博士論文

Autonomous Human Action Recognition based on a Novel Incremental Approach of Clustering

(新しいインクリメンタル分類法を用いた
人間行動の自律的認識法の研究)

ワン　ウエイホン

博士論文
A Doctoral Dissertation

Autonomous Human Action Recognition based on a Novel Incremental Approach of Clustering

(新しいインクリメンタル分類法を用いた
人間行動の自律的認識法の研究)

氏名: ワン　ウエイホン
Name: Ong Wee Hong

指導教員: 古関隆章教授
Supervisor: Professor Takafumi Koseki

平成 26 年 9 月
September 2014

Abstract

Human activity support or assisted living systems are useful to address various social needs in the modern society such as personal assistant and elderly care. For such systems to sensibly support us, it is useful for the systems to understand what we are doing. Human Activity Recognition (HAR) is therefore an important component in a system that will support human activities. However, we have not seen wide adoption of HAR technologies in our homes.

Two main hurdles to the wide adoption of HAR technologies in our homes are the expensive infrastructure requirement and the limitations in the HAR technologies. Many HAR researches have been carried out assuming an environment embedded with sensors. In addition, the majority of HAR technologies use supervised approaches, where there are labeled data to train an expert system.

In reality, our natural living environment are not embedded with sensors. Labeled data are not available in our natural living environment. The training and labeling processes are inconvenient for users. Further more, there lack a framework for HAR in an autonomous manner for our natural living environment.

In this dissertation, we have proposed a framework for autonomous HAR suitable in our natural living environment, i.e. the sensor-less homes. We consider human action as the primitive building blocks for human activities. The ability to recognize actions will enable recognition of higher level activities. For this framework, we have considered suitable set of features for generalization of human action representation. We define the feature set based on human range of movements. The framework is complete with three essential phases or stages: discovery, learning and recognition of human actions.

We developed strategies and unsupervised learning approach to autonomously carry out the three stages without requiring user annotation and operation. Unsupervised learning is less developed in comparison to supervised learning algorithms due to the added uncertainties in unlabeled data. In unsupervised learning, in particular the clustering algorithms, a major difficulty is to determine the number of classes or clusters in the data without prior knowledge. We developed an incremental approach of clustering to solve the problem of the number of clusters in action discovery. Given the nature of our problem, our clustering approach does not determine a definite number of clusters. Instead, it looks for good clusters in an incremental manner. Utilizing this clustering approach in the discovery phase, we use probabilistic modeling, Mixture of Gaussians Hidden Markov Model, to learn and recognize human actions.

We have evaluated the framework with our own dataset as well as a third party dataset. The results indicate the potential of the framework to autonomously discover, learn and recognize human actions.

The contributions of this dissertation are summarized as below:

- It enabled discovery of human actions in our natural living environment through an incremental approach of clustering.
- It developed a framework for autonomous Human Action Recognition incorporating a novel human action discovery technique
- It is the first study in the unsupervised approach of Human Action Recognition using data from low-cost depth sensor.
- The development of the autonomous Human Action Recognition framework will improve usability of HAR technologies in our homes.

Acknowledgements

The course of my doctoral study has been an extra ordinary experience. It is a journey full of uncertainties and hurdles. It is a journey that I cannot make it to the end if I had been alone.

I owe my gratitude to many people who have helped, supported and encouraged me during the course of the doctoral study.

I owe my deepest gratitude to Professor Takafumi Koseki.

Professor Koseki has accommodated me when I had nowhere to go. He has continued to trust me even when I had lost my confidence. He has been tolerating my shortcomings in doing research. He has provided me with the best support to conduct my research. He has been patiently guiding me to complete my doctoral study. I would not have made it this far if it wasn't of his kindness.

I am grateful to the other professors in the review committee of my dissertation: Professor Katsu Ikeuchi, Professor Takashi Kubota, Professor Hiroyuki Ohsaki, Professor Kaoru Sezaki and Professor Hitoshi Iba. They are lenient to me despite my shortcomings, and they have given constructive comments to help improve my dissertation and complete my doctoral study.

I am grateful to the Universiti Brunei Darussalam and the Government of Brunei Darussalam for giving me the opportunity to do my doctoral study, and for generously sponsoring my cost of study and living in Japan.

I very much appreciate the support from the Embassy of Brunei Darussalam in Tokyo during my stay in Japan. Hj Hazri Hj Julay from the Embassy of Brunei Darussalam and Doris Wong Pek Leng of the Public Service Department have been particularly helpful.

I am thankful to Professor Hideki Hashimoto for giving me the opportunity to study in The

University of Tokyo and for connecting me to Professor Koseki.

I am grateful to Professor Mihoko Niitsuma for accommodating me in her laboratory when I was struggling to find my research direction.

I deeply appreciate the help and friendship from many people including everyone in the Koseki Laboratory as well as many others.

Takada-san has been very helpful to give the best support to facilitate my research. He has always given more than I asked for. It is always refreshing to see the cheerful Matsuzaki-san, who has been helping me with matters pertaining to university procedures, as well as the submission of my dissertation. Travis's help with the submission of my dissertation and his comments in the preparation of my pre-defense are of great help to me. The very constructive comments and suggestions from Valerio have been of great help to me. Valerio has helped to steer my research direction into one that is appropriate at a doctoral course level. Having Shin-san around is an essential part of my life during the course of my study in Todai. Sharing grievances, staying late at night in the laboratory and the mutual encouragement with him have relieved a significant amount of stress and depression from the doctoral study.

Leon has persistently been keeping me company in my research work. Without him, I have doubt I can make it this far. His knowledge in machine learning has helped me in developing machine learning tools to solve my research problems.

Hiromu-san's friendship has made the time I spent in Niitsuma Laboratory a comfortable one.

Peshala has been generous to share his experience and materials in the doctoral course. His materials have served useful references to me.

It was good to have the occasional chit chat with Donny when I needed a break from the lab. Donny has helped me in dealing with some of the mathematical aspects in computer vision.

The list could go on, however this should not be the biggest chapter of this dissertation. I shall stop at this point.

Last but not least, I owe my sincere apology to my wife and family that I had not been a responsible member in the family for more than three years. I am especially thankful to them for being understanding. Without their that, I would not be able to pursue my doctoral study.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The focus of the work reported in this dissertation is to address an important issue in deploying human activity analysis technologies in our natural living environment or the normal household setting.

In its broad sense, human activity analysis is a research area covering various technologies required for understanding the activities taking place in our everyday life. It has found its applications in ambient intelligence, assisted living, human-machine interaction, security surveillance and media analysis. Human activity analysis involves at least two processes: discovery of the activities and recognition of the activities. Despite that recognition is part of understanding human activities, the term Human Activity Recognition (HAR) has been commonly used to describe the general human activity analysis process encompassing both discovery and recognition. In this dissertation, we will use HAR to refer to the general human activity analysis, and we will specifically point out discovery or recognition phase when we are referring to the specific process.

In this dissertation, we present an autonomous human action recognition framework using data from a low-cost device that we believe will facilitate the wide adoption of human activity analysis technologies for applications in the normal household settings. By autonomous, the framework completely avoid manual process in all stages of its operation. It does not require manual segmentation of input data. It does not require manual annotation of the training data. It determines all necessary parameters in its learning algorithms without requiring pre-specification.

In this chapter, we explain the motivation behind our approach, clarify the use of our terminology and describes the contributions of this dissertation.

1.1 Motivation

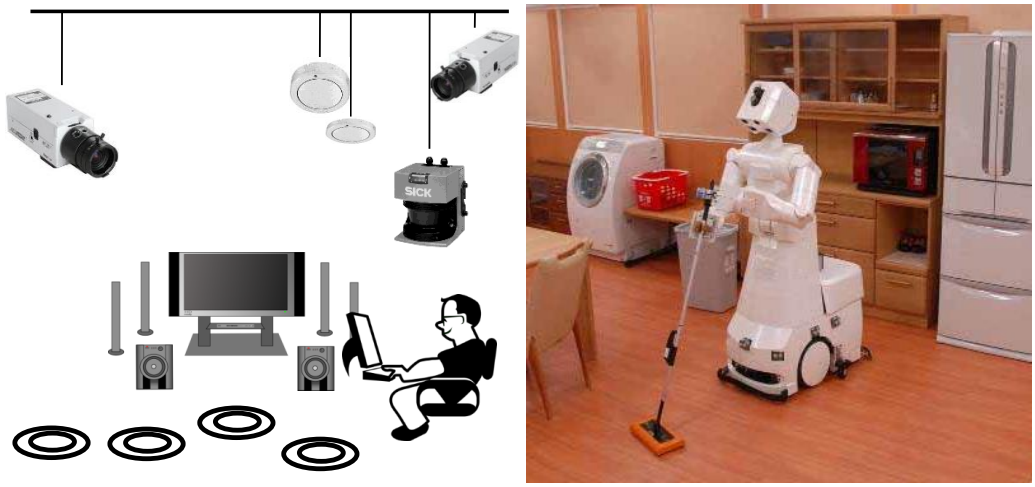
Most people in the modern society live a busy life and spend most time away from home. Many of them stay away from home town living alone and leaving their family, especially the elderly, living on themselves. Here we are talking about two parties: the younger generation that lives alone in work city, and the older generation that lives alone in home town. Both generations require support to maintain their personal life in both physical and psychological aspects. On the physical aspect, help can be sought from professional or the society such as engaging nurse, nanny, maid or resorting to nursing home. On the psychological aspect, pets offer compensation for loneliness. However, these options may not be always available, affordable and may not be favorable to all concerned people.

Alternatively, technologies can be deployed to provide solutions to support the needed people. Such technologies are often referred as assisted living. For an assisted living environment to be sensitive and responsive to the presence of people, its ability to understand human activities is a fundamental requirement. Human Activity Recognition (HAR) is therefore at the core of the various technologies in an environment that supports human daily activities.

HAR technologies are typically developed for an environment where extensive sensors are embedded, i.e. sensor-ed space. To date we have not seen wide adoption of HAR in our homes without embedded sensors, i.e. sensor-less homes. Such environment at current stage is expensive and obtrusive. It requires modification to existing home and installation of potentially expensive sensors.

There are a number of issues that hinder the use of HAR technologies in our natural living environment, i.e. sensor-less homes. We identify two of them as being relevant in this dissertation.

One is the difficulty in obtaining reliable data for HAR without the sensor-ed space. The use of vision sensor or camera is an approach to avoid requiring large number of sensors embedded



(a) (b)

Figure 1.1: Assisted living technologies. (a) Ambient Intelligence. (b) Home-Assistant Robot [1].

Figure 1.2: A sensor-less home.

in the environment or requiring users to wear sensors. A single camera can potentially capture a human subject in its wide field of view and observe what the subject is doing. However, doing HAR from vision data is challenged by the difficult computer vision problems. Detecting human from a scene requires dealing with illumination change, background clutter, motion of sensor and change of view point. To improve the reliability of the vision data, often high quality cameras are required.

Another issue is that the majority of HAR technologies use supervised learning algorithms.

Due to the need for labeled data and the use of supervised approaches, the generalization of HAR technologies in natural human living environments is limited. Human perform their daily activities in wide variation. It is difficult to come up with a definite set of human activity models that will suit everyone. Ideally, we want to learn the activities of a user from observing the user himself or herself. In natural human living environments, such observations are not labeled.

For human activity recognition technologies to be deployed in such environments, we require the use of unsupervised approach. And, we want to use "cheap" data.

Here's the scenario of deploying supervised HAR by converting a normal home into a smart environment. We purchase a system from a service provider. The service provider makes necessary installations in our house including installations of sensors and intelligent devices. The server of the system is pre-programmed with a set of activity models. The system is capable of detecting and recognizing the set of defined activity models. When we require the system to recognize activities not defined the system, we contact the service provider. They come to collect samples of the activities we want the system to recognize. We work with the service provider to train the system for the new activity models. We perform the activities multiple times while the service provider collect the examples and label them. The system is trained to recognize the new activities. The system and the re-training both cost us good amount of money and time.

Here's the scenario of deploying HAR by incorporating our unsupervised framework. We purchase a nice looking personal robot off the shelf. Let it accompany us at home. It appears to just follow us around like a pet robot. Under the hood is the unsupervised HAR technology. The personal robot starts without knowing any activity model. It simply follows us and record our activities. It discovers activities as it observe our daily activities. It learns new activity models by itself and recognize the learned activities. It learns more activity models as it spends more time with us. It communicates with other intelligent devices such as smart phones and computers to provide information regarding our daily activities for further actions. The personal robot costs us about the price of an average computer. It is a one device and one implementation that adapts to different users.

We are motivated by the intention to facilitate the deployment of HAR technologies in natural

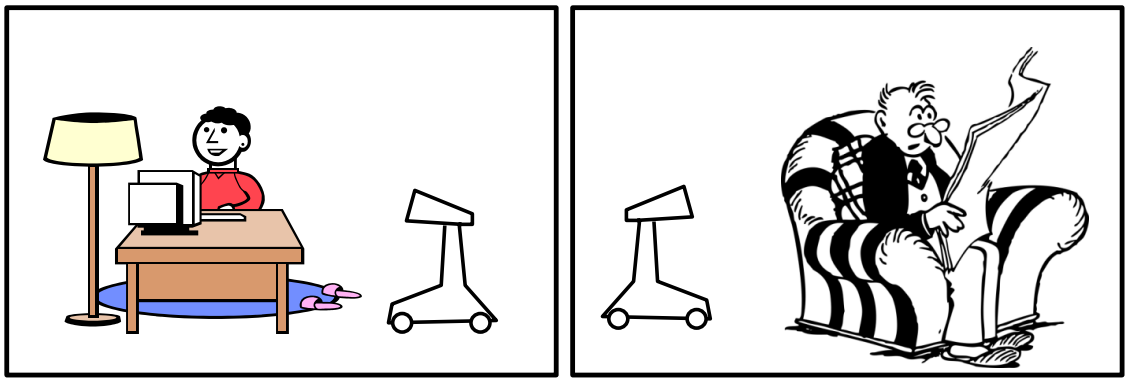


Figure 1.3: One device, one implementation that adapts to different users.

human living environments with the following characteristics:

- A minimal financial cost, i.e. low-cost.
- Require minimal modifications to the living environment of the user, i.e. unobtrusive and low-cost.
- Avoid the use of wearable sensors for the users, i.e. natural and unobtrusive.
- Autonomous in its operation, i.e. user friendly
  - Do not require reprogramming of the system to learn new activities.
  - Do not require manual segmentation of the vision data.
  - Do not require manual annotation of the data.
- Able to learn indefinite number of activities from a single algorithm implementation, i.e. scalable.

