also to reduce the effect of noise in recognizing the actions. In short, in our thesis we aimed at dealing with every research challenges there is for recognizing human actions in an efficient manner. The goals that we are trying to achieve in this thesis are summarized below:

- View Invariance
- Noise Reduction
- Efficiency

We equipped ourselves with the joint locations of different subjects performing different locations with the help of a dataset that is available in the referenced link [18].
The dataset has ten subjects with each subject performing twice where they performed ten actions. There is a list of 20 3D joint locations of a subject associated with each frame. The dataset also includes the definition of action performed within the video frame range.

1.5 Thesis Contribution

We have proposed a human action recognition method using 3D human joint locations which is more robust and efficient than others. The major contributions of this thesis are given below:

- We have proposed an efficient algorithm for generating the key frames. The algorithm is robust against view variance, scale variance and noise. We verified our algorithm with the previously mentioned dataset and the results points towards the robustness of the algorithm.
- We implemented some of the existing algorithms to investigate their efficiency and also to measure their accuracy.
- Our proposed model includes the view invariance feature that is very important in human action recognition.
- For experimental results, we implemented our proposed model and we compared with the existing methods.
- We have given an efficient approach to extract significant frames from a video sequences after extracting the 3D skeletal joint positions. Our algorithm is inspired from a paper written for video summarization.
- Our proposed algorithm proved to be computationally efficient and it reduces a significant amount of overhead both in the clustering and the classification section.
- Our algorithm is also flexible in terms of number of frames extracted.
- Our proposed action representation proved to be very accurate even when a very small number of frames were extracted.

1.6 Organization of Thesis

The rest of the thesis will be organized as follows:

- In Chapter 2, we present the literature review of the existing models as well as their performance and limitations with proper illustrations.
- In Chapter 3, we propose our human action recognition method. There we discuss about our action recognition process and step by step implementation.
- In Chapter 4, experimental set up, result of proposed model and challenges of implementation are discussed. A comparative analysis with existing method and our proposed method is also included in this chapter.
- In Chapter 5, we conclude our thesis and show the future prospects and research scopes of our proposed method.

Chapter 2

Literature Review

2.1 Related Works

Human action recognition is one of the challenging fields in computer vision research. Since the 1990s, several methods have been developed to give machines the ability to recognize different actions performed by humans. In simple cases, the machines should have the ability to analyze the ongoing video and take decisions of their own. Sometimes that can be done offline. The approaches taken by the researchers to develop methods for recognizing human activities are quite diverse. Though it was tough to recognize human activities more complex than gestures back in 1990s, now, with the advancement of technology and research work, more complex actions and gestures are being recognized by machines.

2.1.1 Taxonomy

In [1], we can see that a tree-structured, approach-based taxonomy is used to categorize all the different approaches to recognize human activities. There is extensive literature in action recognition in a number of fields, including computer vision, machine learning, pattern recognition, signal processing, etc. [2] [1]. Conventional classifiers are frequently used for recognition [3] [4].The other one is to extract features from each silhouette and model the dynamics of the action explicitly [5] [6].
The taxonomy is given if figure 2.1.

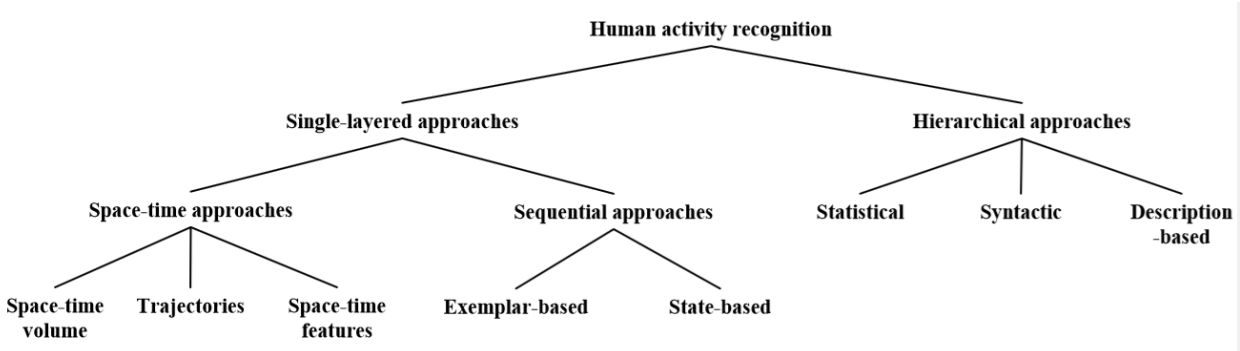


Figure 3: Hierarchical Approach-based taxonomy proposed in [2]

All activity recognition methodologies can be first classified into two broad categories. These are: (a) single-layered approaches and (b) hierarchical approaches. In Single-layered approaches human activities are represented and recognized directly based on sequences of images. Due to their nature, single-layered approaches are suitable for the recognition of gestures and actions with sequential characteristics. On the other hand, hierarchical approaches represent high-level human activities by describing them in terms of other simpler activities, which they generally call sub-events.

Single-layered approaches are again classified into two types depending on how they model human activities: space-time approaches and sequential approaches. Space-time approaches view an input video as a 3-dimensional (XYT) volume while sequential approaches interpret it as a sequence of observations. Space-time approaches are further divided into three categories based on what features they use from the 3-D space-time volumes: volumes themselves, trajectories, or local interest point descriptors. Sequential approaches are classified depending on whether they use exemplar-based recognition methodologies or model-based recognition methodologies.

Hierarchical approaches are classified based on the recognition methodologies they use: statistical approaches, syntactic approaches, and description-based approaches. Statistical approaches construct statistical state-based models concatenated hierarchically (e.g. layered hidden Markov models) to represent and recognize high-level human activities. Similarly, syntactic approaches use a grammar syntax such as stochastic context-free grammar (SCFG) to model sequential activities. Essentially, they are modeling a high-level activity as a string of atomic-level activities. Description-based approaches represent human activities by describing sub-events of the activities and their temporal, spatial, and logical structures.

2.1.2 Single Layered Approaches

There are three types of approaches classified as single layered approaches. They are:

Space Time Approach:

Space-time approach view an input video as 3-dimensional (XYT) volume. Feature is extracted as:

- Volume. [7]
- Trajectories. [15]
- Local Interest Point. [16]

The challenges of this approach are:

- Removing background clutter noise.
- 3D object matching.
- Separating overlapping objects.
- Joint location tracking.
- View invariance.

Figure 2.2 depicts space time approach.

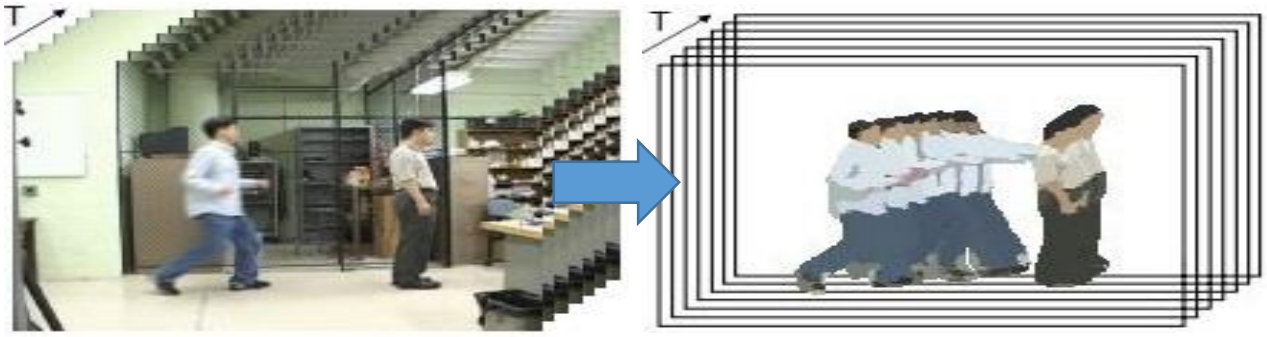


Figure 4: Space Time Approach for Human Action Recognition

Sequential Approach:
Sequential approach view an input video as a sequence of observation (i.e. feature vector). Mainly two types of sequential approach is used:

1. Exemplar Based Model. [17]
2. State Based Model. [18]

1. Exemplar Based Model:

- For a new input video sequence compare the sequence of feature vectors extracted from the video with the template sequence (or sample sequences)
- DTW matches two nonlinear sequence.

Figure 2.3 shows Dynamic Time warping example:

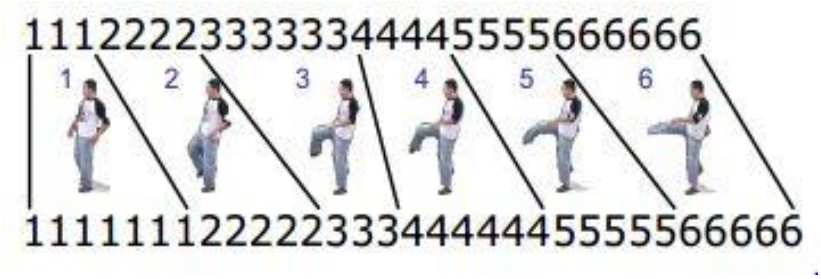


Figure 5: Dynamic Time Warping

2. State Based Model

- Considers a human activity as a model composed of a set of states.
- HMM and DBN (Dynamic Bayesian Network) is widely used in this approach.

Figure 2.4 shows Dynamic Bayesian Network.

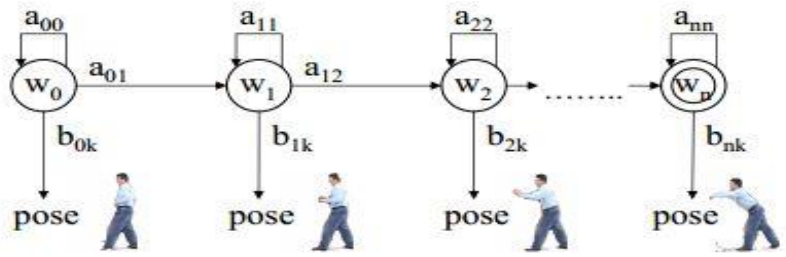


Figure 6: Dynamic Bayesian Network

In our proposed model, we will use this State Based Model with HMM as classifier. So most of our research work will be focused on this approach.