1.2 Action versus Activity

Different terminologies have been used to describe human activities at different level of complexity. The boundaries of the terminologies are not well defined and different researchers will have their own definition. For examples, Aggarwal and Ryoo [5] categorize activities into five levels:

gestures, actions, interactions and group activities; whereas Moeslund *et. al.* [6] use the following hierarchy: action primitive, action and activity. Turaga *et. al.* [7] defined "Actions" as simple motion patterns typically executed by a single human and "Activities" as more complex and involve coordinated actions among a small number of humans.

In our context, we simply define an action as an atomic activity that cannot be further decomposed into meaningful intentional act. Activities are composed of series of actions. For example, cooking is an activity comprising various actions including chopping and stirring. The activity can be inferred if we know the actions.

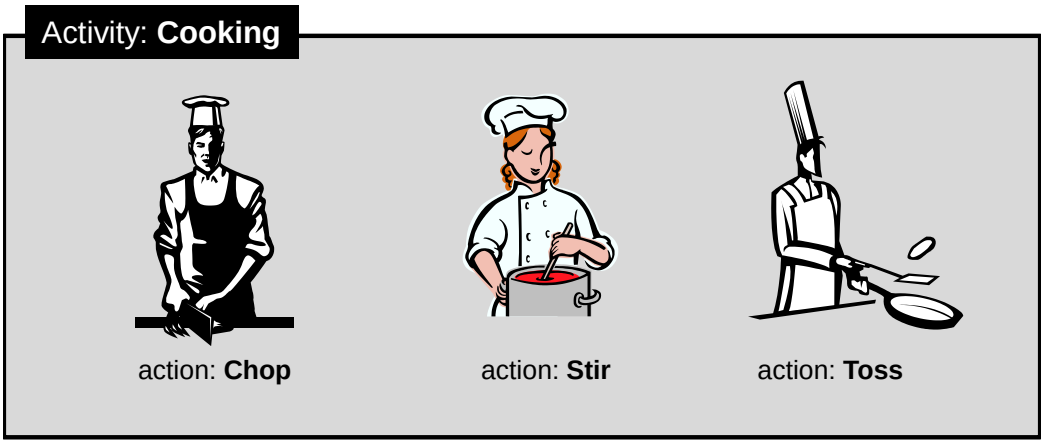


Figure 1.4: Given action(s), we can infer activity.

We have used the acronym of “HaR” to represent Human Action Recognition. The lower case “a” distinguishes action from activity. While we are particular in the use of the terminology of action, all discussions on HAR in this dissertation are relevant to HaR.

1.3 Contributions

This dissertation makes the following contributions.

1. *Unsupervised human action recognition based on Kinect data.* It investigated the use of data from a low-cost depth sensor, the Microsoft Kinect, for human action recognition in an unsupervised manner.

2. *Investigated generalized features for human action recognition.* It proposed the use of features based on the human range of movement and investigated the effect of reduced frame rate. It investigated the negative effect of dimension reduction using correlation.

3. *Human action recognition in three phases.* The division of the human action recognition framework into three stages or phases enables handling of human action recognition using different approaches in each stage to achieve autonomous human action recognition.

4. *Incremental approach of clustering.* For the discovery phase, a novel incremental approach of clustering has been proposed. The use of unsupervised learning algorithm avoided the need for user annotations and retraining for individual action. The clustering approach does not require prior knowledge of the number of actions in the data. It deals with an undefined number of actions and rejects random movements or noise. The incremental approach resembles the way children learn their environment by observation over long period of time.

5. *Parameters of Mixture of Gaussians HMM.* In the learning phase, a strategy to autonomously learn the parameters for Mixture of Gaussians Hidden Markov Model (MG HMM) has been developed. This strategy enables self-learning of the action model for each of the discovered action from earlier phase.

6. *Human action recognition in normal home setting.* This dissertation is a ground work for improving usability of human activity recognition technologies in normal home setting. It developed human action recognition technique suitable for normal home setting where embedded sensors are not available and wearable sensors are not desirable. It uses low-device and is transparent to the users, i.e. no user annotation and training.

1.4 Dissertation Organization

This dissertation comprises of seven chapters.

The main purpose of Chapter 1 is to highlight the contributions of this dissertation. The account of the contributions sets the context and scope of this dissertation.

Chapter 2 provides the literature review of the existing research related to human action recognition. It highlights the gaps that this dissertation will fill. It specifies the research problems that this dissertation addresses.

Chapter 3 describes the autonomous HaR framework being proposed in this dissertation, while Chapter 4 and Chapter 5 further elaborate on the details of the two major phases in the framework.

In Chapter 6, a prototype implementation of the autonomous human action recognition framework is described. It intends to show how the proposed framework can be used in real life situation.

Finally, conclusions are made in Chapter 7, with pointers for further research directions that can be extended from this dissertation.

Fig. 1.5 shows the organization of this dissertation.

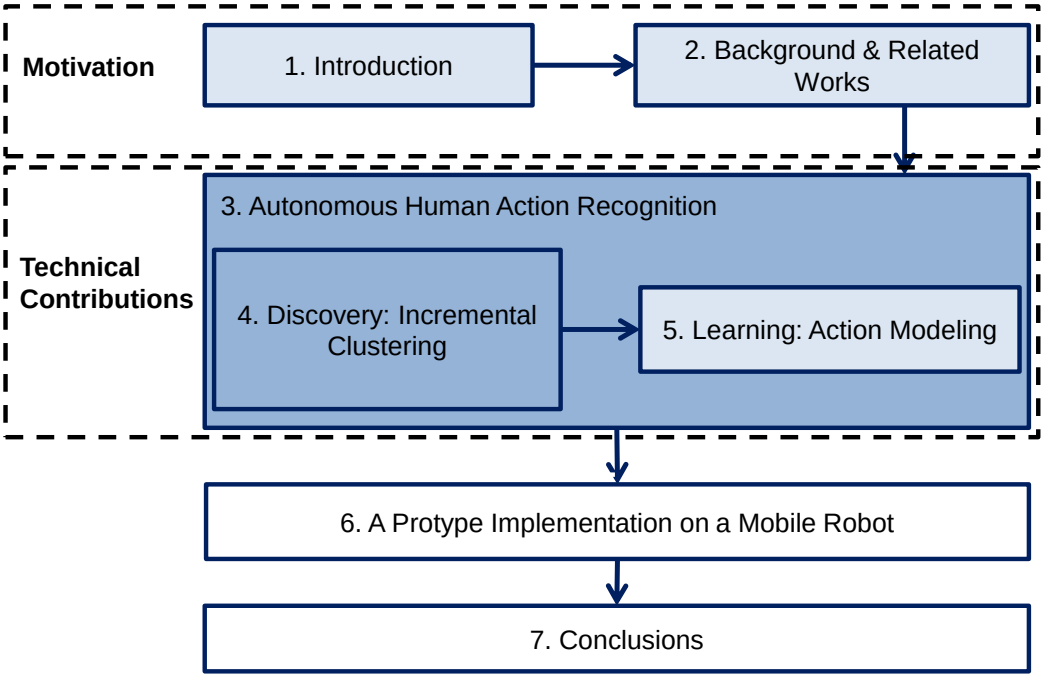


Figure 1.5: Organization of the dissertation.

Chapter 2

Background and Related Works

Human Activity Recognition (HAR) technologies can be broadly divided into two categories: sensor-based and vision-based. Sensor-based HAR uses sensors either installed in the environment or as wearable sensors positioned directly or indirectly on human body. Chen *et al.* [8] presented a comprehensive survey on the development and current status of various aspects of sensor-based HAR. Due to the fact that human living premises are not normally installed with extensive sensors and that wearing sensors can be obtrusive, sensor-based HAR technologies have been limited to laboratory settings, specially conditioned premises and visionary smart environment.

Given the above arguments, our work is based on vision-based HAR.

2.1 Vision-based Human Activity Recognition

Vision-based HAR offers advantages in ease of use and non-obtrusiveness as it can potentially work without wearable sensors. For example, a single security camera can potentially capture a good number of activities within a wide field of view as compared to the requirement to install multiple sensors on site.

HAR has therefore been an important area of computer vision research. Vision-based HAR uses the data from vision sensors to recognize activities. The technology has been used to explore video content, in interactive applications and animation production to name a few.

Turaga *et al.* [7] presented a comprehensive survey of the efforts to address the problems of representation, recognition and learning of human activities from video and related applications. Their survey described the different methods to model activities at different level of abstractions. In a survey on vision-based human action recognition, Poppe [9] addressed the challenges of vision-based action recognition. In the survey, Poppe discussed technologies for image representation and action classification. A recent review from Aggarwal *et al.* [5] continues to highlight that human activity recognition is an important area of computer vision research. They discussed both the methodologies developed for simple human actions and those for high-level activities. Their review made comprehensive comparison between the different human activity recognition technologies.

From the three survey papers, we note three observations.

1. Firstly, solving computer vision problems represent a significant amount of effort in vision-based HAR. The problems remain unsolved and reliable detection of human from the visual data remains a challenging task.

2. Secondly, commercial use of HAR technologies in normal home setting has not been widely realized.

3. Thirdly, perhaps the most important hurdle in realizing HAR in normal home setting is that all the works discussed in the above survey papers require labeled data. They use supervised approach to learn the activity models. In normal home setting, labeled data are scarce and requiring user annotation of daily activities is impractical.

2.2 Commercial Vision Sensor with Depth Information

While vision-based technologies offer the benefit of easy installation and non-obtrusiveness, a few issues have hindered its wide adoption in normal home setting. The challenging computer vision problems being an important issue. Detecting human in a visual scene has been made difficult with changes in illumination, cluttered background, motion of the sensor and view point. To reliably detect human body from the scene, high quality cameras are often required. This translates into

high cost requirement. Another issue is privacy concern of having camera to monitor our activities, although we have seen increasing acceptance of surveillance cameras in residential premises.

With the availability of low cost commercial depth sensors, fast and accurate extraction of human postures have been significantly improved. We can obtain the 3-D coordinates of the joints of human body reliably with little influence from varying illumination in indoor environment. In other words, the availability of depth information from such sensors has overcame the computer vision problems to a great extend.

Figure 2.1: Commercial RGB-Depth sensors.

The Microsoft Kinect [10], a consumer grade RGB-D (RGB-Depth) sensor, has successfully been used in game console where human interact and play the game without game controller and without wearable sensors or markers, through a natural user interface using gestures and spoken commands. Xia *et al.* [11] demonstrated the capability of the Kinect to detect human in visual scenes. They presented a novel 2-stage model based human detection method using depth information taken by the Kinect. The work by Shotton *et al.* [12] is another evidence of the ability of the consumer depth sensor to provide highly reliable 3-D positions of human body joints. Their method uses randomized decision forests and can estimate body part labels without invariant to pose, body shape and clothing in under 5ms. We expect to see further improvement in detection of human body joints using depth sensor.

Consumer grade depth sensors open up opportunity to bring HAR technologies into the home of general consumers. A number of works on human activity detection using depth sensor have been published since the introduction of Microsoft Kinect. Sung *et al.* [13] used a hierarchical approach to learn human activities from data obtained from Kinect. In their work, they used 3-D

joint coordinates and joint orientation to compute the features. They further improved the result by adding Histogram of Oriented Gradients (HOG) feature descriptors of the RGB and depth images [4]. Li *et al.* [14] performed human action recognition based on a bag of 3-D points in the depth images and achieved high accuracy. Wang *et al.* [15] used ensemble model to represent and learn human actions from the 3-D joint positions and local occupancy pattern or LOP features. In their work, LOP is the "depth appearance" of each 3-D joint. A recent work by Chen *et al.* [16] continued to take advantage of the depth data in human action recognition. They proposed an approach to recognize single-person action based on spatio-temporal local features and a bag-of-words model.

All the works on HAR using depth sensor data that we came across have been using supervised approach. In supervised approach, each activity model is learned from annotated examples. The approach require manual collection of the activity examples and label the examples according to the corresponding activity. In the case of the gaming console using Microsoft Kinect, the gestures were learned from large number of labeled examples.

2.3 Unsupervised Human Activity Recognition

For wide adoption of HAR technologies in normal home setting, it is necessary to learn activity models from unlabeled data. This requirement calls for unsupervised approach in human activity recognition. This requirement calls for technologies that can discover activities by themselves. Kim *et al.* in their review paper [17] pointed out that human activity understanding encompasses activity recognition and activity pattern discovery. Activity discovery is the automatic recognition of activity patterns in an unsupervised manner. Activity discovery has received less attention than activity recognition. It is a challenging task to deal with unlabeled data.

Wyatt *et al.* [18] published the first unsupervised activity recognition work. Their approach was unsupervised in the sense that labeling of the data was automated. They used the signals from wearable sensors and object tagging to infer activities from object interactions. They tagged over a hundred objects in a normal home. They mined from the web to associate object interactions

with activities to automate the annotation process. They dealt with a fix number of activities and there was no notion of noise.

Huynh *et al.* [19] used topic models to automatically discover activity patterns in user's daily routines. They used wearable sensors at wrist and in hip pocket to provide the data stream and used data clustering to generate a vocabulary of a specified size and infer high level activities using topic model with bag-of-words representation. Their approach assumed a fixed number of activities and did not have the notion of noise.

Stikic *et al.* [20] used sparse labels to label nearby unlabeled activities. They assumed similar activities are nearby in time. They proposed a weakly supervised recognition of daily life activities with wearable sensors. They grouped sets of successively recorded activity data into so-called bags-of-activities and assign the sparse labels obtained through experience sampling to bags-of-activities. Users were required to provide minimal labeling of their activities. The number of activities were predefined as the number of bags. Their approach dealt with noise.

Hamid *et al.* [21] presented a framework to discover and characterize different classes of everyday activities from event-streams. They represented activities as bags of event n-grams and used variable length Markov process to model the activities. They dealt with high level activities in unsupervised manner while they provide the set of defined vocabulary to the recognition system. They did not deal with random movements.

Chikhaoui *et al.* [22] proposed an unsupervised model for human activity discovery and recognition in pervasive environments using combination of sequential patterns and latent Dirichlet allocation. Their approach discovered high level activities from event-stream. The signals for the event-stream were derived from sensors embedded in the environment as well as sensors attached to objects that people interact with. The number of activities to discover was fixed in their implementation and they did not deal with noise.

We can see from the above review that a significant number of the works in activity discovery or unsupervised activity recognition are based on event stream of the signals from sensors. They infer high level activities, such as commuting, dinning or lunch, from event stream of the signals from sensors embedded in smart environment or tagged to people and objects. These works are

sensor-based and target applications in smart environment. Sensor-based recognition systems are dominant in the unsupervised approaches due to highly reliable data from the sensors. In certain cases, the sensors greatly ease the detection of activities. For example, a motion sensor and a door sensor can easily detect that a person is entering a room. On the other hand, the capability of the recognition system is limited to the extend of the sensors being installed. For example, motion and pressure sensors can detect that a person is sitting at the dinning table. However, these sensors are insufficient to recognize if the person is reading or having meal unless the person wears sensors to detect his movement. In many research works, the activity at the dining table is pre-defined.