The main contribution of this work is the extraction of discriminative patterns as local features and using statistical approach in text classification to recognize actions [8].The basic workflow starts with acquisition of the image from different sensing devices. In our research work we will be acquiring image from the Microsoft Kinect sensor. After acquiring the image some pre-processing operations is applied to make the data appropriate for further processing. Then we extract the feature descriptor suitable for describing action sequence. Then some feature

reduction algorithm such as PCA, LDA, and TF-IDF is applied to reduce overhead for the subsequent operations.

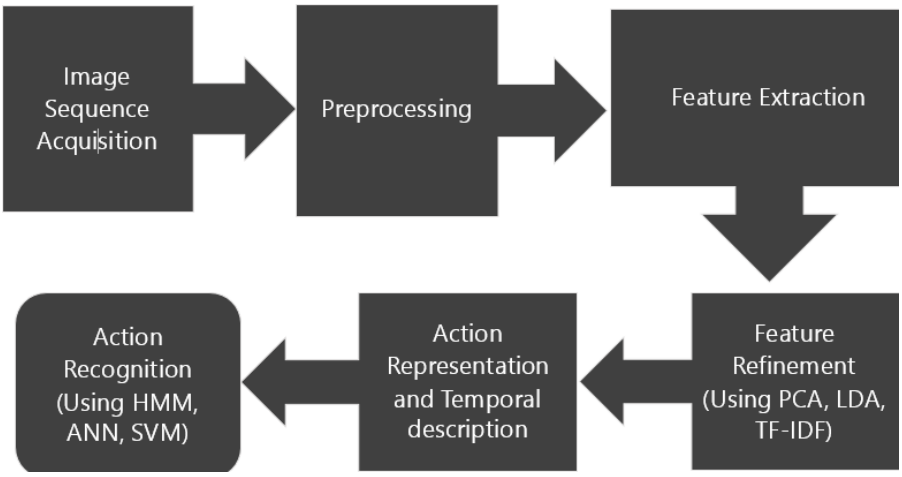


Figure 7: Basic Workflow of Sequential Approach

Generally after that further refinement operations such as clustering is applied and appropriate temporal representation is given to the actions. Lastly a classifier such as ANN, Naïve Bayes Nearest Neighbour, and Hidden Markov Model is used to recognize actions.

Chapter 3

Proposed System

This chapter describes the proposed computer vision-based human action recognition system using 3D skeletal joint position. The word “action” and “gesture” has been used alternatively.

3.1 Methodologies

Actions are meaningful movement of body parts. This work formulates a computer-vision based method that will recognize 3D full body action using 3D skeletal joint position.

A general method of recognition should be able to recognize gestures performed by any human users regardless of their gender, stature, and ethnicity, angle of view, action variations and action length.

Full body gesture includes body parts such as hand, arm, leg, head, face, even fingers. But our system do not incorporate action involving face and finger movements. But we proposed a robust system that can incorporate these minor variations very easily with simple modifications.

In our work we deal with actions that have significant motions in all three dimensions. So if we can accurately segment each body joint positions subsequent actions regarding the decisions that have to be made regarding recognition and classification of actions becomes a lot easier. So attempt has been made on extracting 3D skeletal joint positions [1] [2] using depth cameras. Depth cameras go by many names: ranging camera, flash Lidar, time-of-flight (ToF) camera, and RGB-D camera. There are many manufacturers of depth camera found in the market like Mesa Imaging SwissRanger 4000 (SR4000), PMD Technologies CamCube 2.0, Prime-sense, and Microsoft Kinect. Microsoft Kinect is probably the most popularly used depth camera available for research purposes. Kinect sensor has the ability to approximate and capture human poses, reconstruct and displayed 3D skeleton in the virtual scene using OPENNI, NITE Prime-sense and CHAI3D open source libraries. In our work we have developed a computer vision based full body action recognition system.

The key points of the method are:

- Human Body Parts Tracking.
- Mapping all joints of a human in a frame.
- Several pre-processing steps performed on the 3D skeletal joint positions before action modelling.
- Feature Description.
- Key frame extraction.
- Code book generation.
- Train and classify action sequence using Hidden Markov Model (HMM).

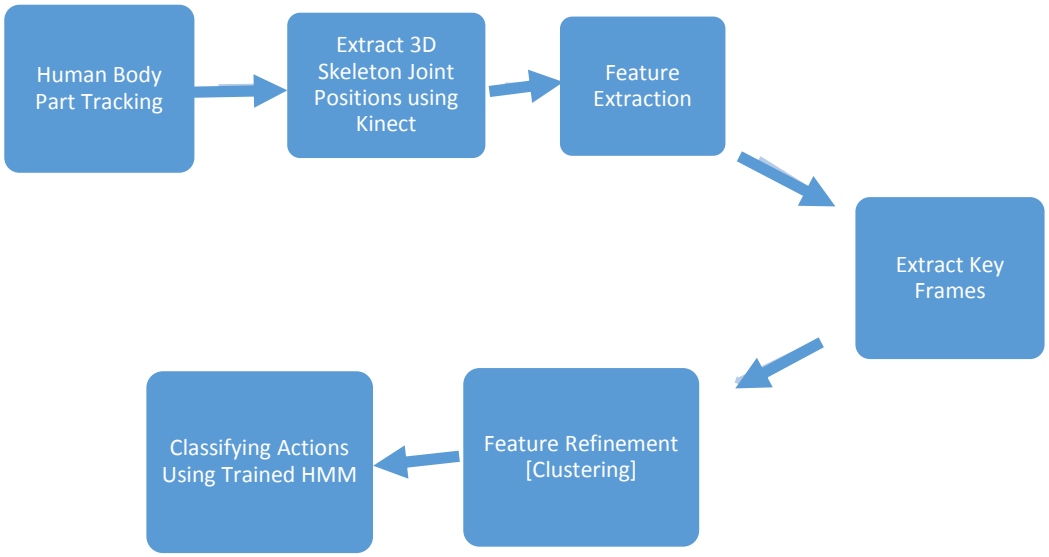


Figure 8: Overview of the Proposed System

3.2 Human Body Parts Tracking

Localization of human from the image is done with the help of Kinect sensor. Using infra-red (IR) light Kinect sensor provides the depth image of the scene in front of it. From this depth image, objects from the image can be recognized. Figure above shows the depth image of a human. Calibrating and tracking the skeleton of the human is done using OpenNI along with Kinect. Calibration of human body parts means identifying the body joints of the human. An example output of skeleton tracking is shown in figure.

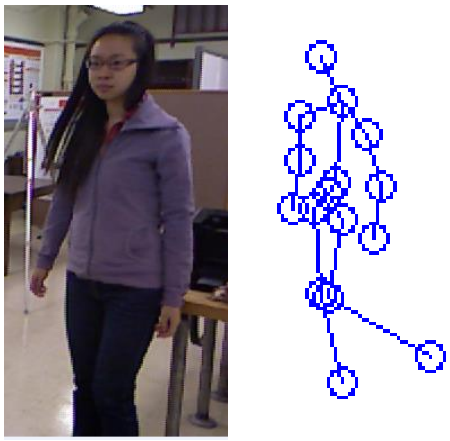


Figure 9: Stick Figure of a Human

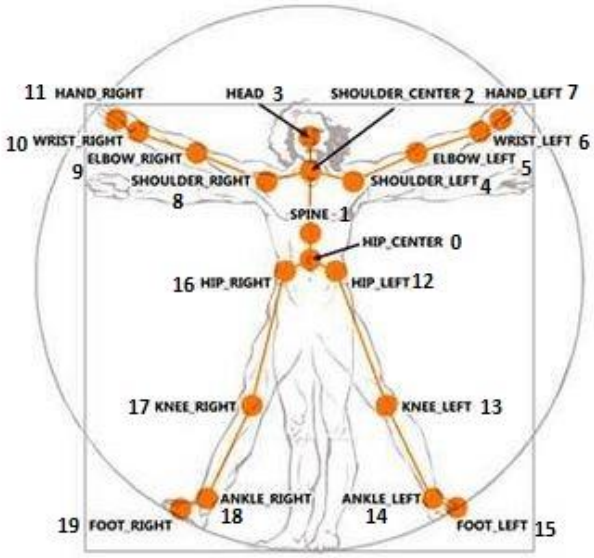


Figure 10: Labels of different Joint Locations

OpenNI enumerates 20 joints from human body. These are:

1. Hip centre: The centre location of human body. Middle point of two hips.
2. Spine: Located just above the hip centre. Positioned near the naval region.
3. Shoulder Centre: Positioned below the neck.
4. Head: Coordinate of the centre of the head. Approximately positioned near the nose.
5. Shoulder Left: It's the coordinate of the starting position of left collar bone.
6. Elbow Left: Coordinate of the left elbow.
7. Wrist left: Coordinate of the left wrist.
8. Hand Left: Coordinate of the left hand positioned on the centre of the palm.
9. Shoulder Right: It's the coordinate of the starting position of right collar bone.
10. Elbow Right: Coordinate of the right elbow.
11. Wrist Right: Coordinate of the right wrist.
12. Hand Right: Coordinate of the right hand positioned on the centre of the palm.
13. Hip Left: Positioned at the centre of the left hip.
14. Knee Left: Positioned at the centre of the left knee.
15. Ankle Left: Positioned at the centre of the left ankle.
16. Foot Left: Positioned at the centre of the left foot.
17. Hip right: Positioned at the centre of the right hip.
18. Knee right: Positioned at the centre of the right knee.
19. Ankle right: Positioned at the centre of the right ankle.
20. Foot right: Positioned at the centre of the right foot.

The locations are generated by Kinect using the approach proposed in [19]. In this paper, Shotton er al. proposed that a new method to quickly and accurately predict 3D positions of body joints from a single depth image, using no temporal information. From a single input depth image, a per pixel body part distribution is inferred. (Colors indicate the most likely part labels at each pixel, and correspond in the joint proposals). Local modes of this signal are

estimated to give high-quality proposals for the 3D locations of body joints, even for multiple users.

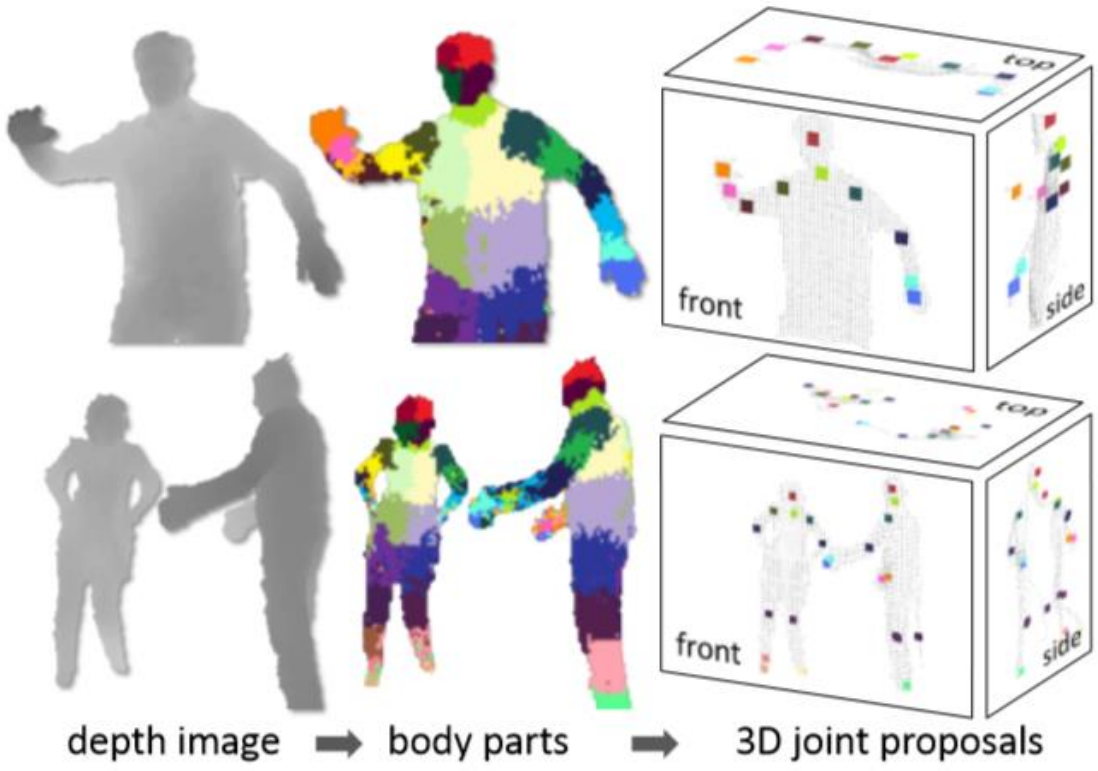


Figure 11: 3D joint location generation from Kinect

3.3 Mapping All 3D Joints of a Human in a Frame

A frame in general refers to a still image of a scene. A collection with certain speed of transition creates the illusion of movement or motion. A single frame thus describes the current properties of the scene.

In this work we have extracted 16 joints excluding hand left, hand right, foot left and foot right because these joints don't take part in differentiating between different gestures. Thus a frame is represented as a matrix of X, Y, Z co-ordinate of all joints.

3.4 Pre-processing

Before we can use these 3D skeletal joint positions to model human actions we need to perform few pre-processing operations on these values. These are:

- User to Kinect Sensor Distance Adaptation.
- Normalising each joints with respect to radial distance.
- Rotating and aligning the model along the right shoulder.
- 3D joint position normalization.

3.4.1 User to Kinect Sensor Distance Adaptation

First problem of 3D skeletal joint positions extracted is that the distance of the user with respect the camera changes alongside their movement. This value of the joints changes directly with distance of the camera. The Y values of the joints changes with the height of Kinect with respect to the user. The X value of the joints changes with the position in the frame. To resolve this problem, we take the 3D joint coordinates with respect to the centre of gravity (COG) of the user. Equation below is used for distance adaptation:

$$P = U - COG$$

Where COG = Centre of Gravity (Hip Centre)

U = Coordinate before COG correction

P = Coordinate after COG correction

This correction is done in all three axis. The correction in Z-axis ensures that the variance of distance between Kinect sensor and the user doesn't affect the performance. Correction in Y-axis nullifies the effect of the height of the Kinect sensor. Correction in X-Axis makes sure, that human can perform the gesture from any position of the frame.

3.4.2 Normalising each joints with respect to radial distance

Before starting to describe next two steps we need to convert the Cartesian co-ordinates into spherical co-ordinates using the equations below:

$r = \sqrt{x^2 + y^2 + z^2}$
$\theta = \arccos(z/\sqrt{x^2 + y^2 + z^2})$
$\varphi = \arctan(\frac{y}{x})$

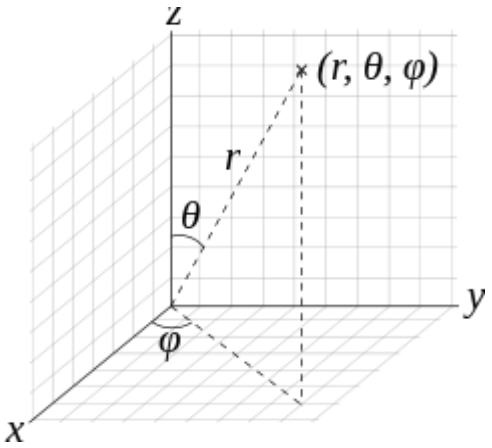


Figure 12: Spherical Co-ordinate System

To define a spherical coordinate system, one must choose two orthogonal directions, the zenith and the azimuth reference, and an origin point in space. These choices determine a reference plane that contains the origin and is perpendicular to the zenith. The spherical coordinates of a point P are then defined as follows:

- The radius or radial distance is the Euclidean distance from the origin O to P.
- The inclination (or polar angle) is the angle between the zenith direction and the line segment OP.
- The azimuth (or azimuthal angle) is the signed angle measured from the azimuth reference direction to the orthogonal projection of the line segment OP on the reference plane.

The sign of the azimuth is determined by choosing what a positive sense of turning about the zenith is. This choice is arbitrary, and is part of the coordinate system's definition.

The elevation angle is 90 degrees ($\pi/2$ radians) minus the inclination angle.

If the inclination is zero or 180 degrees (π radians), the azimuth is arbitrary. If the radius is zero, both azimuth and inclination are arbitrary.

Then we need to use the equation below to normalize:

$R_i = R_i / R_{max;}$

R_i = radial distance of i^{th} coordinate

R_{max} = maximum radial distance of the user

3.4.3 Rotating and aligning the model along the right shoulder

One problem that we will often face is the fact that camera views with respect to the human will constantly be changing as long as humans are performing their actions arbitrarily in different directions. So we need to maintain a consistent angle to counter the variations of the same actions.

So after normalizing the distance we need to align the model along the right shoulder. Figure below shows how the model is rotated along the subject's right shoulder.

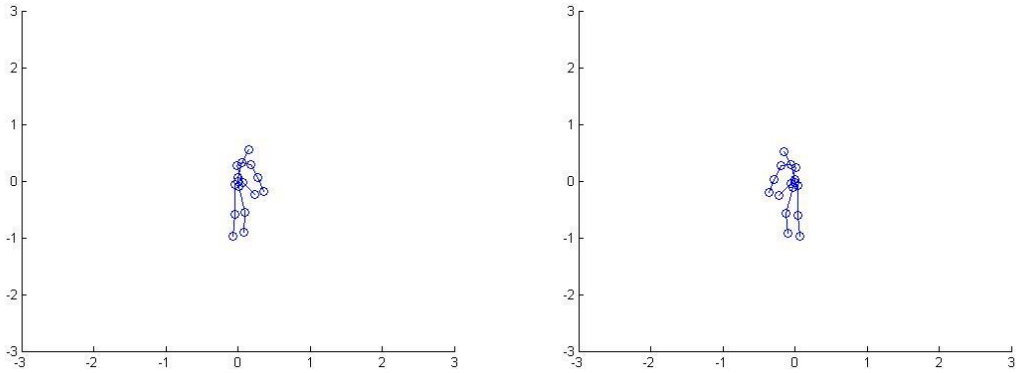


Figure 13: Rotating and Aligning a Skeleton Stick Figure with Respect to the Right Shoulder

After that we need to convert the 3D joints back to the Cartesian co-ordinate system using the formula below:

$$x = r \sin\theta \cos\varphi$$

$$x = r \sin\theta \sin\varphi$$

$$x = r \cos\theta$$

3.4.4 3D joint position normalization by down sampling

It is highly unlikely that when showing the same gesture, a person will show it exactly the same position. The coordinates of the joint positions can be approximately equal but not exactly. So if we keep exact coordinates of body joints, a specific gesture shown by a person can produce different mapped data in different times. In this work, we have solved this problem using position normalization. Normalization is done by associating a range of values to a particular value and dividing the whole range in discrete ranges. So to reduce variations in data we can use the formula below:

$$\text{new}_{\text{value}} = \left\lfloor \frac{\text{cur}_{\text{value}} - \min}{\max - \min} * \text{grid}_{\text{size}} \right\rfloor$$

$$\text{new}_{\text{value}} = \text{ new 3D joint position after normalization}$$

$$\text{cur}_{\text{value}} = \text{ old 3D joint position before normalization}$$

$$\min = \text{minimum value of the 3D joints}$$

$$\max = \text{maximum value of the 3D joints}$$

$$\text{grid}_{\text{size}} = \text{size of the total number of grids along any principal axis}$$

We can consider this approach as viewing the whole image as a collection of grids and mapping all points falling inside the grid to the centre of the grid.