In sensor-based smart environment, high level activities can be recognized from sensor embedded in the environment or tagged to objects, however atomic activities or actions will require wearable sensors to detect hands and legs movements. Also, the algorithm assumed clean data without noise such as random movements. We further note most of the unsupervised activity recognition approaches use fixed size vocabulary in topic model. One major problem with the system that uses predefined vocabulary in the form of bags is that the defined vocabulary restrict the scalability of the system. As mentioned at the beginning of this chapter, sensor-based technologies are not easy to implement in existing human living premises. Our interest is in the vision-based HAR. With vision data, a single camera can potentially capture movements of the whole body.

There are significantly fewer works in unsupervised approach in vision-based than sensor-based human activity recognition. Niebles *et al.* [2] proposed a novel unsupervised learning method for vision-based human action categories using spatial-temporal words. They addressed computer vision problems using probabilistic models to handle noisy feature points arisen from dynamic background and moving cameras. In the training stage, they assumed that there is a single person performing only one action per video. They used unsupervised learning framework to automatically discover semantic clusters called codebook in the training data. The training data are prepared videos of single action. The codebook size was predefined and the words in the codebook were used to represent the videos in an encoded form. A probabilistic topic model was used to learn and recognize the actions in the training data. The number of topics was fixed to the number of actions to be learned. In their approach, the data segmentation was a manual process

and the learning is supervised, i.e. input has been organized into separate classes. They did not deal with random movements as noise.

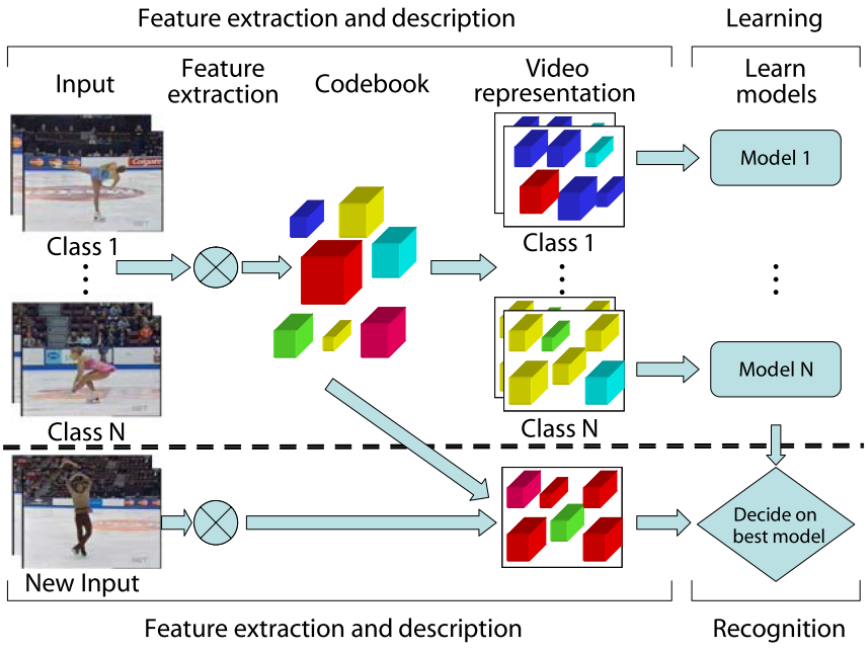


Figure 2.2: Unsupervised learning to discover codebook [2].

Cui *et al.* [23] proposed a matrix-based approach to unsupervised human action categorization. Like Niebles *et al.* [2], their work addressed video content analysis. The input to their system was manually prepared video sequences of actions. The number of actions was known to the system. They represented multi-action into matrices and proposed a matrix factorization method that simultaneously clusters video sequences into action classes. They assumed all observations are valid actions, i.e. there was no notion of noise.

We note both vision-based unsupervised action recognition technologies targeted offline analysis of video sequences. They addressed computer vision problems and had manually prepared video sequences of each actions as their input. Their frameworks knew the expected number of actions to be learned in a supervised manner, while they used unsupervised approaches for feature extraction from vision data.

2.4 Incremental Clustering

The term "incremental clustering" usually refers to clustering approach that deals with dynamic data, and often in an on-line situation. In our work, we have developed an incremental approach of clustering to discover human actions in a continuous basis. The existing incremental clustering algorithms are not designed for human action recognition.

M Charikar et al [24] presented first proposal for incremental clustering to deal with dynamic information retrieval. Their paper presented extensive mathematically proof for the feasibility of the incremental clustering. However, their proposed model was not tested on any dataset. The proposed incremental clustering requires prior knowledge of the data, in particular a suitable threshold to assign each data points into a cluster. It also requires specification of a desired value of k, i.e. number of clusters. The model maintains a hierarchical agglomerative clustering (HAC) and is computationally expensive.

The algorithm proposed by M Ester et al [25] does not use hierarchical approach and is efficient on large databases. They proposed a density based incremental clustering algorithm for mining in a data warehousing environment. They can dynamically generate undefined number of clusters and deal with noisy data. However, the algorithm requires prior knowledge on the data to specify a suitable value of threshold. They have assumed clean data.

J Lin et al [26] proposed an iterative incremental clustering of time series. Their algorithm is based on wavelets and works by leveraging off the multi-resolution property of wavelet decomposition. Their approach did not require specification of a threshold value. However, they fit the features of the data to the learning set and will not be appropriate when we are dealing with data not in the categories already found in the learning examples. Their approach does not deal with dynamic dataset, and requires prior specification of k-value. They have also assumed clean data.

In recent years, there have been a little progress in incremental clustering.

Z Li [27] proposed an incremental clustering for trajectories. The approach requires specification of a threshold for the distance in the clustering algorithm. They dealt with dynamic data and can perform clustering without prior specification of k-value. They have assumed clean data.

S Young [28] proposed a fast and stable incremental clustering algorithm using winner-take-all

(WTA) inspired partition approach. Their algorithm assumes a finite, fixed number of centroids. There was no notion of noise and all data points would be assigned to a cluster.

M Halkidi [29] proposed a semi-supervised incremental clustering algorithm for streaming data. Like the other incremental clustering algorithms, their algorithm requires prior knowledge of the data to specify suitable threshold value. It also requires prior knowledge in the form of constraints derived from partial labeling.

Figure 2.3: Human activities are complex, difficult to generalize, and have wide variation.

In human action recognition, different individuals can perform different actions with different degree of variation. This poses difficulties in determining a global threshold value that has been used in existing incremental clustering approach. Furthermore, many incremental clustering algorithms still require specification of k-value at certain stage. Most of the incremental clustering assumed that input data are not noisy. For human action recognition, it is desirable to avoid the use of global threshold, the need to specify a fixed k-value as well as the approach that fit features to learning set. It is also desirable to be able to deal with the noise, i.e. random movements.

2.5 Summary in Comparison with this Dissertation

Based on the review of the related works, we note that significant amount of work has been done on human action or activity recognition using a supervised approach. The majority of these works were focused in solving computer vision problems. On the other hand, there are less research works on unsupervised approach in human activity recognition than works based on supervised approach.

The majority of the unsupervised approaches were developed for sensor-based environment and they discover high level activities. Smart environment with extensive embedded sensors will have problem detecting basic actions unless users are required to wear sensors. They are also difficult to be implemented in existing human living premises.

Existing vision-based unsupervised approaches in human action recognition target applications in offline video sequence analysis. They address feature extraction in an unsupervised manner, while deal with a finite set of actions.

In comparison, our framework targets application in normal home setting where the observations are continuous and not prepared. In contrast to existing unsupervised approaches in HAR, our framework discovers actions that can be used to compose high level activities. The number of actions to be discovered and learned is not known in advance and our framework is designed to incrementally discover and learn undefined number of human actions.

We note existing unsupervised human activity or action recognition methods do not address the requirement to discriminate random or unintentional movements during the discovery of activities. Our framework is designed to be able to reject random or unintentional movements.

Our framework can be implemented with simple algorithms. We take advantage of depth sensor to overcome the problems of computer vision.

Based on our survey, at this moment, there is no work in unsupervised approach in human activity recognition based on depth sensor data. We do not find existing work that has proposed a complete framework to deal with human activity or action recognition in normal home setting. To the best of our knowledge, our framework is the first framework for a complete human action discovery, learning and recognition that takes into consideration all aspects and parameters required

to implement human action recognition in natural human living environment without requiring modification to the environment, without requiring users to wear sensors and at an affordable price.

Chapter 3

Autonomous Human Action Recognition (auto-HaR)

In this chapter, we describe the overall framework we have proposed for the unsupervised human action discovery, learning and recognition. It provides the big picture of the works in this dissertation and the details of individual phases will be described in subsequent chapters.

3.1 The Proposed Framework

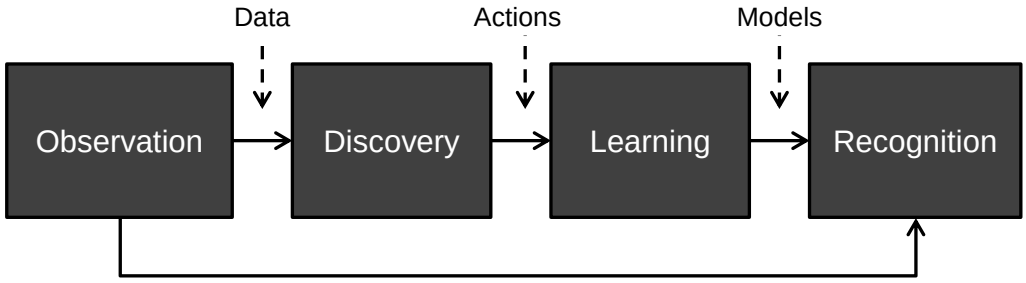


Figure 3.1: Required stages for human action recognition.

A complete human action recognition framework should comprise of at least three stages or phases as shown in Fig. 3.1: discovery, learning and recognition. In addition, an observation stage is required to segment and collect action instances from the sensor.

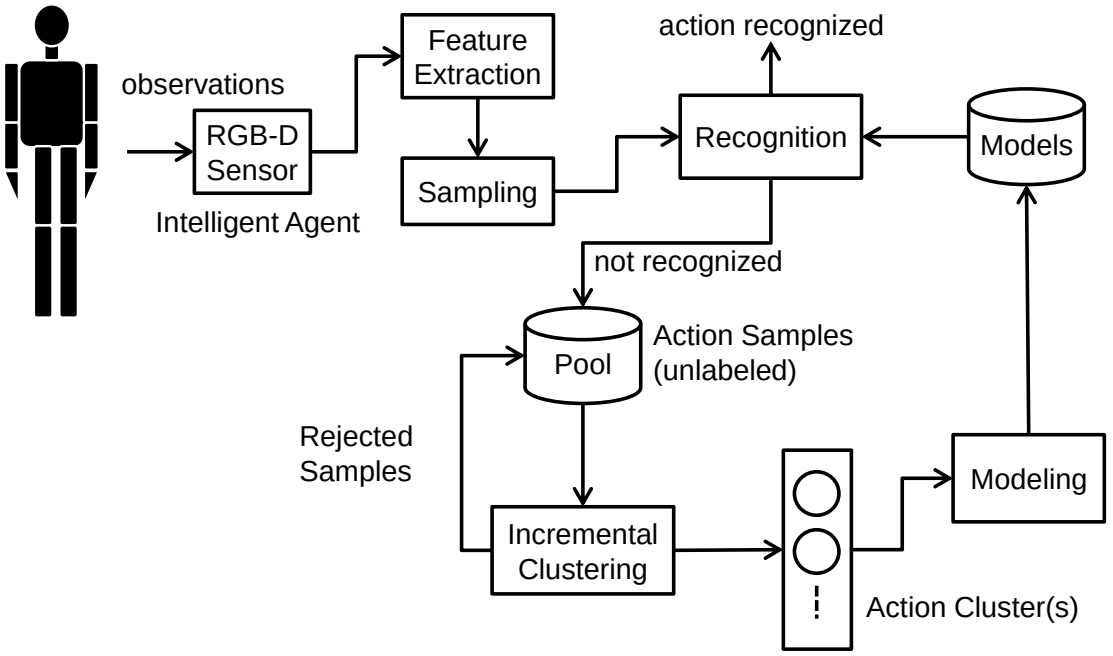


Figure 3.2: Proposed Unsupervised Human Action Discover, Learn and Recognition Framework.

Fig. 3.2 shows our framework to autonomously discover, learn and recognize human actions. We have used the abbreviation "auto-HaR" to label the framework. The system or an intelligent agent continuously observes the person of interest and collect samples of the actions performed throughout a day or a specified period of time. The system performs the process of discovery and learning the models during idle time when the person is not available for observation, for example when the person is not at home or is sleeping. This process of sample collection, discovery and learning is repeated continuously throughout the time that the system or the intelligent agent is with the person of interest. The learned models of actions are used to recognize new observations in real time.

The process in the framework resembles the way children learn to recognize activities, or their environment. Children observe the activities around them without knowing the label of the activities. However, they have the ability to distinguish between different and similar activities. They form models of these groups of activities and eventually label these activities through asking adults. While many activities are taking place, children do not learn all of them at one time. They

learn incrementally.

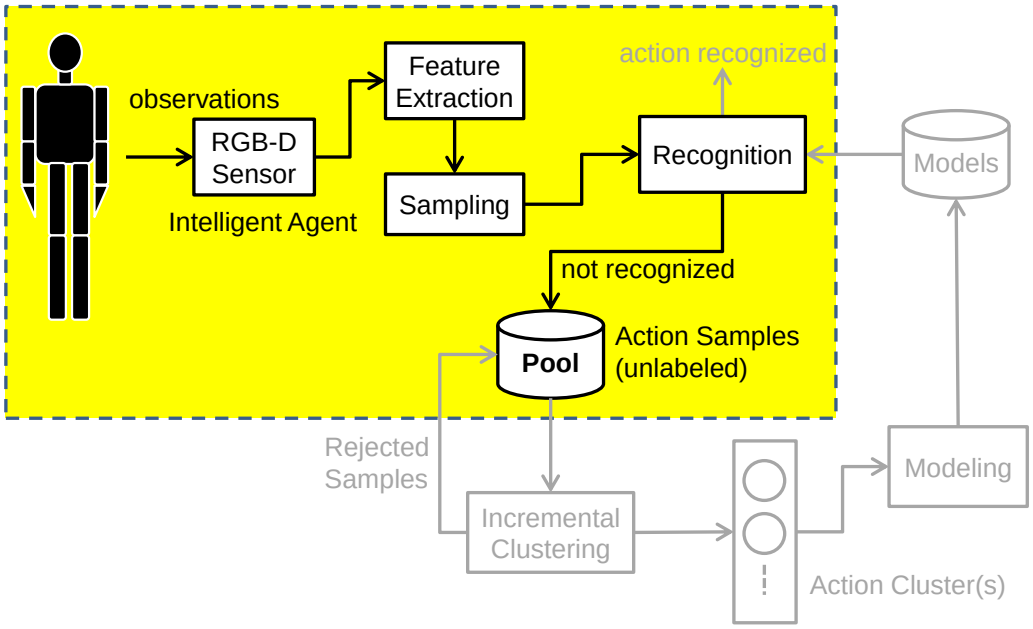


Figure 3.3: Observation phase.

The observation stage as shown in Fig. 3.3 comprises of five components. It starts with capturing the sensor data. Predefined set of features are extracted from the data and the continuous data is sampled or segmented into action instances. Each new action instance is checked against existing action models. Unrecognized action instances are collected into action pool for next phase to discover new actions from the pool. If there is no existing action model, as at the initial use of the framework, all action instances will be collected.

Details of the implementation of the sampling and feature extraction are given in Chapter 6. Details of the recognition module are given in Chapter 5.

Once sufficient action examples have been collected in the pool, the discovery phase will be activated. The discovery phase as shown in Fig. 3.4 comprises of an incremental clustering module. This module performs an incremental approach of clustering on the data in action sample pool to discover new actions by grouping them into separate clusters. The details of the incremental approach of clustering are given in Chapter 4.

The outputs from the discovery phase are clusters of action samples. Each cluster contains

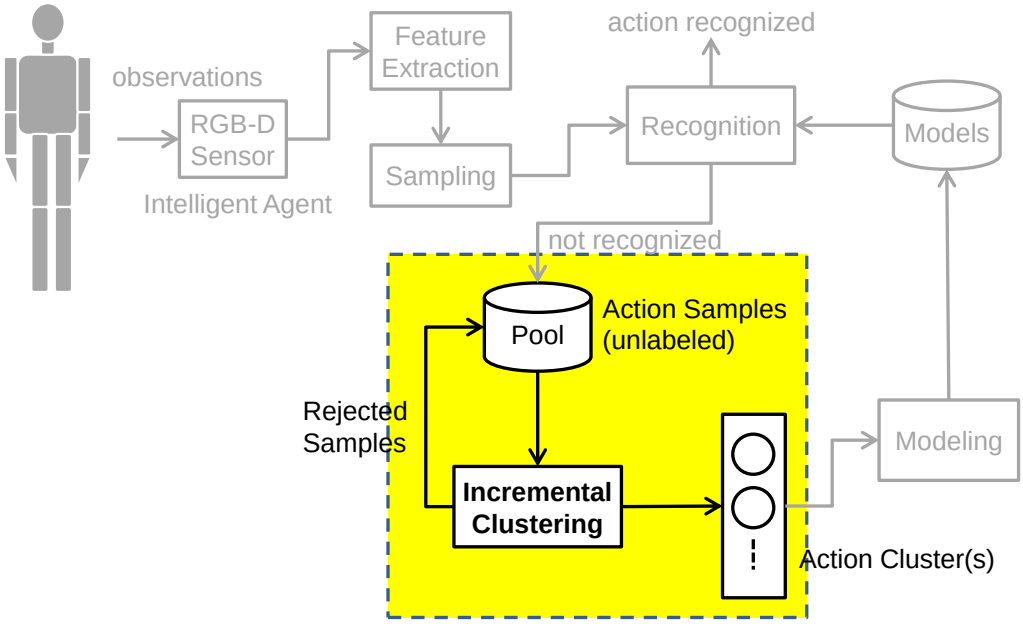


Figure 3.4: Discovery phase.

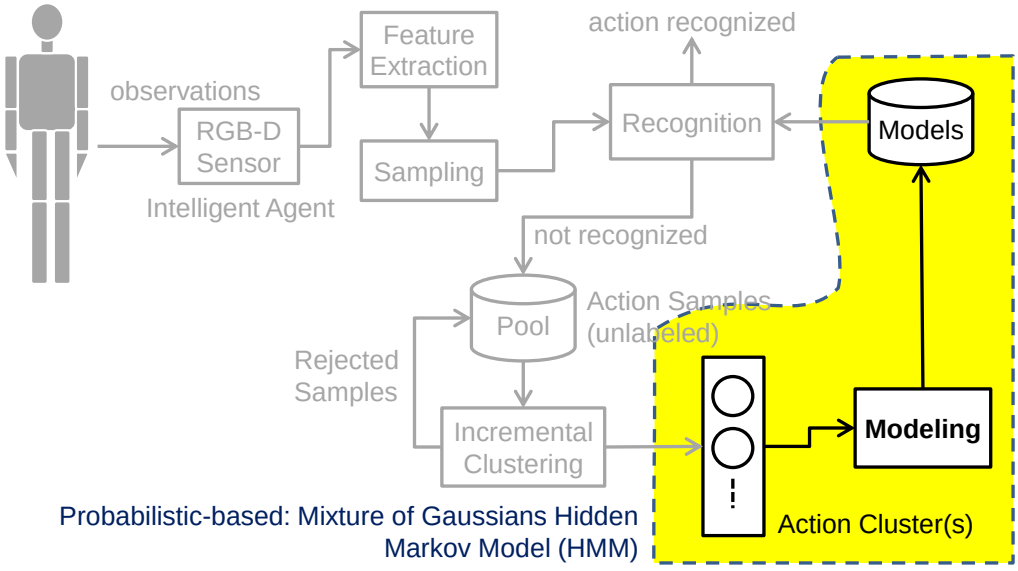


Figure 3.5: Learning phase.

samples of one action. The learning phase will be activated when a new action cluster is discovered. As shown in Fig. 3.5, the learning phase comprises of a modeling module. The discovery phase has collected the examples of a newly discovered action into a single dataset (cluster). It

enables the learning phase to apply supervised learning on the dataset. The learning phase learns a probabilistic model for each action cluster. The algorithm divides a cluster into its own training and cross-validation sets and determines suitable values of the number of states Q and number of mixtures M. The details of the modeling process are given in Chapter 5. The outputs from the learning phase are models of known actions.

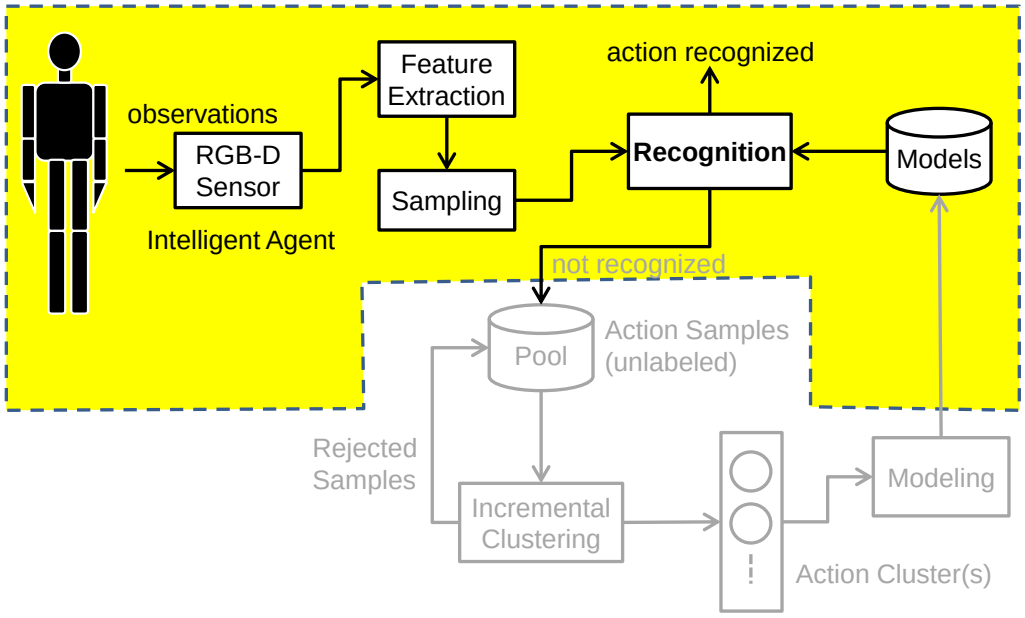


Figure 3.6: Recognition phase.

With the models of known actions, the recognition phase shown in Fig. 3.6 can recognize the known actions from new observations. Any unrecognized action instances are collected to the pool for further discovery of new actions.

3.2 Overall Evaluation

We evaluated the proposed framework on both the learning and test sets of two datasets. Details of the datasets are given in Appendix A. Briefly, the KOS-H16 is a dataset we have collected ourselves. It comprises of sixteen actions. The CAD-60 is a dataset from Sung *et al.* [13]. It comprises of nine actions. Both datasets have random actions included.

3.2.1 Experiment

The learning sets are used for discovery phase, which the outcome is a set of clusters to be fed into the model learning phase. Since the framework does not have labels of the observations, it will learn a model for each cluster irrespective of the purity of the cluster. The test sets are used to evaluate the recognition ability of the learned model. In our experiment, we let the algorithms in the framework to exhaust all samples in each dataset. The datasets have finite samples. When implemented in a real life situation, the algorithms in the framework do not have to discover all actions at once. The algorithm can be implemented to discover and learn a small number of actions within a period of time, and collect more data to discover more actions.

In our experiment, we have used minimum points per cluster $MinPt = 25$, minimum number of mixtures $M_{min} = 1$, maximum number of mixtures $M_{max} = 4$, minimum number of states $Q_{min} = 2$ and maximum number of states $Q_{max} = 6$. Details on these parameters are given in Chapter 4 and Chapter 5.

3.2.2 Performance Evaluation Criterion

To evaluate the performance of the action discovery and recognition, we have used the precision and recall rate.

Precision is the measure of how many of the identified instances are relevant or correct, while recall is the measure of how many of the relevant instances are identified. Ideally we want to achieve high precision and high recall rate.

$$\begin{aligned} Precision &= \frac{identified \quad correctly}{total \quad identified} \\ &= \frac{identified \quad correctly}{identified \quad correctly \quad + \quad identified \quad wrongly} \end{aligned} \tag{3.1}$$

$$
\begin{aligned}
Recall &= \frac{identified \quad correctly}{total \quad truth} \\
&= \frac{identified \quad correctly}{identified \quad correctly \quad + \quad not \quad identified}
\end{aligned}
\tag{3.2}
$$

We do not report the F-score that combines the effects of precision and recall in one figure. It is trivial to compute the F-score from the precision and recall as given in Eq. 3.3.

$$
F_\beta = \left(1+\beta^2\right) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{3.3}
$$

3.2.3 Result and Discussion

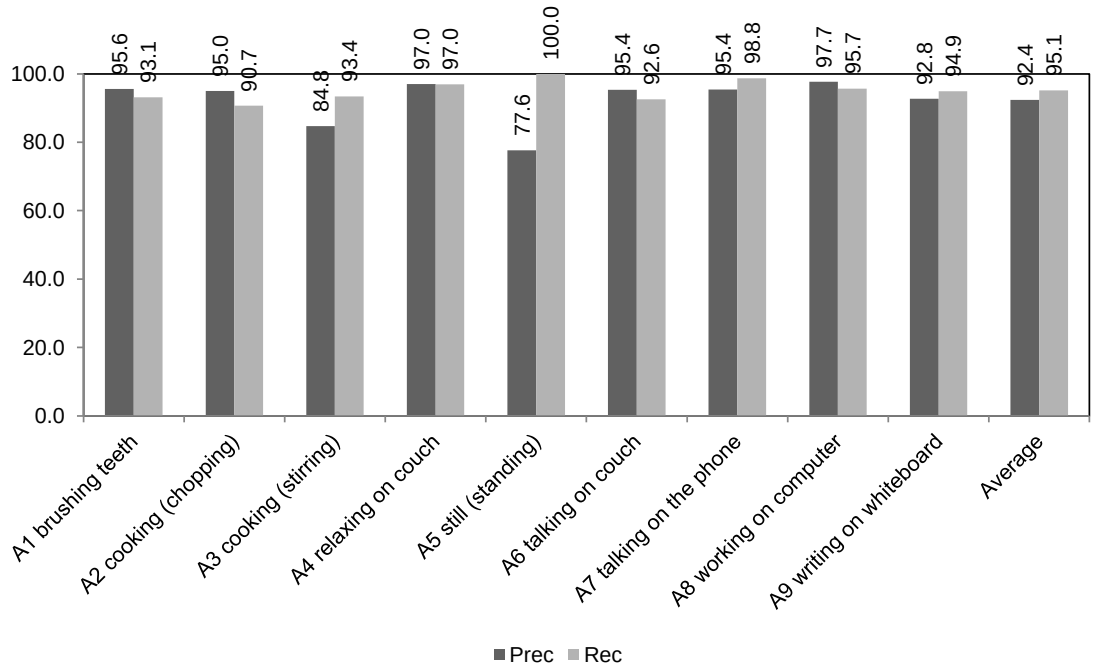


Figure 3.7: Average precision and recall across all four subjects (Person 1 to 4) for each action in the discovery phase.

Fig. 3.7 and Fig. 3.8 show the overall performance of the discovery phase for the two datasets. Table 3.1 and 3.2 give the detailed results.

Fig. 3.7 shows the average precision and recall across all four subjects for each action in CAD-

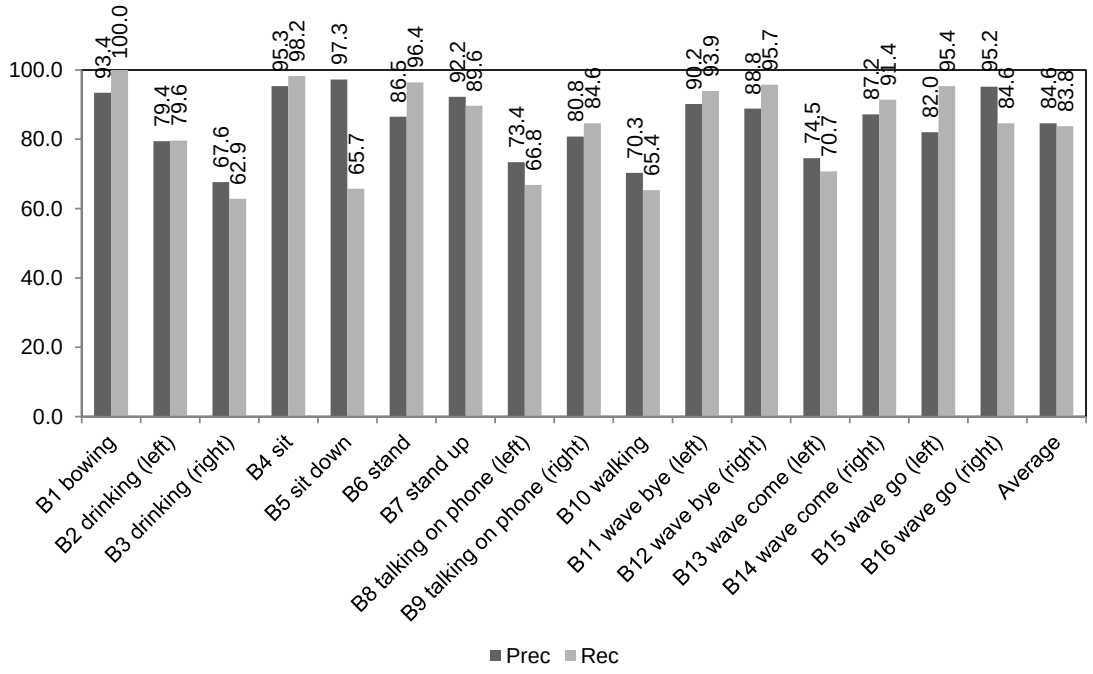


Figure 3.8: Precision and recall across all four subjects (Person 5) for each action in the discovery phase.