3.5 Feature Description

To describe the action sequence we need to describe each and every frame. A single frame can represent a static pose. But to represent a motion gesture, we need to accumulate a sequence of frames. In this thesis, a dynamic gesture is represented by a fixed number of frame sequences.

That means, gesture is modeled using a sequence of N frames, where N is predefined. The value of N needs to be fixed and extracted as key frames from consecutive frame sequence. So

we fix the value of N. The sequence needs to he ordered, because the meaning of a gesture depends on the sequence of the motion. The order in which the frames occur is very important in recognizing gesture. Same set of frames occurring in different order means different gestures.

We employ 3D position differences of skeleton joints to characterize action information including static posture feature fcc and histogram of joint 3D skeletal co-ordinate system (HOJ3D) [8].

We then need to concatenate these two feature channel to get the final feature descriptor.

$$\text{fc} = [\text{ fcc HOj3D}]$$

Static posture feature captures the relative positions of 3D joints of a particular subject. It describes how different 3D joint positions are located with respect to each other joint position.

We can describe fcc using the formula below:

$$\text{fcc} = \{\text{Xi} - \text{Xj for all i} \neq \text{j}\}$$

$$\text{Here Xi} = \text{ith number co} - \text{ordinate of a user}$$

But the static posture feature alone cannot describe the shape and distribution of 3D joint positions. So we need to combine shape information of the 3D joints. So we need to calculate the histogram of joint 3D skeletal co-ordinate system (HOJ3D) using the approach described below:

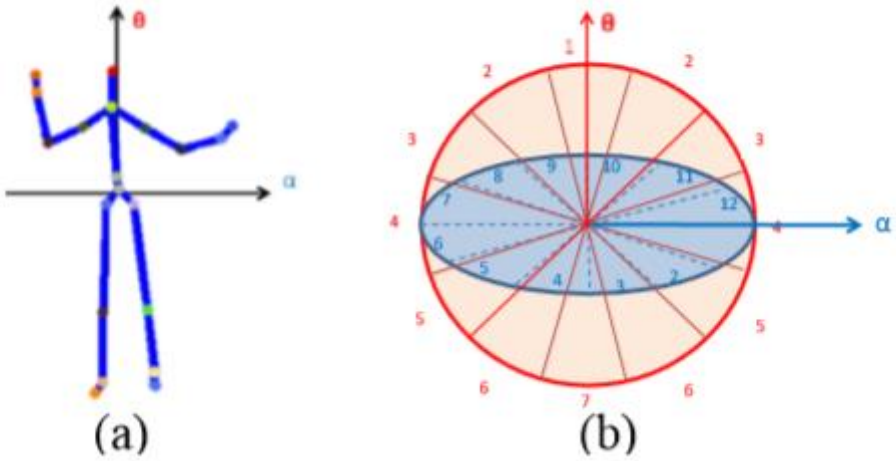


Figure 14: (a) Direction of Azimuth and Elevation in the Stick Figure. (b) Dividing the Range of Azimuth and Elevation into bins

We partition the 3D space into n bins as shown in Fig. above (in our experiment, we take n=144). The inclination angle is divided into 12 bins from the zenith vector θ: [0, 15], [15, 30], [30, 45] … [165, 180]. Similarly, from the reference vector α, the azimuth angle is divided into

12 equal bins with 30 degree resolution. The radial distance is not used in this representation to make the method scale-invariant. With our spherical coordinate, any 3D joint can be localized at a unique bin.

Our HOJ3D descriptor is computed by casting the rest 16 joints into the corresponding spatial histogram bins. For each joint location, weighted votes are contributed to the geometrically surrounding 3D bins. To make the representation robust against minor errors of joint locations, we vote the 3D bins using a Gaussian weight function:

$$p(X, \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(X-\mu)^T\Sigma^{-1}(X-\mu)}$$

, where $P(X, \mu, \sum)$ is the Gaussian probability density function with mean vector μ and covariance matrix $\sum$ (For simplicity, we use an identity matrix here). For each joint, we only vote over the bin it is in and the 8 neighbouring bins. We calculate the probabilistic voting on θ and α separately since they are independent. The probabilistic voting for each of the 9 bins is the product of the probability on α direction and θ direction. Let the joint location be ($\mu\alpha$, $\mu\beta$). The vote of a joint location to bin $[\theta 1, \theta 2]$ is:

$$p(\theta_1 < \theta < \theta_2; \mu_\theta, \sigma) = \Phi(\theta_2; \mu_\theta, \sigma) - \Phi(\theta_1; \mu_\theta, \sigma)$$

The vote of a joint location to bin $[\alpha 1, \alpha 2]$ is:

$$p(\alpha_1 < \alpha < \alpha_2; \mu_\alpha, \sigma) = \Phi(\alpha_2; \mu_\alpha, \sigma) - \Phi(\alpha_1; \mu_\alpha, \sigma)$$

Then, the probability voting to bin $\theta 1 < \theta < \theta 2$, $\alpha 1 < \alpha < \alpha 2$ is:

$$p(\theta_1 < \theta < \theta_2, \alpha_1 < \alpha < \alpha_2; \mu, \Sigma) = p(\theta_1 < \theta < \theta_2, \mu_\theta, \sigma) \cdot p(\alpha_1 < \alpha < \alpha_2, \mu_\alpha, \sigma)$$

Graphically we can visualize this way:

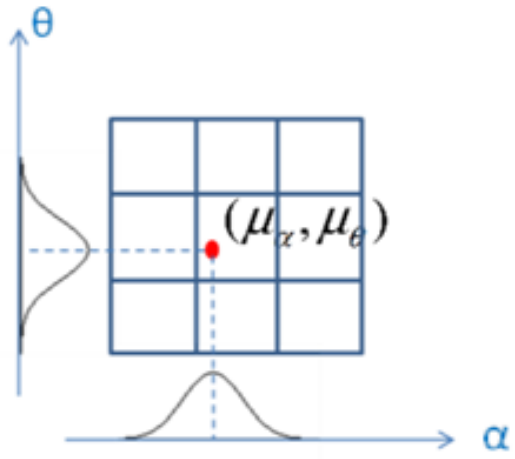


Figure 15: Probabilistic Gaussian Voting in X & Y Axis

3.6 Key Frame Extraction

There are certain reasons to extract significant frames. First we need to define what significant frame is and why do we need it. As in the affect recognition [8], temporal segments of an action can be intuitively approximated by the statuses of neutral, onset, apex, and offset. The discriminative information is not evenly distributed in the four statuses, but concentrates more on the frames from onset and apex statuses. On the other hand, motions of neutral and offset statuses are usually similar across different action categories. So informative frame selection corresponds to extract frames from onset and apex but discard frames from neutral and offset. For this purpose we have proposed a method where we extract a fixed number of frames that best summarizes the action sequence. We need to follow these steps below:

We suppose that each action consists of several fundamental unit actions. Given an action sequence, we first intend to divide it into a series of short temporal segments, each of which represent some unit action. From the segment of each unit action, we choose one frame as its representative state, which is named as a "Key-frame".

First we need to calculate self-similarity matrix S. For any two skeletons A and B, let their feature descriptor be expressed as histograms $f_c(A)$ and $f_c(B)$ where l =1, 2, 3... L. A measure of similarity between the skeletons can be computed by using the following distance:

$$d(A, B) = \sqrt{(f_c(A) - f_c(B))^2}$$

$$\text{And, } S_{ij} = d(i, j)$$

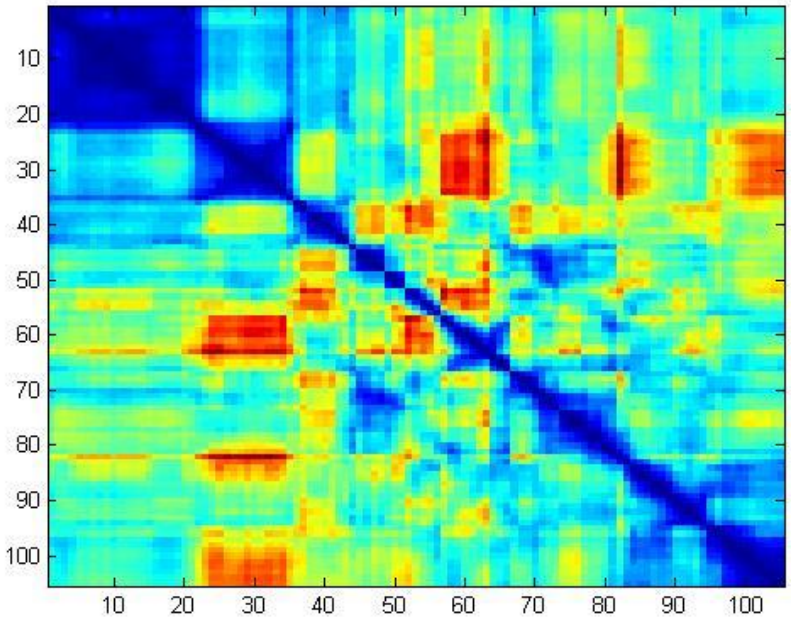


Figure 16: Self-Similarity Matrix generated from a frame sequence of length 100

Then we need to calculate cost matrix C using the formula below:

$$C_{ij} = \sum_{n=i}^{j-1} S_{n,n+1}$$

To extract a single frame where the first frame is f and last frame is l and the name of the frame to be extracted is p then,

Pick frame p if $C_{fp} \geq C_{pl}$
we repeat this process n no. of times.

Now suppose we need to extract K no. of frames. Then we run the

$$K = 2^n + 1$$

If we run this process 3 times we get 9 frames.