60 dataset. The right most bars are the average for all actions. Precision and recall rates are high in all actions except standing. Standing has a significantly low average precision. This is due to the fact that some random action instances resemble the standing action. We will see more details on this matter in Chapter 4.

Fig. 3.8 shows the precision and recall in the discovery phase for each action in KOS-H16 dataset. The right most bars are the average for all actions. In this result, a few actions were discovered with low precision and recall. This is due to the highly similar actions in the dataset. For examples, drinking with left hand (B2) was easily confused with talking on phone with left hand (left) giving low precision and recall in both actions. Waving go with left hand and waving come with left hand are similar, and likewise for the right hand cases. We will see more discussion on this matter in Chapter 4.

The overall performance of the discovery phase has an average precision of 95.1% and recall of 92.4% for CAD-60 dataset, and 84.6% precision and 83.6% recall for KOS-H16 dataset.

Once the discovery phase had discovered the clusters, an HMM model was learned for each

Table 3.1: Precision and recall score in the discovery phase for Person 1 to 4.

	Person 1		**Person 2**		**Person 3**		**Person 4**		**Average**	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
brushing teeth	97.2	96.1	94.4	97.5	95.6	99.6	95.1	79.3	95.6	93.1
cooking (chopping)	99.5	74.6	88.0	95.7	98.9	98.9	93.6	93.6	95.0	90.7
cooking (stirring)	80.7	100	83.8	80.0	91.6	99.6	82.8	93.9	84.8	93.4
relaxing on couch	97.2	88.6	95.3	100	95.6	100	100	99.3	97.0	97.0
still (standing)	82.2	100	77.5	100	76.7	100	74.1	100	77.6	100
talking on couch	95.6	91.1	100	100	93.4	100	92.5	79.3	95.4	92.6
talking on the phone	95.6	99.6	100	100	97.5	97.5	88.6	97.9	95.4	98.8
working on computer	90.7	100	100	96.8	100	86.1	100	100	97.7	95.7
writing on whiteboard	98.9	100	87.6	86.1	100	98.9	84.5	94.6	92.8	94.9
Average:	**93.1**	**94.4**	**91.8**	**95.1**	**94.4**	**97.9**	**90.1**	**93.1**	**92.4**	**95.1**

Table 3.2: Precision and recall score in discovery phase for Person 5.

	Prec	Rec
bowing	93.4	100
drinking (left)	79.4	79.6
drinking (right)	67.6	62.9
sit	95.3	98.2
sit down	97.3	65.7
stand	86.5	96.4
stand up	92.2	89.6
talking on phone (left)	73.4	66.8
talking on phone (right)	80.8	84.6
walking	70.3	65.4
wave bye (left)	90.2	93.9
wave bye (right)	88.8	95.7
wave come (left)	74.5	70.7
wave come (right)	87.2	91.4
wave go (left)	82.0	95.4
wave go (right)	95.2	84.6
Average:	**84.6**	**83.8**

cluster. For those actions that have more than one clusters, they would have more than one model. As this is a fully autonomous system, the algorithms do not know the actual actions in each cluster and would assume each cluster contains instances of a single action.

To evaluate the performance of the learning phase, we test the recognition ability of the learned

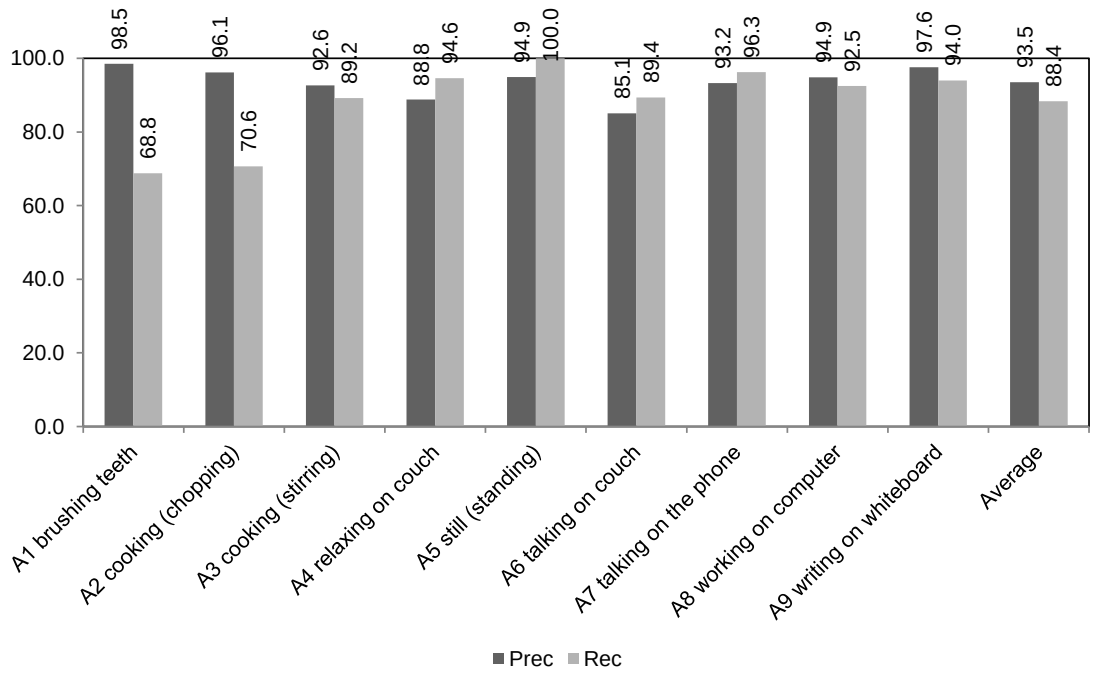


Figure 3.9: Average precision and recall across all four subjects (Person 1 to 4) for each action in evaluation of the learning and recognition phases.

models on unseen observations, i.e., the test sets. The results of the evaluation are summarized in Fig. 3.9 and Fig. 3.10. Table 3.3 and 3.4 give the detailed results of the performance of the learning and recognition phases. The overall performance of the learning phase, i.e., the recognition ability of the learned models, has an average precision of 93.5% and recall of 88.4% for CAD-60 dataset, and 81.9% precision and 72.1% recall for KOS-H16 dataset. The precision and recall in the recognition phases is dependent on the outcome of the discovery phase. They follow similar profiles. Note the accuracy of the learned models depended on the purity of the clusters from the discovery phase.

For the CAD-60 dataset, the low recall rate is due to the poor performance in brushing teeth action (A1). From Table 3.3, we can see that the low recall rate of the brushing teeth action is largely due to the poor performance in the data of Person 4. Apart from this case, the performance to recognize the other actions was good. There are many cases with 100% precision. For the KOS-H16 dataset, due to more actions are similar and confused, a cluster of a confused action from the discovery phase contains instances of confused actions. The model learned from this

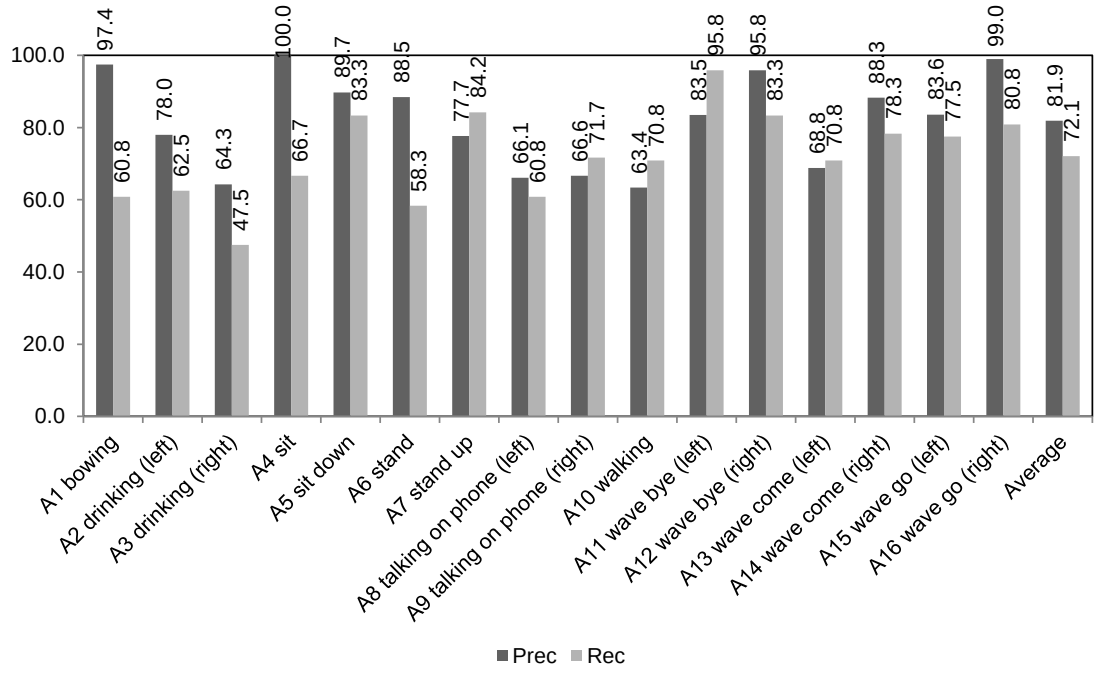


Figure 3.10: Precision and recall for Person 5 for each action in evaluation of the learning and recognition phases.

Table 3.3: Precision and recall in evaluation of the learning and recognition phases for Person 1 to 4.

	Person 1		**Person 2**		**Person 3**		**Person 4**		**Average**	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
brushing teeth	100	81.7	99.0	80.0	100	81.7	95.0	31.7	98.5	68.8
cooking (chopping)	100	48.3	90.8	70.8	100	75.0	93.8	88.3	96.1	70.6
cooking (stirring)	88.0	100	92.3	64.2	96.9	99.2	93.3	93.3	92.6	89.2
relaxing on couch	93.3	83.3	96.0	98.3	72.9	100	93.0	96.7	88.8	94.6
still (standing)	100	100	99.2	100	99.2	100	81.3	100	94.9	100
talking on couch	74.5	100	90.4	95.8	93.5	96.7	81.9	65.0	85.1	89.4
talking on the phone	97.8	100	82.1	92.5	98.5	95.0	94.5	97.5	93.2	96.3
working on computer	92.6	99.2	100	92.5	100	84.2	86.9	94.2	94.9	92.5
writing on whiteboard	100	100	94.8	86.7	100	100	95.5	89.2	97.6	94.0
Average:	**94.0**	**90.3**	**93.9**	**86.8**	**95.7**	**92.4**	**90.6**	**84.0**	**93.5**	**88.4**

cluster would result in recognition at low precision and recall.

To address the issue of confused actions for closely similar actions, more works on feature selection and manipulation will be required. Many actions are difficult to discriminate solely from

Table 3.4: Precision and recall score in learning and recognition phases for Person 5.

	Prec	Rec
bowing	97.4	60.8
drinking (left)	78.0	62.5
drinking (right)	64.3	47.5
sit	100	66.7
sit down	89.7	83.3
stand	88.5	58.3
stand up	77.7	84.2
talking on phone (left)	66.1	60.8
talking on phone (right)	66.6	71.7
walking	63.4	70.8
wave bye (left)	83.5	95.8
wave bye (right)	95.8	83.3
wave come (left)	68.8	70.8
wave come (right)	88.3	78.3
wave go (left)	83.6	77.5
wave go (right)	99.0	80.8
Average:	**81.9**	**72.1**

the vision data. Nevertheless, the results indicate the ability of the framework to autonomously discover, learn and recognize actions from unlabeled data. We summarize the overall recognition performance of the framework in Fig. 3.11. The framework achieved an overall precision of 91.2%, and recall of 85.1% on both datasets.

3.2.4 Comparison with Third Party Result

As we have not came across unsupervised learning using the skeleton data from RGB-D sensor, we compare our results with the work of the authors of CAD-60 [4] that was based on supervised learning using the same dataset. Sung *et al.* [4] performed supervised learning for human activity detection of the activities in CAD-60 dataset. They achieved 84.7% precision and 83.2% recall in detecting the correct activity when the person was seen before in the training set. This result was an average of twelve (12) activities in CAD-60. They did not detect still action, whereas we have omitted four activities from their dataset due to insufficient examples for the discovery phase. This is explained in Appendix A. Sung *et al.* [4] have used two-layered maximum entropy Markov

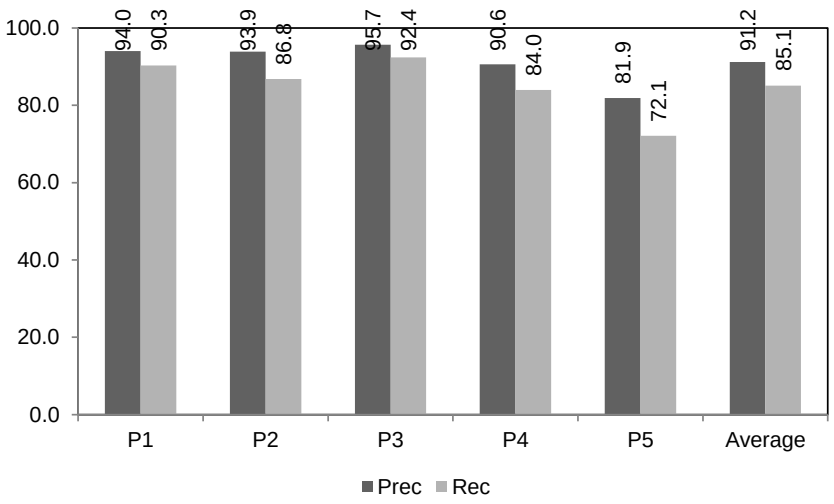


Figure 3.11: Average recognition precision and recall for the overall framework across all actions for the five subjects.

model (MEMM) to learn activity models from feature set combining skeleton data and specially placed Histogram of Oriented Gradient (HOG) computer vision features.