Figure 17: Nine Key Frames extracted from a walking video sequence

3.7 Codebook Generation

As each action is represented by an image sequence or video, the key procedure is to convert each frame into an observation symbol so that each action may be represented by an observation sequence. Note that the vector representation of postures is in a continuous space. In order to reduce the number of observation symbols, we perform vector quantization by clustering the feature vectors. We collect a large collection of indoor postures and calculate their HOJ3D vectors. We cluster the vectors into K clusters (a K-word vocabulary) using K-

means. Then each posture is represented as a single number of a visual word. In this way, each action is a time series of the visual words.

3.8 Action Recognition Using Discrete HMM

We recognize a variety of human actions by the discrete HMM technique. In discrete HMM, discrete time sequences are treated as the output of a Markov process whose states cannot be directly observed. We have encoded each action sequence as a vector of the posture words, and we input this vector to learn the HMM model and use this model to predict for the unknown sequences.

A HMM that has N states S = {s1, s2… sN} and M output symbols Y = {y1, y2… yM} is fully specified by the triplet $\lambda = \{A, B, \pi\}$. Let the state at time step t be S_t. The N×N state transition matrix A is,

$$A = \{a_{ji} \mid a_{ij} = P(S_t + 1 = q_j \mid s_t = q_i)\}$$

The N×M output probability matrix B is,

$$B = \{b_i(k) \mid b_i(k) = P(v_k \mid s_t = q_i)\}$$

And the initial state distribution vector π is

$$\pi = \{ \pi_i \mid \pi = P(S_i = q_i))$$

We use a HMM to construct a model for each of the actions we want to recognize: the HMM gives a state based representation for each action. After forming the models for each activity, we take an action sequence V= {v1, v2 … vT} and calculate its probability of a model λ_i for the observation sequence, $P(V \mid \lambda_i)$ for every model, which can be solved using the forward algorithm. Then we classify the action as the one which has the largest posterior probability.

$$\text{Decision} = \text{argmax} \{L_i\}, i = 1, 2\ldots M;$$

$$L_i = P(V \mid \lambda_i)$$

Caused by differences in the duration of performing the actions.

Experimental Result and Discussion

To test the proposed computer vision-based human action recognition system using 3D skeletal joint position we performed an extensive experiment. The experimental result and discussion focuses on the performance analysis of the system which can be divided into several categories.

4.1 Experimental Setup

The experiment was conducted in computer with Intel core i3 processor and 8 GB RAM. The operating system was "Windows 8.1 Pro 64 bit". The Kinect motion sensor is used to capture the motion data of the user. The experiment was conducted in Matlab. The major component used in the experiment are Kinect, OpenNI, and Matlab 2012.

4.1.1 Microsoft Kinect

The first generation of Microsoft Kinect shown in figure below is designed for the Xbox 360 game console. It is made to be compatible with other operating systems via the SensorKinect driver published by PrimeSense Company. The driver is also an open-source module which supports multiple platforms (Windows 7, 8, Mac OSX, and Linux). The recent release of the Microsoft Kinect addresses these issues by providing both an RGB image and depth image streams [9].

Figure 18: Image of a Microsoft Kinect Device

By working with the SensorKinect driver, the OpenNI, as discussed in the next section, is able to initialize its production nodes and to start extracting data from Kinect. Kinect is used as a gaming device but due to its capability as natural user interface, it is used for research in the field of computer vision arid human machine interaction. In our experiment, Kinect is the primary device for motion sensing.

4.1.2 OpenNI

OpenNI is an open-source framework that provides APIs for natural interaction (NI) applications in multi-language and for cross-platform utilization. Currently it supports three major platforms: Windows 7, 8, Mac OSX, and Linux. Applications built on OpenNI are usually portable and easy to be deployed to other H I iddlewarei9j.

OpenNI is a standard interface for 3D sensor data processing algorithms. It is open for all (published on the web site) and open source. The purpose is to define data types (depth map, colour map, user pose etc.) and an interface to a module that can generate them (sensor, skeleton algorithm etc.), for 3rd party users.

There are three abstract layers on the OpenNI concept. Each of the Layers represent an integral element. The OpenNI concept is shown in figure:

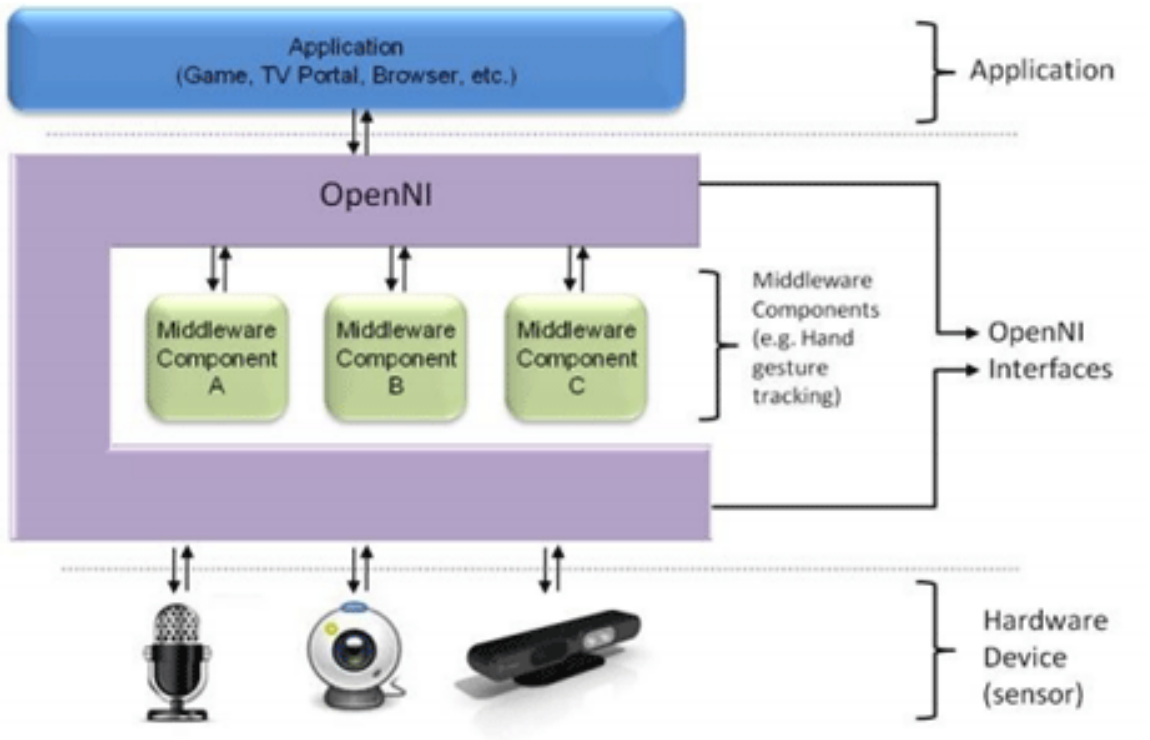


Figure 19: Layers in OpenNI Software

- The top layer represents the applications implemented for natural interactions.
- The middle layer represents the OpenNI framework, in which it not only interacts with physical sensing devices as well as software, but also communicates with middleware components.
- The bottom layer represents all kinds of sensing devices, including visual and audio sensors.

To interact with the OpenNI framework using its API, the concept of production node, capabilities and context is important to understand.

Production Node

Creating a 3D vision product is usually more complex than simply getting the output of a specific sensor. It usually starts with an actual device (the sensor) producing some sort of output (the most common case is a depth map, where each pixel has its distance from the sensor. Some sort of middleware is then used to process this output, and produce a higher-level output, like the location of a user, or its current pose.

OpenNI defines “production units" where each such unit can review data from other such units, and, optionally producing data that might be used by other units or by the application itself.

OpenNI provides two kinds of production nodes sensor-related production nodes and middleware-related production nodes. Each node is a set of components that generate a specific type of data for NI based applications. There are six sensor-related production nodes currently supported in OpenNI:

- Device
- Depth Generator
- User Generator
- Image Generator
- IR Generator
- Audio generator

This gesture recognition system makes use of the first three of them, where the Device node enables device configuration, Depth Generator generates a depth-map and User Generator helps to detect and track human.

Capabilities

OpenNI acknowledges that different providers might have different configuration option, for or their production nodes, so a set of common configurations was chosen to be mandatory from all providers. In addition some non-mandatory options were defined, and each provider can decide whether it wishes to implement them or not. These are called capabilities.

Each capability is composed of a set of functions which OpenNI exposes. A production node can be asked if it supports a specific capability. If it does, those functions can be called for that specific node.

One may think of capabilities as extensions to the common interface. Currently, all capabilities must be defined by OpenNI.

Context

The main object in OpenNI is the Context. A context is an object keeping a complete state for applications using OpenNI, including the entire production graph used by the application. The same application may create more than one context, but the contexts cannot share information between them (for example, an algorithm node cannot use a device node from another context). The context needs to be initialized once before starting to use it. In this point, all plug-ins are loaded and analysed. In order to free all memory used by the context, one should call the shutdown function.

4.2 Dataset

For experimental purpose we used a dataset that was collected as part of research work on action recognition from depth sequences. The research is described in detail in [24].

The videos was captured using a single stationary Kinect with Kinect for Windows SDK Beta Version. There are 10 action types: walk, sit down, stand up, pick up, carry, throw, push, pull, wave hands, clap hands. There are 10 subjects, each subject performs each actions twice. Three channels were recorded: RGB, depth and skeleton joint locations. The three channel are synchronized. The frame rate is 30f/s. Note we only recorded the frames when the skeleton was tracked, the frame number of the files has jumps. The final frame rate is about 15f/sec. (There is around 2% of the frames where there are multiple skeleton info recorded with slightly different joint locations. This is not caused by a second person. You can choose either one.)

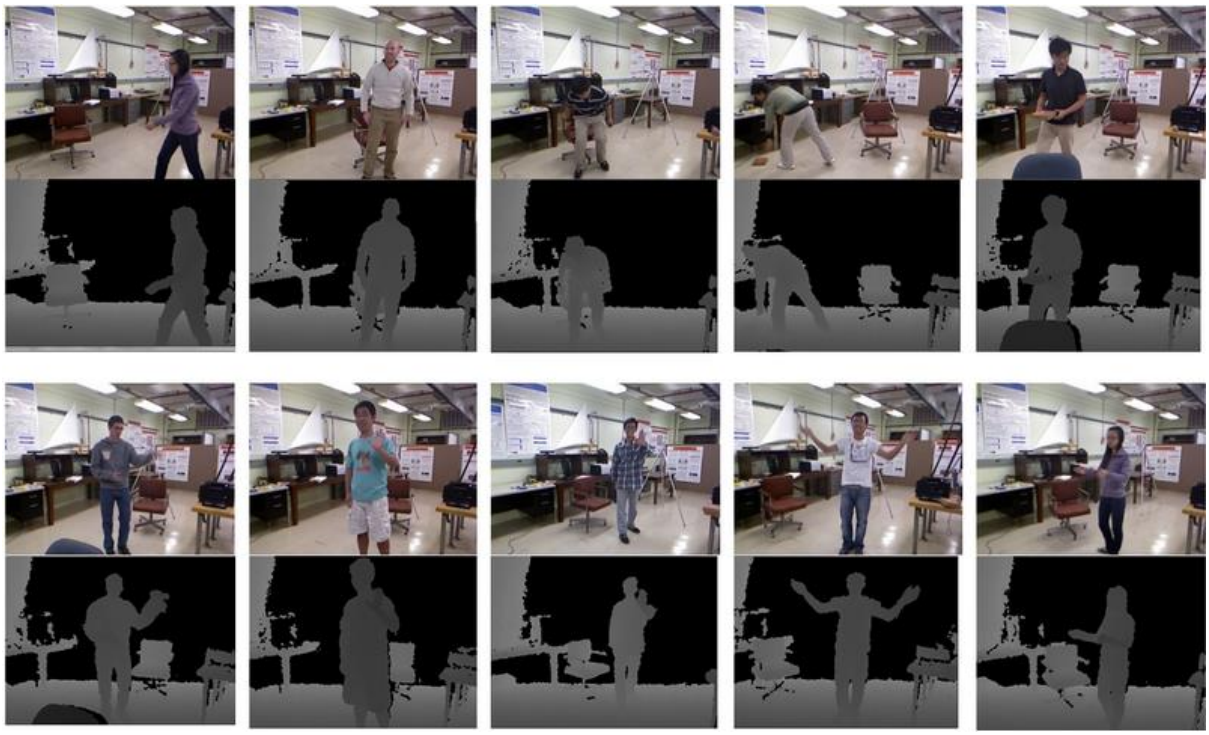


Figure 20: 2D Image & Their Corresponding Depth Image for Each Subjects

In each video, the subject performs the 10 actions in a concatenate fashion, the label of the each action segment was provided in text file. The dataset contains 4 parts:

a) RGB images (.jpg), the resolution is 480x640.
b) Depth images (.xml), the resolution is 320x240.
c) Skeletal joint Locations (.txt) each row contains the data of one frame, the first number is frame number, the following numbers are the (x, y, z) locations of joint 1-20. The x, y, and z are the coordinates relative to the sensor array, in meters.

4.3 Dataset Features

As shown in Fig. 7, we took action sequences from different views to highlight the advantages of our representation. In addition to the varied views, this dataset features 3 other challenges which are summarized as follows. First, there is significant variation among different realizations of the same action. For example, in this dataset, some actors pick up objects with one hand while others prefer to pick up the objects with both hands. Fig below is another example, individuals can toss an object with either their right or left arm or producing different trajectories. Second, the durations of the action clips vary dramatically. Figure shows the mean and standard deviation of individual action length. In this table, the standard deviation of the carry sequence lengths is 27 frames, while the mean duration of carry is 48 frames longer than that of push. Third, object-person occlusions and body part out of the field of view (FOV) also add to the difficulty of this dataset. Producing different trajectories. Second, the durations of the action clips vary dramatically. Figure below shows the mean and standard deviation of individual action length. In this figure, the standard deviation of the carry sequence lengths is 27 frames, while the mean duration of carry is 48 frames longer than that of push. Third, object-person occlusions and body part out of the field of view (FOV) also add to the difficulty of this dataset.

Table 1: Mean and Standard Deviation of Different Frame Sequence

No.	1	2	3	4	5
Mean	43.60	34.15	25.60	35.50	58.15
Standard Deviation	8.89	9.40	6.44	11.89	27.04
No.	1	2	3	4	5
Mean	11.95	10.30	15.05	45.70	31.00
Standard Deviation	4.10	4.24	7.72	16.30	20.14

4.4 Implementation

The major part of the implementation of the experiment is:

1. Training & Recognition Module.
2. Performance Analysis Module.

4.4.1 Training & Recognition module

For training purposes we used k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k − 1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged (or otherwise combined) to produce a single estimation. The advantage of this method over repeated random sub-sampling (see below) is that all observations are used for both training and validation, and each observation is used for validation exactly once. In our experiment the value used for k is 10.

4.4.2 Performance Analysis Module

This module is designed to test and verify the performance and accuracy of the action recognition system. We have used several performance measures, like:

1. Confusion Matrix.
2. Accuracy.

Confusion Matrix

Confusion Matrix is a specific table to visualize the performance of an algorithm. Each column of the matrix represents the instances in a predicted class, which each row the instances in an actual class. It is also called the error matrix.

Accuracy

We have measured the performance by calculating accuracy. Accuracy means the percentage of test action sequence correctly classified. We use the formula below to calculate the accuracy on the dataset.

$$\text{Accuracy} = \frac{\text{Number of signatures classified correctly}}{\text{Total number of test cases}}$$

Higher accuracy means a less possibility of error. As in banking sector, financial transactions are related with signature verification, that's why accuracy should be good enough.

We also varied the number of key frames, different cluster size and number of hidden states. We further compare our results against the results proposed in the paper [24].

4.5 Results and Analysis

First we extract 9 significant frames from each of the 200 video sequences. We divide the whole dataset into two parts. One for training and the other for testing. As we previously mentioned we will be using 10-fold cross subject validation we pick all video sequence of 9 subjects for training subset and 1 subject for cross validation. This approach signifies how our algorithm can be generalized for unknown samples as it is not only testing each video sequence but also it is a cross subject test. Then we take all samples of the extracted frame sequences

and use an unsupervised clustering algorithm to generate the codebook. We used a cluster size of 40 in this case. For HMM training purposes we used (Murphy's toolbox) and we used 5 hidden states for this test run. We ran this way 10 times and the confusion matrix generated in this approach is:

Table 2: Confusion Matrix of Our Proposed System

	Walk	SitDown	StandUp	PickUp	Carry	Throw	Push	Pull	WaveHand	Clap
Walk	168	0	0	2	0	0	0	0	0	0
SitDown	0	180	0	0	0	0	0	0	0	0
StandUp	0	0	180	0	0	0	0	0	0	0
PickUp	0	9	0	181	0	0	0	0	0	0
Carry	10	0	0	4	166	0	0	0	0	0
Throw	10	0	0	0	0	160	13	2	0	15
Push	0	0	0	0	0	10	185	5	0	0
Pull	0	0	0	0	0	1	3	196	0	0
WaveHand	0	0	0	0	0	0	0	0	200	0
Clap	0	0	0	0	10	0	0	0	0	190

From this confusion matrix we can get an overall accuracy. The overall accuracy here is 94.96%, standard deviation 0.2092 and maximum accuracy is 95.26%.

We can represent the confusion matrix with accuracy on the diagonal cells instead of the actual values. This is shown below:

Table 3: Confusion Matrix of Our Proposed System in Percentage

	Walk	SitDown	StandUp	PickUp	Carry	Throw	Push	Pull	WaveHand	Clap
Walk	98.82	0	0	1.18	0	0	0	0	0	0
SitDown	0	100	0	0	0	0	0	0	0	0
StandUp	0	0	100	0	0	0	0	0	0	0
PickUp	0	4.74	0	95.26	0	0	0	0	0	0
Carry	5.56	0	0	2.22	92.22	0	0	0	0	0
Throw	5	0	0	0	0	80	6.5	1	0	7.5
Push	0	0	0	0	0	5	92.5	2.5	0	0
Pull	0	0	0	0	0	0.5	1.5	98	0	0
WaveHand	0	0	0	0	0	0	0	0	100	0
Clap	0	0	0	0	5	0	0	0	0	95

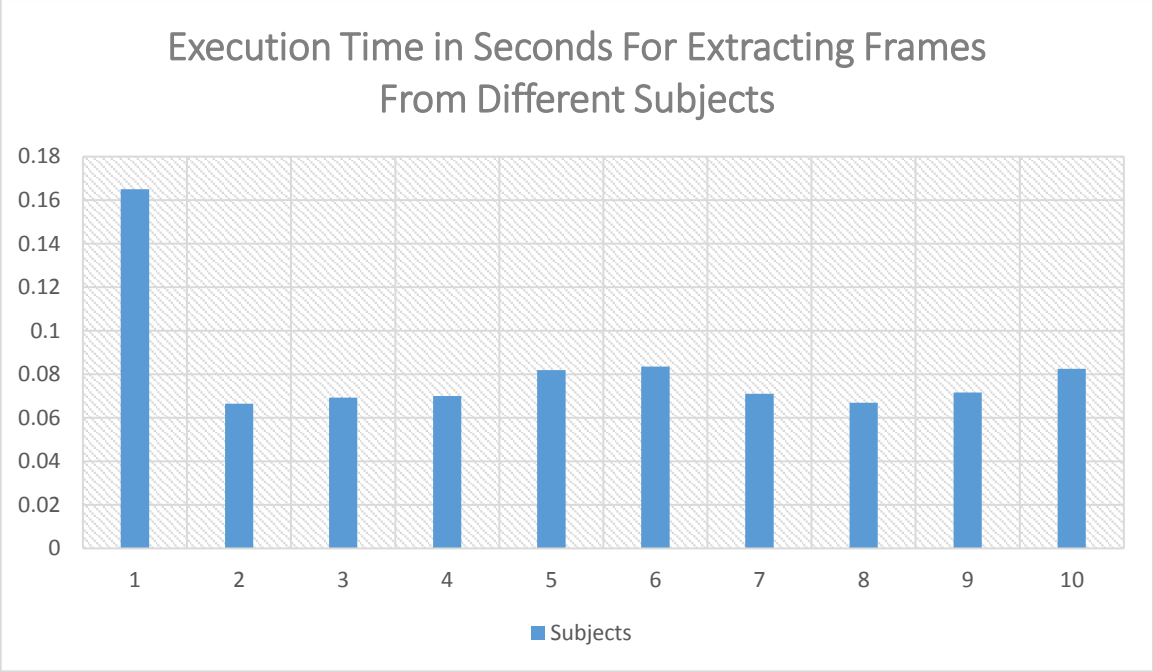


Figure 21: Execution Time in Seconds for Extracting Frames from Different Subjects

We have calculated the response time for key frame extraction from different video sequence. In our experiment we extracted 9 key frames from a video sequence of length 100 from different subjects. Results show promises for online implementation.