Their experiments involved detecting activities based on locations such as kitchen and living room. By zoning the activities, their algorithms were required to discriminate at most four activities and reject the random activities at any one time. In contrast, our results were obtained through an unsupervised approach from discovery to recognition. Our algorithms were required to discriminate nine actions and reject the random action at any one time in both discovery and recognition phases. We compute the average precision and recall of the results of Sung *et al.* [4] for the nine activities we have used in our experiment, and compare with our results in Table 3.5. The numbers for their results are summarized from their Full Model "Have Seen" results [4]. Our results are on average superior.

3.3 Summary

We have proposed a complete framework to discover, learn and recognize human actions. The framework operates autonomously in all phases. The incremental approach of clustering can discover undefined number of actions. The framework does not require prior specification of the

Table 3.5: A comparison of our results and the results of Sung *et al.* [4].

	Our results		**Sung *et al.***	
	Prec	**Rec**	**Prec**	**Rec**
brushing teeth	98.5	68.8	96.7	77.1
cooking chopping	96.1	70.6	70.3	85.7
cooking stirring	92.6	89.2	74.3	47.3
relaxing on couch	88.8	94.6	86.8	82.7
talking on couch	85.1	89.4	98.8	94.7
talking on phone	93.2	96.3	88.4	91.1
working on computer	94.9	92.5	89.5	93.8
writing on whiteboard	97.6	94.0	85.5	91.9
Average	**93.4**	**86.9**	**86.3**	**83.0**
Std Deviation	**4.2**	**10.2**	**9.2**	**14.6**

number of actions and other parameters for the model learning. It does not require labeled observations. It is designed to reject random movements. The experiment results show that the performance of our completely unsupervised approach can achieve precision and recall superior to that achieved using supervised approach. The framework achieved an overall precision of 91.2%, and recall of 85.1%.

Chapter 4

An Incremental Clustering Approach for Human Action Discovery

Human Action Discovery is the autonomous process to discover action model without user intervention. In this chapter, we will start by describing our feature extraction process and explain the features used for action discovery. For an intelligent system to learn or extract information from a given set of features, the quality of the features is as important as the learning algorithm. The features should ideally contain relevant data suitable for the selected learning algorithm to learn the desired information, while avoiding redundant features that confuse the learning algorithm.. Feature extraction in the context of this dissertation is not about image processing. The raw data are coordinates of fifteen joints in human skeleton. These coordinates have been obtained directly from the OpenNI SDK [30] for the RGB-D sensor, Kinect.

With the available data, we use clustering algorithm to distinguish one action from another. In particular, we use K-means clustering algorithm. One problem with using K-means, and many other clustering algorithms, is the requirement to specify the number of clusters, k, in advance. In our context, we expect the intelligent system to autonomously discover new actions and the number of actions to be found is unknown to the system. Another characteristic of K-means is that it does not have the notion of noise. In K-means, all data points are assigned to a cluster. For action discovery, we expect there are noisy data, i.e. random movements, that should not

be assigned to any cluster. In this chapter, we propose an incremental approach to resolve these problems.

4.1 K-means Clustering

To autonomously discover actions, we use an unsupervised learning algorithm, in particular a clustering algorithm, to group different actions into individual clusters. There are a number of clustering algorithms developed to address different challenges in data mining and machine learning. The choice of the algorithm depends on the nature of the data. In data mining problems, the underlying distribution profiles of the data are often unknown and are not always isotropic [31]. In our application, however, we have clear expectation of the data. In ideal situation, if a person performs an action consistently, we expect the features to have the same values. We expect the variation for each action to be isotropic from the average execution of an action by the person. This allows us to use one of the simplest clustering algorithm, K-means [32].

K-means looks for similarity among the examples in the dataset by using simple distance measurement. Given the required number of clusters, K-means group the data points (observations) in the dataset by minimizing the distance from each data point to a cluster center (centroid). In our work, we have used squared Euclidean distance as the distance measure.

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2 \tag{4.1}$$

where k is the number of clusters, n is the number of data points (observations), $x_i^{(j)}$ is ith data point in Cluster j and c_j is the centroid of Cluster j, i.e., C_j.

Clustering using K-means involves the following steps:

1. Initialize cluster centroids $\mu_1, \mu_2, \cdots, \mu_k$ randomly, where k is the number of clusters.

2. Repeat until convergence {

   (a) Assign every data point $x^{(i)}$ to its nearest centroid μ_j: $c^{(i)} := arg \min_j \left\| x^{(i)} - \mu_j \right\|^2$ where $c^{(i)}$ is the membership assignment of ith data point.

(b) Recompute the centroid from data points in each cluster: $\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)}=j\}}$ where m is the total number of data points. }

Fig. 4.1 illustrates the steps of K-means to cluster a fictitious 2D dataset.

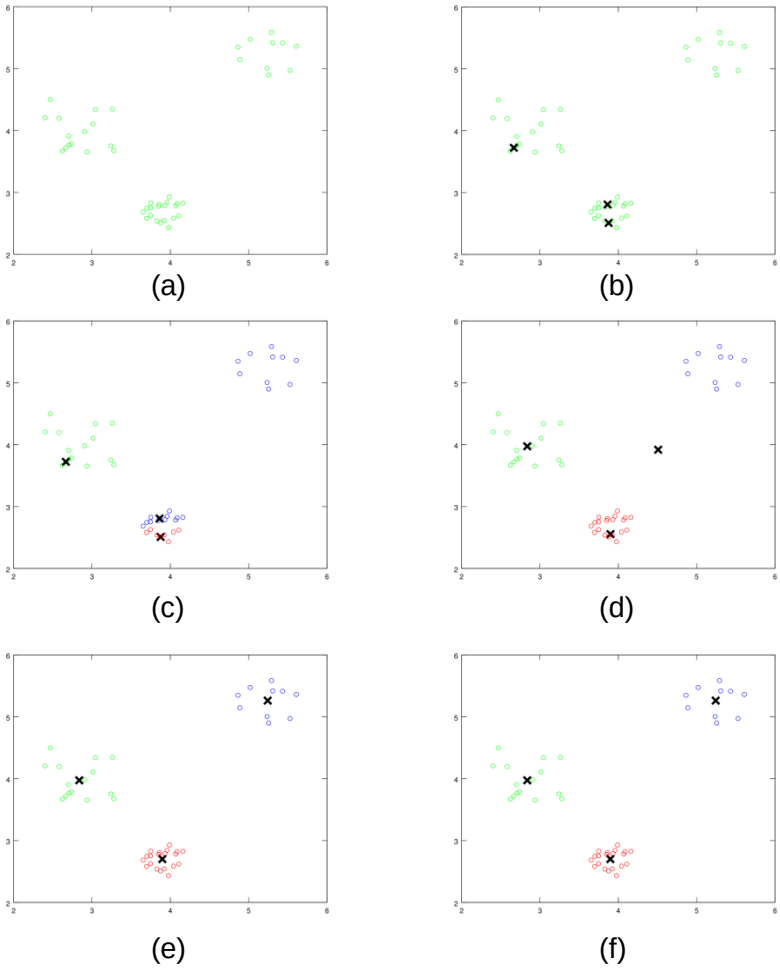


Figure 4.1: Steps in K-means. (a) Original dataset. (b) Random initialization of centroids (crosses). (c) Iteration 1: points assigned to nearest centroids. (d) Iteration 2: compute new centroids, points assigned to nearest centroids. (e) Iteration 3: compute new centroids, points assigned to nearest centroids. (f) Iteration 4: no further move of centroids and data points, algorithm completes.

The outcome of K-means clustering is subjective to the random initialization of the centroids. To account for the problem of random initialization in K-means, ten rounds were run and the outcome with the lowest cost function was taken.

4.2 Feature Extraction

4.2.1 Sampling

There are three typical ways to sample or segment human action or activity observations: manually identify the start and the end of every action, automatic detection of the start and the end of every action and, sliding window. Many research works analyze an entire video clip from a standard dataset, where the start and the end of every action or activity have been manually defined. Manual process is not only laborious but also not feasible in applications in the natural human living environment. Automatic detection is usually achieved by detecting significant changes in the movement. However in everyday life, people spend significant amount of time in stationary state such as sitting and standing, which in our context are considered actions. Further, many actions are not carried out abruptly and such actions will not easily trigger automatic detection. Automatic detection while ideal remains a challenging task. For human activity understanding in normal home setting, it is desirable to have a continuous observation. That makes sliding window a suitable choice. Using fixed width keeps the operation simple. In our work, we have used fixed width window of two seconds to represent an action observation or instances. More details on action segmentation is given in Chapter 6.

4.2.2 3-D Joint Position Coordinates from an RGB-D Sensor

OpenNI SDK [30] can effectively determine 3-D coordinates of fifteen joints in human body, referred as skeleton data as shown in Fig. 4.2, from the depth data of Kinect. Kinect can deliver the depth data at 30 fps. The raw skeleton data from OpenNI has three coordinates (x, y and z) of fifteen joints at 30fps. For two seconds, there are $3 \times 15 \times 30 \times 2 = 2700$ dimensions in total. In Section 4.2.4, we will describe our approach to reduce the dimension of the feature by forming local vectors based on human range of movements and at a lower frame rate.

Figure 4.2: Human skeleton composed from fifteen (15) joints.

4.2.3 Problem with Correlation Based Feature Reduction

The typical ways to reduce dimension of data rely on finding some relationship between the features and the classification of the data. They select features based on the "relevance" of the features to correctly represent the input data. In doing so, the feature set is fit to the input data, i.e. learning set.

In our framework, if we do not have the ground truth of the data given, it is difficult to apply the conventional dimensional reduction techniques. Further more, we would like the features to be generalized for HAR but not fit to the sample data with finite set of actions at any one time. For example, if our training actions do not have legs motion, the feature space would be reduced to omit features related to legs. The framework is designed to discover undefined number of actions. We have avoided applying correlation based dimensional reduction to reduce the dimension of the feature space.

To demonstrate the negative effect of the correlation based dimension reduction for our application, we have used minimum Redundancy Maximum Relevance (mRMR) [33] to select 500 features from the 2700 features. We performed K-means clustering on the learning set of KOS-

H16 dataset (see Appendix A), and followed by cross-validation by assigning members of the test set of KOS-H16 to the centroids obtained from the learning set. Table 4.1 shows the result. Prec and Rec are the precision and recall during the learning phase, i.e. clustering with ground truth. Prec Cross and Rec Cross are the precision and recall during the cross-validation, i.e. assigning unseen data to existing clusters. We observe significantly poor precision and recall during the cross-validation.

Table 4.1: Precision and recall during clustering and cross-validation on KOS-H16 dataset, with 500 features selected by mRMR.

	Prec	Prec Cross	Rec	Rec Cross
bowing	52.0	0.0	61.3	0.0
drinking (left)	100	0.0	98.0	0.0
drinking (right)	36.0	30.2	100	55.6
sit	63.9	66.7	62.7	4.4
sit down	84.2	2.0	84.0	1.1
stand	84.1	19.4	94.0	71.1
stand up	79.6	52.5	96.0	60.0
talking on phone (left)	97.7	0.0	54.0	0.0
talking on phone (right)	33.3	0.0	33.3	0.0
walking	94.0	3.0	82.7	3.3
wave bye (left)	100	0.0	88.7	0.0
wave bye (right)	0.0	0.0	0.0	0.0
wave come (left)	98.0	68.1	78.0	38.9
wave come (right)	80.1	45.1	96.7	64.4
wave go (left)	100	16.0	92.7	18.9
wave go (right)	33.3	9.9	18.7	18.9
Average:	**71.0**	**19.5**	**71.3**	**21.0**

4.2.4 Human Range of Movements

While it is difficult to model human activities and actions due to its wide variety and complexity, human movements are constrained by the range of movement (ROM) [34]. Studies in kinematics of human motion [35] have identified possible movements around human joints including flexion, extension, lateral flexion, rotation of spinal column (the body movements); flexion, extension, abduction, adduction of shoulder joint (the arm movements); flexion, extension of elbow joint (the forearm movements); flexion, extension of knee joint (the leg movements); flexion, extension,

adduction of hip joints (the thigh movements). Fig. 4.3 gives some illustrations of human range of movement.

In our work, these angular movements have been used as features for human action detection.

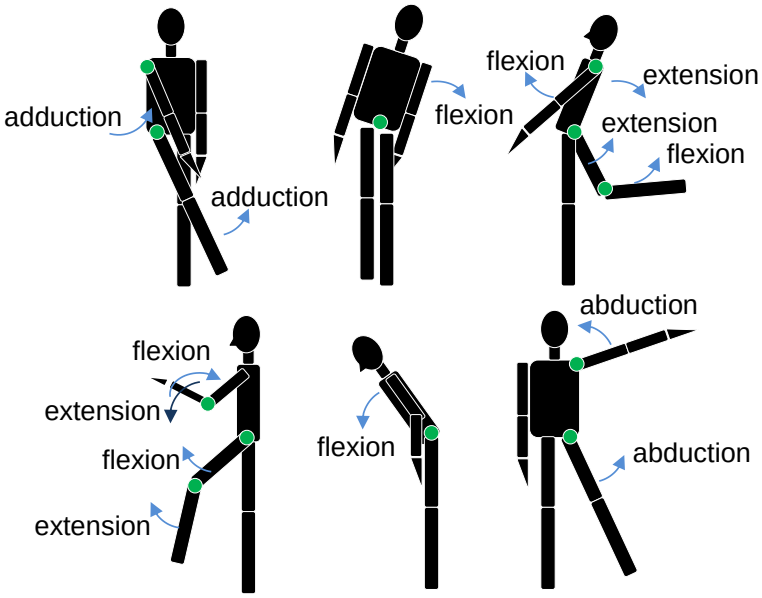


Figure 4.3: Illustrations of range of movement.

For each pose, i.e., in each frame, the following features are extracted from the coordinates of the joint positions: four vectors describing body flexion and turn; four vectors describing arms abduction and flexion; and four vectors describing leg abduction and flexion. Fig. 4.4 illustrates a vector to represent right hand flexion. $\overrightarrow{rh}$ and $\overrightarrow{rs}$ are the vectors that represent the joint coordinates of the right hand and the right shoulder, respectively, in camera coordinate frame. In total we have 42 features per frame. The details are given in Chapter 6.