4.5.1 Varying the Number of Key Frames Extracted

We performed our experiment with the above mentioned setup but this time we changed the number of key frames extracted. Performance with this varying number of key frames is shown in the chart below. We found in our experiment that as we increase the number of key frames performance of our system increases because more details were available with increased number of frames. As a result less confusion occurred between actions resulting the performance increase. Optimal number of key frames proved to be 9.

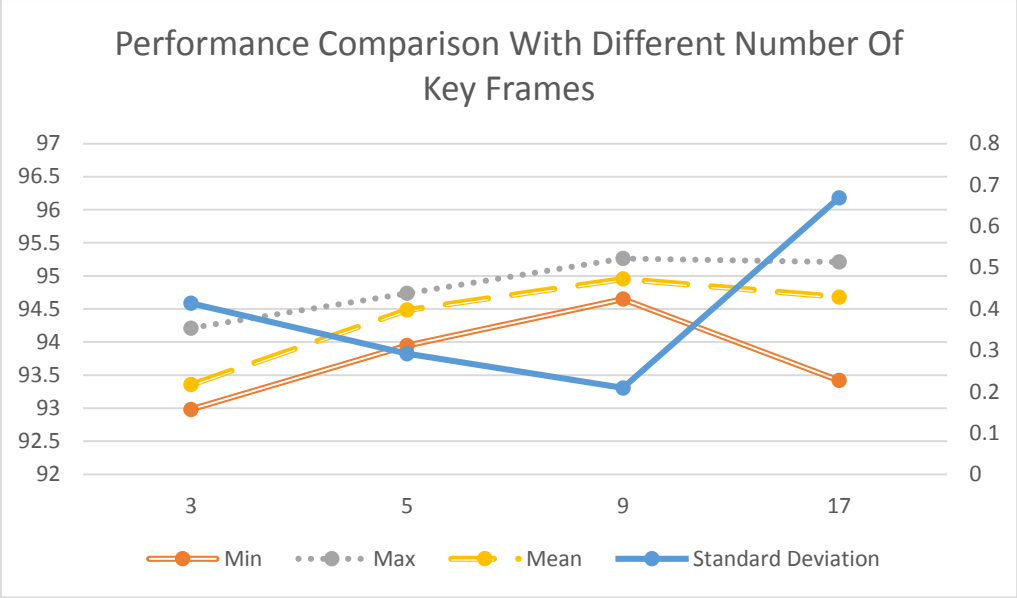


Figure 22: Performance Comparison with Different Number of Key Frames

4.5.2 Varying the Number of Hidden States

In order to measure performance while changing number of hidden states used in the HMM we extract 9 key frames which was found optimal in the earlier stage and we cluster the dataset using a size of 40. Then we perform the same experiment with different number of hidden states. The result is shown below.

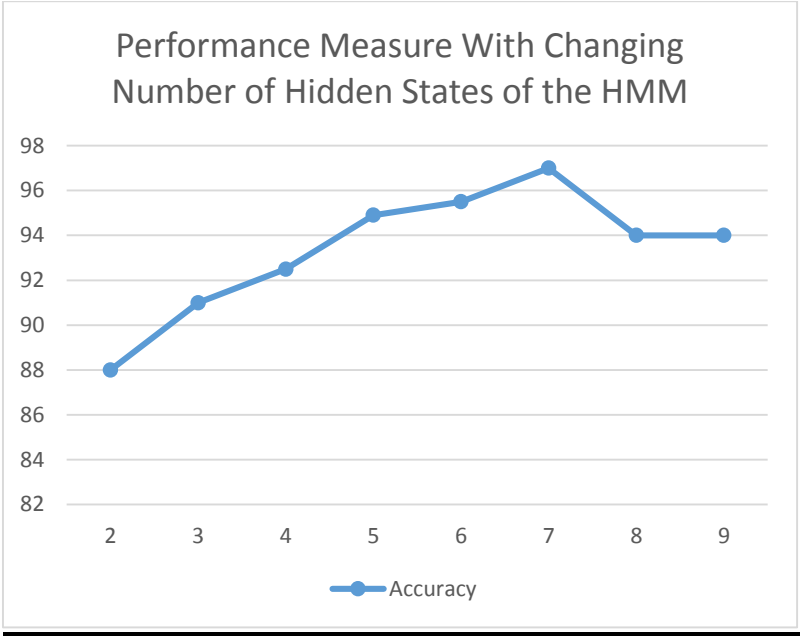


Figure 23: Performance Comparison with Changing Number of Hidden States of the HMM

In this experiment we found that the optimal number of hidden states is 7. But one observation is as we increase the number of hidden states performance increases and after a certain number of increment the performance becomes fairly consistent.

4.5.3 Varying Cluster Size

In order to measure performance while changing cluster size we extract 9 key frames which was found optimal in the earlier stage and we declare 5 hidden states. Then we perform the same experiment with different cluster size. The result is shown below.

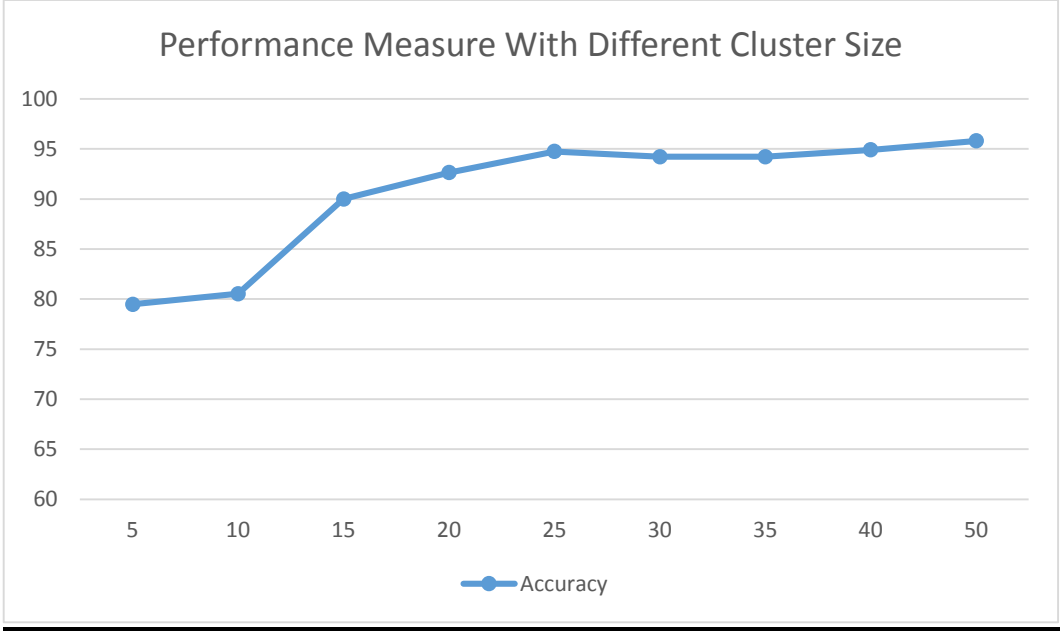


Figure 24: Performance Comparison with Different Cluster Size

From the chart we can say that as we increase the cluster size performance will gradually increase because the confusion between different action sequences is reduced when the cluster size is large.

4.6 Comparative Analysis

Now, we will compare our representation with the existing methods. Some of the existing methods varies in how the action is modelled, how key frames are extracted and how different classifier is used. As the experimental setup will vary widely in different implementations we will only focus on overall accuracy as the performance measurement. We have collected these information from different papers we came across while implementing our proposed method.