By taking local vectors and perform coordinate frame transformation, the features are view-invariant. While it is not important to be scale-invariant when working on a single person, we scale the vectors by the shoulder width to maintain scale-invariant. This will become useful when we want to share the learned action models between different systems. In addition to vectors representing the range of movement, we added two vectors to represent the interaction of each hand with head. We note our hands interact most with the head and some actions can be difficult to discriminate, e.g., drinking and talking on phone.

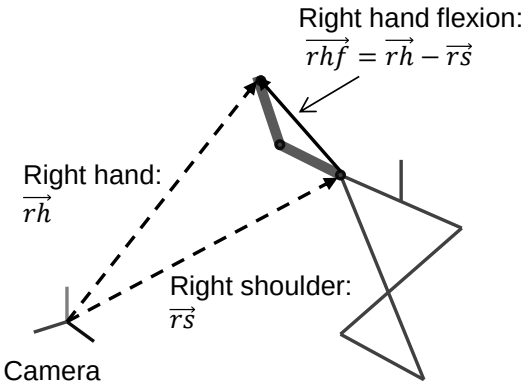


Figure 4.4: A example of vector to represent right hand flexion in ROM.

4.2.5 Investigation of Features for Human Action Recognition

Given that each action observation is sampled for a window of two seconds, at 30fps, each action observation contains 60 frames at full resolution. In this section, we test the effectiveness of our feature set in clustering and the effect on clustering outcome when the frames per observation are reduced to 30, 15 and 6.

Note that Schindler and Gool [36] have showed that 5 to 7 frames are sufficient to recognize basic actions, and Sung *et al.* [4] have used fixed window of three seconds on the same dataset that we are using in this work.

4.2.5.1 Experiment on Feature Extraction

We have conducted two experiments to determine the effectiveness of the feature set. We first perform K-means clustering (see Section 4.1) at different frames per observations to determine a suitable reduce frame rate. Using the frame rate determined, we evaluate the clustering performance in detail. We have used the two datasets (learning sets) described in Appendix A, without the random movements. All actions from the dataset of each subject were pooled together, and K-means was ran on the dataset of each subject. The results presented here are the average of five runs of the experiment.

4.2.5.2 Result and Discussion on Feature Extraction

Fig. 4.5 gives the average precision and recall score of all nine actions from the four subjects (Person 1 to 4) of CAD-60 dataset at different frames per observation. It can be seen that reducing the frames from 60 to 30 has no effect on the performance of clustering while reducing to fifteen frames only lowered the precision and recall by not more than 1%. Further reduction to 6 frames only lowered the clustering performance by further 1%. This however is anticipated as majority of the actions in the dataset do not involve much motion. For actions with significant motion, e.g., waving hand, we expect the clustering performance to degrade significantly at low frame rates.

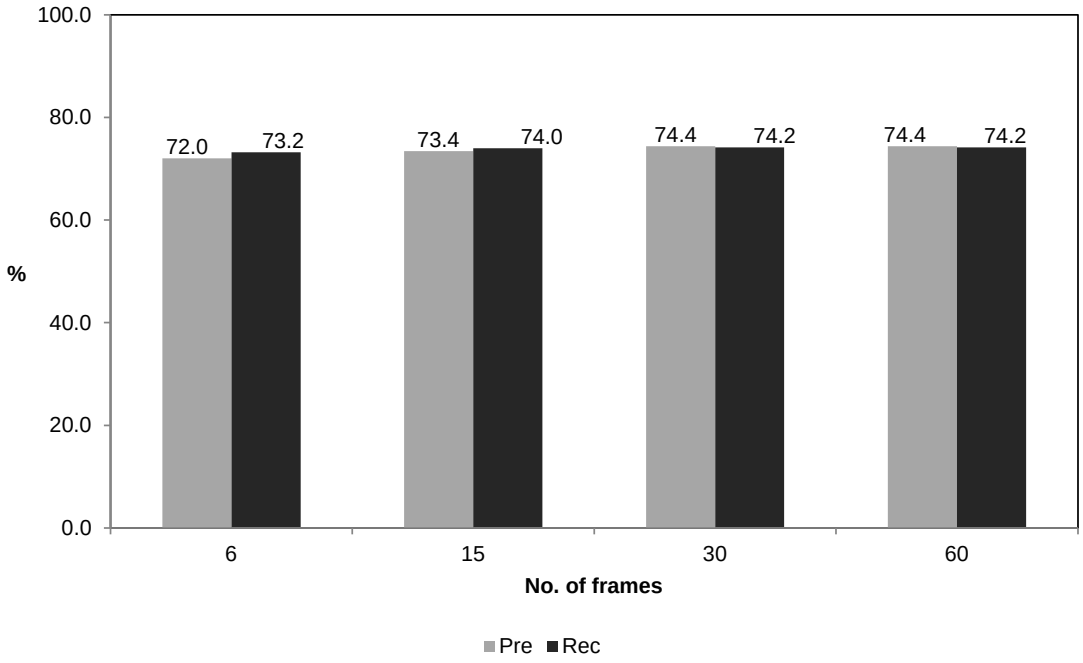


Figure 4.5: Average precision and recall of all nine actions for four subjects (Person 1 to 4) at different frames per observation.

Fig. 4.6 shows the clustering performance on our dataset, KOS-H16, at different frames per observation. This dataset has actions with significant movements. It is surprising to see that at six frames per observation, the clustering result was significantly better than the results at higher frame rates. The inconsistency between the results for the two datasets indicates that at six frames per observation, the clustering outcome is highly dependent on the nature of the data. We can

however observe that the clustering performances at 15, 30 and 60 frames per observation were consistent in both datasets. The results on both datasets indicate that at 15 frames per observation, the clustering result was close to that at full resolution of the sensor, i.e., 60 frames per observation. The result suggests that 15 frames per observation is a suitable reduced frames per observation without compromising the clustering performance.

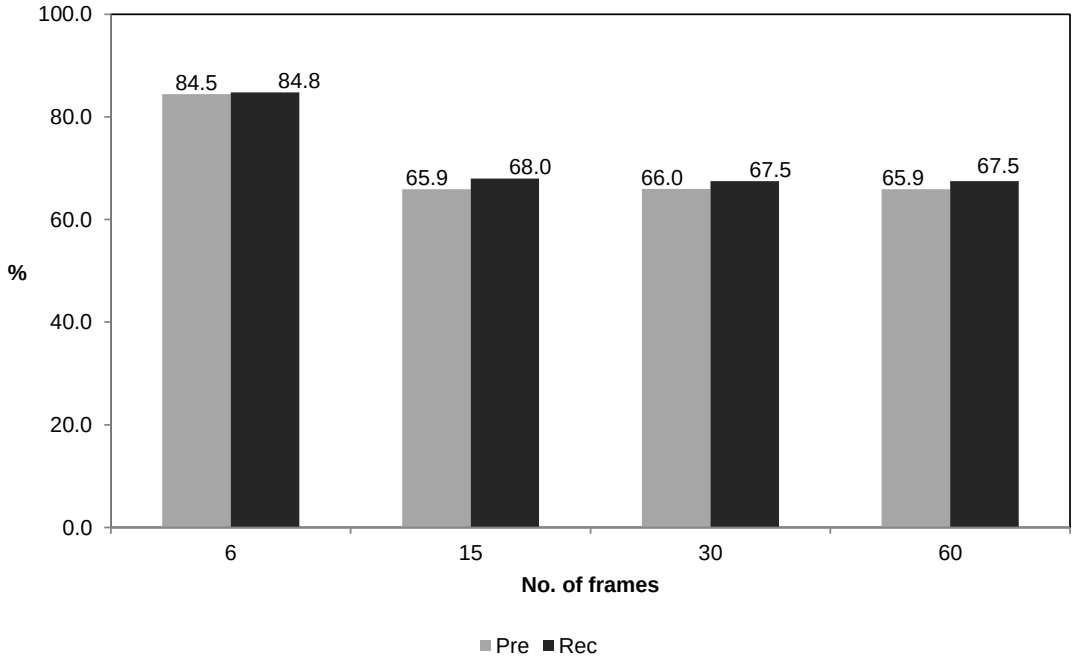


Figure 4.6: Average precision and recall of all sixteen actions for single subject (Person 5) at different frames per observation.

From here on, we discuss the results at 15 frames per observation. Fig. 4.7 shows the average precision and recall of the clustering of all the actions for each of the five subjects. The average clustering performance across all subjects and all actions has the precision of 80.4% and recall of 83.8%. The dataset of Person 5 has the lowest precision and recall at 68.6% and 75.6% respectively. This is due to the fact that there are significantly more actions to discriminate in this dataset and that there are significantly more movements in the actions.

Table 4.2 and Table 4.3 show the detailed clustering result for each action for each subject. The results in Fig. 4.7 correspond to the bottom rows in Table 4.2 and Table 4.3. From these tables, we see the clustering performance was good in most of the actions, with many achieving

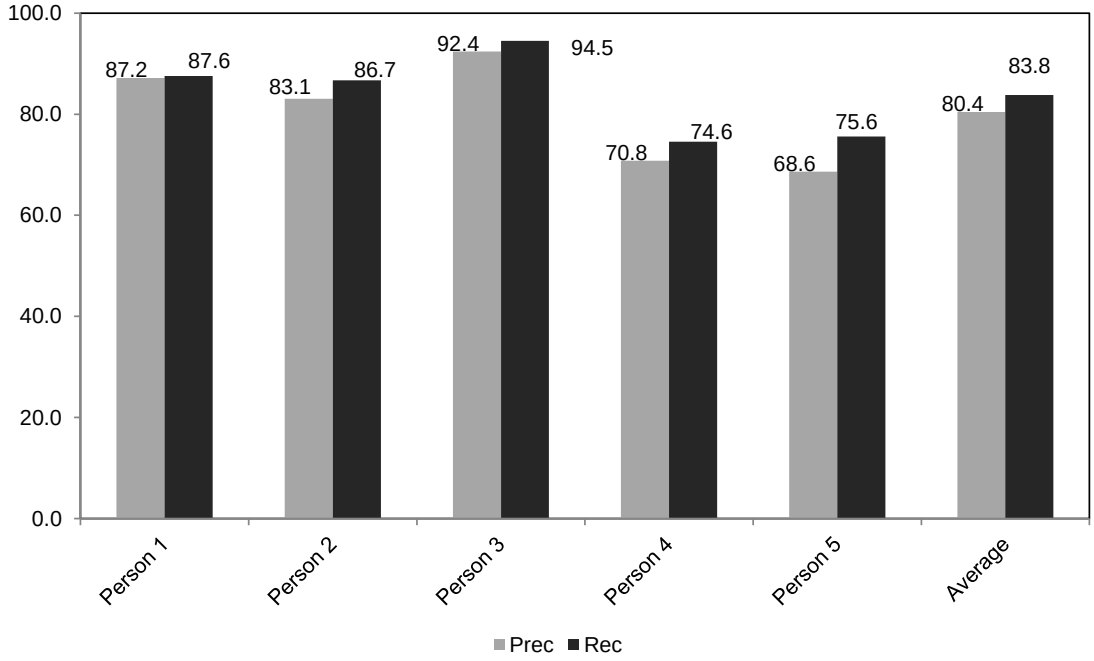


Figure 4.7: Average precision and recall of the clustering of all actions for all five subjects at 15 frames per observation.

100% precision and recall.

Table 4.2: Precision and Recall of the Clustering Result for Person 1 to 4.

	Person 1		**Person 2**		**Person 3**		**Person 4**		**Average**	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
brushing teeth	46.1	77.5	67.8	99.3	89.0	93.9	66.5	63.9	67.3	83.7
cooking (chopping)	98.5	90.0	90.4	94.3	90.0	98.2	77.9	87.9	89.2	92.6
cooking (stirring)	90.9	99.3	84.3	72.5	79.0	80.0	48.9	64.6	75.7	79.1
relaxing on couch	100	100	100	100	100	100	100	100	100	100
still (standing)	100	98.6	100	100	98.6	100	53.8	80.0	88.1	94.6
talking on couch	90.0	100	100	100	100	100	100	77.9	97.5	94.5
talking on the phone	79.2	42.9	38.3	40.0	75.3	78.6	19.9	40.0	53.2	50.4
working on computer	80.0	80.0	97.8	100	100	100	90.2	100	92.0	95.0
writing on whiteboard	100	100	69.2	74.3	100	100	80.0	56.8	87.3	82.8
Average:	**87.2**	**87.6**	**83.1**	**86.7**	**92.4**	**94.5**	**70.8**	**74.6**	**83.4**	**85.8**

K-means clustering outcome is sensitive to the randomness in its initialization. However, the effect is observed only with actions that are very similar. For example, significant confusion was observed between Action 1 (brushing teeth) and 7 (talking on the phone) as they are very similar

Table 4.3: Precision and Recall of the Clustering Result for Person 5.

	Prec	Rec
bowing	81.6	98.6
drinking (left)	56.4	85.0
drinking (right)	11.4	14.3
sit	79.1	98.6
sit down	98.4	85.4
stand	76.2	76.1
stand up	58.6	57.9
talking on phone (left)	33.0	22.5
talking on phone (right)	55.9	87.9
walking	83.6	92.9
wave bye (left)	64.9	77.5
wave bye (right)	84.8	93.9
wave come (left)	69.5	74.6
wave come (right)	78.9	99.3
wave go (left)	87.5	79.3
wave go (right)	78.3	65.7
Average:	**68.6**	**75.6**

as illustrated in Fig. 4.8. Action 2 (chopping) and 3 (stirring) are also very similar as illustrated in Fig. 4.9. In certain runs of K-means, Action 5 (standing) was confused with a few standing actions such as Action 1 (brushing teeth) and 7 (talking on the phone).

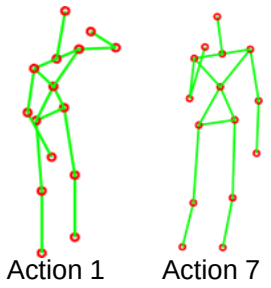


Figure 4.8: Skeleton of Action 1 (brushing teeth) and Action 7 (talking on the phone). They are very similar.

The consequence is that we observe low value of precision and recall in these actions in Table 4.2. In the case of the dataset for Person 5, there are more actions that are similar to each other. In this dataset, Action 2 (drinking with left hand) and 8 (talking on phone with left hand); Action 3 (drinking with right hand) and 9 (talking on phone with right hand), are very similar.

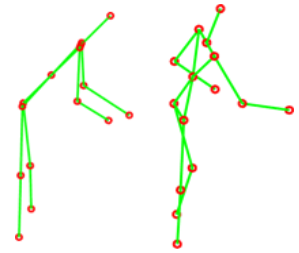


Figure 4.9: Skeleton of Action 2 (chopping) and Action 3 (stirring). They are very similar.