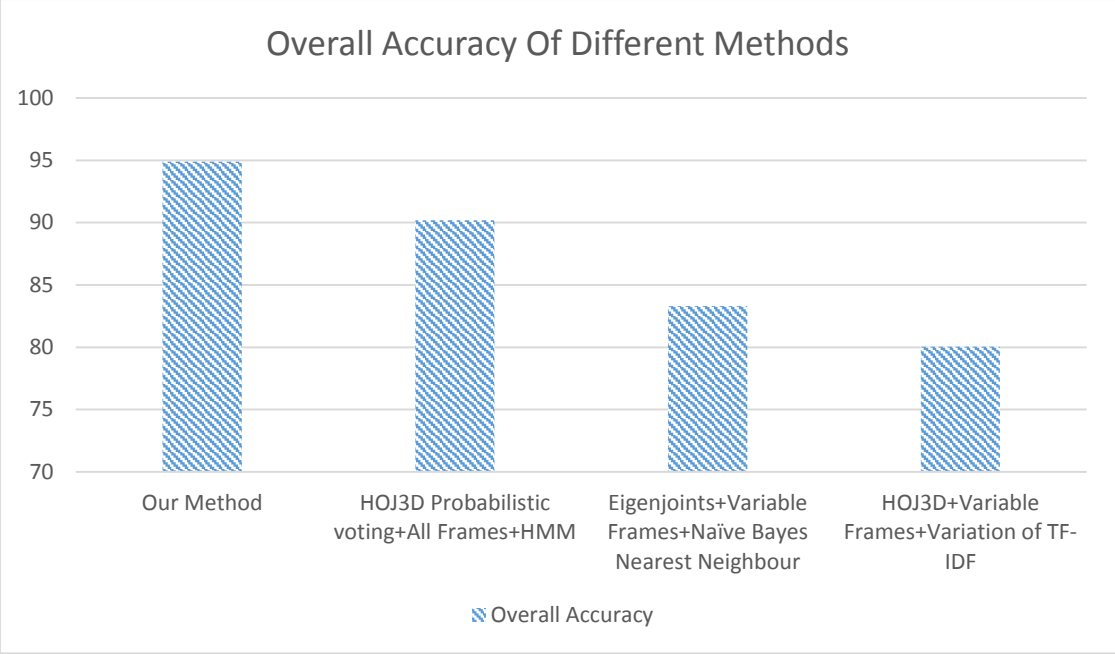


Figure 25: Performance Comparison of Different Methods

We can compare our method with the result of [24]. They used HOJ3D with probabilistic voting as feature descriptor. Then they used PCA and then LDA and they used HMM as classifier. They used a cluster size of 125 much larger than ours and used 6 states instead of 5.

Table 4: Diagonal Values of the Confusion Matrix of Paper [4]

Action	ACC	Action	ACC
Walk	95.6%	Throw	59.0%
Sit Down	91.5%	Push	81.5%
Stand up	93.5%	Pull	92.5%
Pick Up	97.5%	Wave	100%
Carry	97.5%	Clap Hands	100%

Overall: 90.92%

Best accuracy measured in their approach is 95% and standard deviation of 1.74%. Where our method gives a mean accuracy of 94.9%, best accuracy 95.26% and standard deviation of 0.2092.

We can see that our approach produces a mean accuracy much higher than theirs. Mainly because of the fact that our representation not only take accounts the shape and distribution of the joints but also their relative locations with respect to the other joints. Also we can see that accuracy of their throwing action is very poor where our representation provides much higher performance.

Conclusion

This chapter summarizes our work. Section 5.1 summarizes our contribution to the proposed system. Section 5.2 summarizes the limitation of our system. Section 5.3 focuses on future works in the domain of our proposed system.

5.1 Summary of Contributions

Human action recognition has been an active research area for a long time. But the progress in this area has been slow. Partly because of the lack of computation power and availability of depth sensing devices to detect and recognize human action efficiently. So initially action recognition focused mainly on 2D images. But lately with the availability of depth sensing devices like Microsoft Kinect action recognition has been improved. New algorithms has been developed to take advantages of these devices. In this thesis work we have also used the advantages given by these devices and proposed an efficient approach to recognize human action based on their 3D skeletal joint positions extracted using Kinect depth sensor. We have investigated many existing algorithms. The main challenges for an efficient action recognition system using 3D skeletal joint positions is to formulate a temporal representation of action sequence that is view invariant, scale invariant. Action representation should not be affected by duration of an action and how these actions are carried out by different subjects as long as they are same. Our research combines two state-of-the-art approach for an efficient action representation. In this way we combine the advantages of two methods. One approach is HOJ3D which focuses on representing action by capturing the shape of the human by taking shape histogram. But this approach neglect a key element which is the relative positions of the 3D skeletal joints. So we combine these two representation to create a robust action representation. The other contribution is we proposed an efficient algorithm to extract key frames from action sequence because there will be a lot of frames with little motions. These frames will only create overhead for the classifier. Our proposed algorithm is motivated from the paper [10] where variable number of frames are extracted. But the algorithm described in [8] solves different issues compared to our problem. So our algorithm uses their concept and extract a fixed number of key frames.

5.2 Limitations of the Proposed System

No pattern recognition system can be flawless. Our proposed system has also some limitations. The biggest limitation of our system is the data acquisition part. As we are extracting 3D skeletal joint position using Microsoft Kinect our proposed system can only be as good as the system for extracting joint positions. All limitations that comes with the extraction process is also applicable here. But there are other limitations too. We can only extract $2^k + 1$ number of frames from an action sequence where k is the number of frames to be extracted. Also key frames might be extracted that has no motion. This might happen if there is no motion in the action sequence or there might not be enough data available. Also there is a problem when no

frames exist between the first and last frame. In that case our algorithm will generate variable number of key frames.

5.3 Future Works

Our future goal includes making an online action recognition system. We also need to address some of the limitations of our proposed system.

Bibliography

[1] J. K. Aggarwal and M. S. Ryoo, “Human activity analysis: A review,” ACM Computing Surveys (CSUR), vol. 43, no. 3, pp. Article 16, 43 Pages, 2011.

[2] P. Turaga, R. Chellapa, V. S. Subrahmanian and O. & Udrea, “Machine recognition of human activities: A survey,” Transactions on Circuits and Systems for Video Technology, vol. 18, no. 11, p. 1473–1488, 2008.

[3] A. Bobick and J. & Davis, “The recognition of human movement using temporal templates,” Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 23, no. 3, pp. 257-267, 2001.

[4] H. Meng, N. Pears and C. & Bailey, “A human action recognition system for embedded computer vision application.,” in Computer Vision and Pattern Recognition (CVPR), vol. 2, no. 1, pp. 21-22,2007.

[5] J. W. Davis and A. & Tyagi, “Minimal-latency human action,” Image and Vision Computing (IVC), vol. 24, no. 5, pp. 455-473, 2006..

[6] V. Kellokumpu, M. Pietikainen and J. & Heikkila, “Human activity recognition using sequences of postures,” in International Association of Pattern Recognition (IAPR), 2005.

[7] M. D. Rodriguez and J. S. M. Ahmed, “A spatio-temporal maximum average correlation height filter for action recognition,” in Computer Vision and Pattern Recognition (CVPR), 2008.

[8] T. T. Thanh, T. Thang, FanChen, K. Kotani and B. Le, “Extraction of Discriminative Patterns from Skeleton Sequences for Accurate Action Recognition,” Fundamenta Informaticae - Computing and Communication Technologies, vol. 130, no. 2, pp. 247-261, 2014.

[9] “The Teardown,” Engineering & Technology, vol. 6, no. 3, pp. 94-95, 2011.

[10] P. Huang and S. J. Hilton A., “Shape Similarity for 3D Video Sequences of People,” In International Journal of Computer Vision (IJCV) special issue on 3D Object Retrieval, vol. 89, no. 2-3, pp.362-381, 2010.

[11] Y. Sheikh and M. S. M. Sheikh, “Exploring the space of a human action,” In IEEE International Conference on Computer Vision (ICCV), vol. 1, p. 144–149., 2005.

[12] L. Xia, “UTKinect-Action Dataset,” [Online]. Available: http://cvrc.ece.utexas.edu/KinectDatasets/HOJ3D.html.

[13] C. Sinthanayothin and W. B. Nonlapas Wongwaen, “Skeleton Tracking using Kinect Sensor & Displaying in 3D Virtual Scene,” International Journal of Advancements in Computing Technology (IJACT), vol. 4, no. 11, pp. 213-223, 2012.

[14] J. K. Aggarwal, L. Xia and C. C. Chen, “View Invariant Human Action Recognition Using Histograms of 3D Joints,” in CVPRW, pp. 257-267, 2012.

[15] A. Baak, M. Müeller and H.-P. Seide, “An Efficient Algorithm for Key frame-based Motion Retrieval in the Presence of Temporal Deformations,” in MIR '08 Proceedings of the 1st ACM international conference on Multimedia information retrieval, vol. 23, no. 3, pp. 257-267, 2008.

[16] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman and A. & Blake, “Real-Time Human Pose Recognition in Parts from a Single Depth Image,” in Computer Vision And Pattern Recognition (CVPR), pp. 57-60,2011.

[17] M. Z. Uddin, N. Duc Thang and T.-S. & Kim, “Human Activity Recognition Using Body Joint-Angle Features and Hidden Markov Model,” Electronics and Telecommunications Research Institute (ETRI), vol. 33, no. 4, pp. 569-579, 2011.

[18] J. Yamato, J. Ohya and K. Ishii, “Recognizing human action in time-sequential images using hidden Markov model,” in Computer Vision and Pattern Recognition (CVPR), pp. 40-45, 1992.

[19] M. Ankerst, G. Kastenmüller, H.-P. Kriegel and T. Seidl, “3D Shape Histograms for Similarity Search and Classification in Spatial Databases,” Lecture Notes in Computer Science, vol. 1651, pp. 207-226, 1999.

[20] R. Lublinerman, N. Ozay, D. Zarpalas and O. Camps, “Activity recognition from sil- houettes using linear systems and model (in) validation techniques.,” in In International Conference on Pattern Recognition (ICPR), 2006.

[21] S. Savarese, A. DelPozo, J. Niebles and L. Fei-Fei, “Spatial-temporal correlatons for unsupervised action classification.,” in Workshop on Motion and Video Computing (WMVC)., pp. 117-121, 2008.

[22] L. Xia, C.-C. Chen and J. K. Aggarwal, “Human Detection Using Depth Information by Kinect,” in Human Activity Understanding from 3D Data in conjunction with CVPR (HAU3D), vol. 12, no. 2, pp. 557-567, Colorado, 2011.

[23] X. Yang and Y. Tian, “Effective 3D Action Recognition Using Eigen Joints,” Journal of Visual Communication and Image Representation, vol. 25, no. 1, pp. 2-11, 2014.

[24] W. Li, Z. Zhang and Z. Liu, “Action recognition based on a bag of 3D points,” in CVPRW, 2010.