The results in Table 4.2 and 4.3 were compiled from five runs of the clustering. To avoid clutter, we show only two confusion matrices out of the five runs to illustrate the state of confusion. The rows are the actual actions and the columns are the clusters obtained. In Fig. 4.10, we see that Action 2 (drinking with left hand) and 8 (talking on phone with left hand) were confused with each other, and Action 3 (drinking with right hand) was confused with Action 9 (talking on phone with right hand). In this run, Action 1 (bowing) was significantly confused with Action 6 (stand). However, in another run, Action 1 and 6 were not confused as shown in the confusion matrix in Fig. 4.11. In this run, Action 8 (talking on phone with left hand) and 11 (wave bye with left hand) were confused instead. The effect of the random initialization of K-means is observed. However, for most actions, the effect was not significant.

4.3 Number of Clusters in Unsupervised Learning

In unsupervised learning, one of the most challenging problems is to know how many valid classes or clusters are available in the dataset. Given a dataset as shown in Fig. 4.12(a), specifying different k-value will result in different clusters being formed. Without knowledge of the data, it is difficult to know the correct value of k. Further, there may be noise in the data as shown in Fig. 4.12(d).

For the algorithm to be scalable to undefined number of action, it needs to perform clustering without prior specification of the k-value. The algorithm should also have the notion of noise in order to reject random movements.

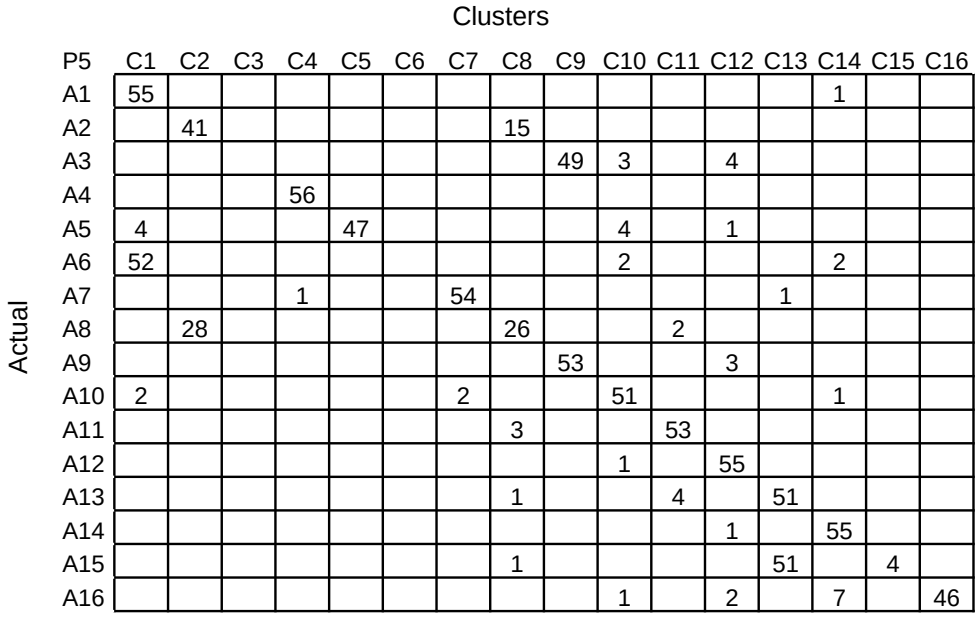


Figure 4.10: Confusion matrix for clustering result for Person 5 in one of the five runs.

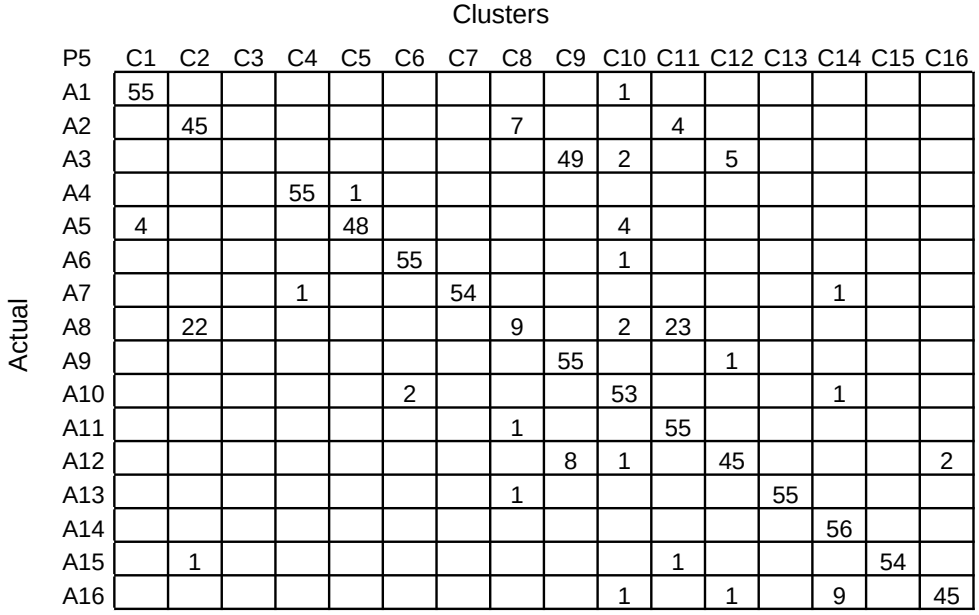


Figure 4.11: Confusion matrix for clustering result for Person 5 in another one of the five runs.

The conventional way to estimate the number of clusters, or k-value, is to perform the clustering for the range of $k = 2$ to n and evaluate the clustering results at each k-value, where n is the number of data points in the dataset. The k-value that gives the "best clustering outcome" based

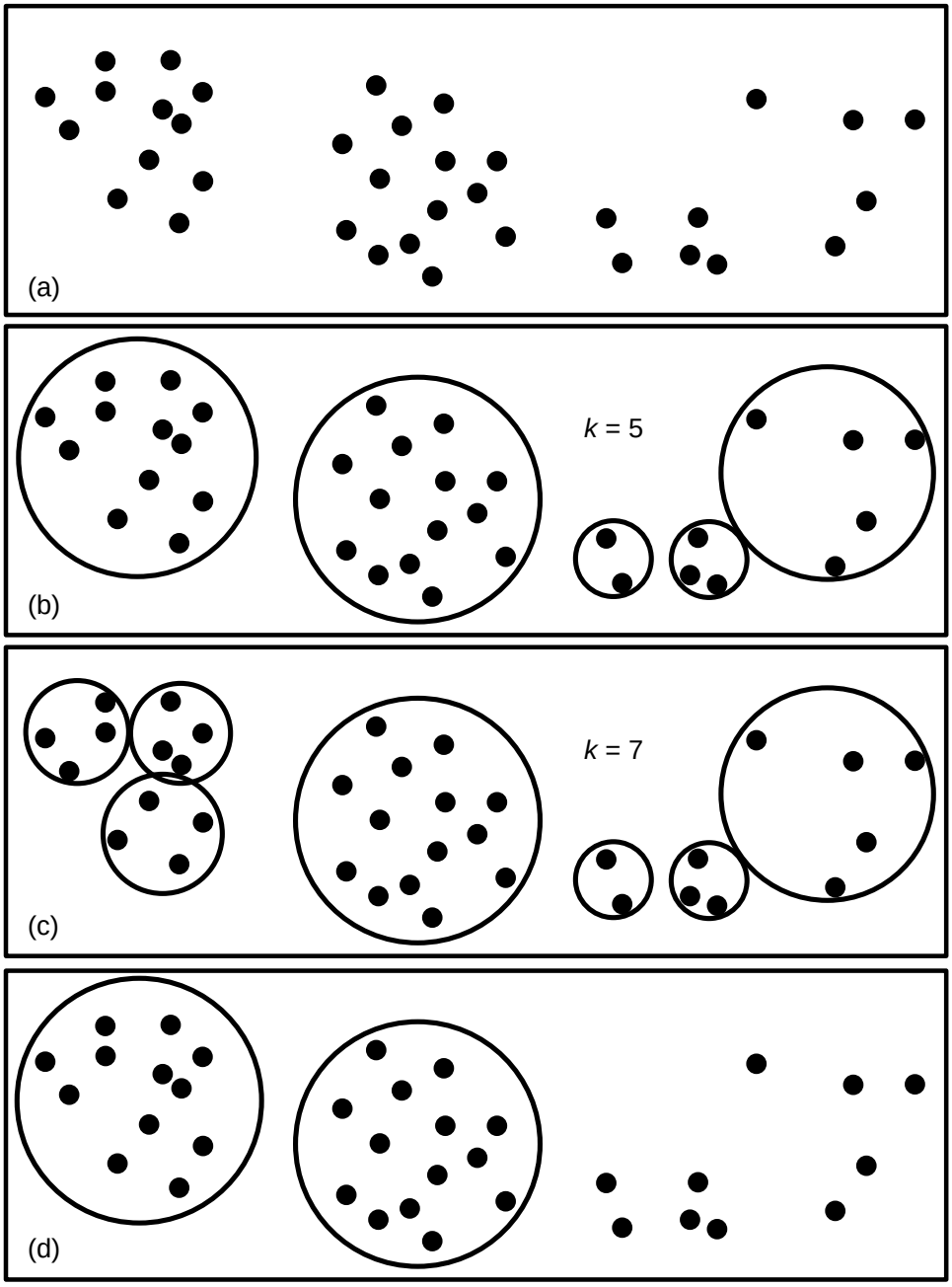


Figure 4.12: Uncertainty in number of clusters.

on certain criteria is taken as the optimal k-value.

There have been various criterion [37] used to evaluate the clustering outcome and to select the "best clustering outcome". They are generally referred as cluster validity indices. We have investigated the use of cluster validity indices for human action discovery. We evaluated the effectiveness

of five indices [38, 39, 40, 41, 42] as given below.

Silhouette (Sil) [[42]]. For each data point, $a(x_i)$ is the average distance from the point to other points in its cluster, and $b(x_i)$ is the average distance from it to all points in nearest cluster. The objective is to maximize $Sil(k)$.

$$per\ point\ Sil(x_i) = \frac{b(x_i) - a(x_i)}{max\{b(x_i), a(x_i)\}} \tag{4.2}$$

$$per\ cluster\ Sil(C_j) = \frac{1}{|C_j|} \sum_{x_i \in C_j} Sil\left(x_i^{(j)}\right) \tag{4.3}$$

$$overall\ Sil(k) = \frac{1}{k} \sum_{j=1}^{k} Sil\left(C_j\right) \tag{4.4}$$

Davies-Bouldin (DB) index [[39]]. The objective is to minimize the $DB(k)$ index.

$$DB\left(k\right) = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \left\{ \frac{s_i + s_j}{d_{ij}} \right\} \tag{4.5}$$

$$s_j = \frac{1}{n_j} \sum_{x_i \in C_j} \left\| x_i^{(j)} - c_j \right\|, d_{ij} = \left\| c_i - c_j \right\| \tag{4.6}$$

Calinski-Harabasz (CH) index [[38]]. The objective is to maximize $CH(k)$.

$$CH\left(k\right) = \frac{trace(SSW(k))/k-1}{trace(SSB(k))/n-k} \tag{4.7}$$

where $SSW(k)$ is the (sum-of-square) within-cluster scatter matrix and, $SSB(k)$ is the (sum-of-square) between-cluster scatter matrix as given below:

$$SSW\left(k\right) = \sum_{j=1}^{k} \sum_{x_i \in C_j} \left(x_i^{(j)} - c_j\right)^T \left(x_i^{(j)} - c_j\right) \tag{4.8}$$

$$SSB\left(k\right) = \sum_{j=1}^{k} \left(c_j - \mu\right)^T \left(c_j - \mu\right) \tag{4.9}$$

where μ is the mean of the whole dataset.

Krzanowski-Lai (KL) index [[41]]. The objective is to maximize $KL(k)$.

$$KL(k) = \left| \frac{DIFF(k)}{DIFF(k+1)} \right| \tag{4.10}$$

$$DIFF(k) = (k-1)^{2/p} trace(SSW(k-1)) - k^{2/p} trace(SSW(k)) \tag{4.11}$$

where p is the dimension (number of variables/features) of the data point; $p = 630$ in our case, see Section 4.

Hartigan (Ha) index [[40]]. The objective is to add cluster until Ha is below a threshold. $Ha \leq 10$ is typically used and we have used this value in our work reported in this dissertation.

$$Ha(k) = \left(\frac{trace(SSW(k))}{trace(SSW(k+1))} - 1 \right) (n-k-1) \tag{4.12}$$

where n is the number of data points in whole data set.

Table 4.4 shows the result of estimating the number of clusters (value of k_s) for K-means using the five cluster validity indices for the the learning set of CAD-60 dataset (see Appendix A). For the purpose of comparison, we have computed the overall error for each index as given in Eq. 4.13. The index with lowest error is considered as giving the best value of k_s.

$$Overall\ error\ e_t = e_{(t(1))} + e_{(t(2))} \tag{4.13}$$

where

$$e_{(t(1))} = \Big(\sum_{(P \in P1, P2, P3, P4)} (k_s^p - k_e^p)^2 \Big)^{(12)} \tag{4.14}$$

$$e_{(t(2))} = \Big(\sum_{(P \in P1R, P2R, PR3, P4R)} (k_s^p - k_e^p)^2 \Big)^{(12)} \tag{4.15}$$

where k_s^p is the k suggested, for the dataset of the subject p, k_e^p is the expected k for the dataset

of the subject p, $e_{(t(1))}$ is the total error for all datasets without random actions, $e_{(t(2))}$ is the total error for all datasets with random actions. P1, P2, P3 and P4 are the datasets of the subjects P1, P2, P3 and P4 without the random movements. P1R, P2R, P3R and P4R are the datasets of the subjects P1, P2, P3 and P4 with the random movements.

For datasets without random actions, k_e^p is 9. For datasets with random actions, we expect k_e^p to be more than 9, however we did not know the exact number. For the purpose of comparison, we have used the average value of all k_s above 9 for datasets with random actions. The value is 12.9. While Sil, CH and DB did well on datasets without random actions, they performed poorly on datasets with random actions that have high variances. The result suggested that Hartigan index was the best choice among the five indices. The approach using a cluster validity index, however, do not reject random movements or noise. It attempts to estimate an optimal number of clusters and all data points are assigned to a cluster. However, in noisy data, the noise ideally should not belong to any cluster.

Table 4.4: Estimated number of clusters, k_s, for K-means using DB, CH, KL, Ha and Sil indices for CAD-60 dataset. P1 to P4 are datasets for the four subjects without random actions. P1R and P4R are datasets for the four subjects with random actions.

	DB	**CH**	**KL**	**Ha**	**Sil**
P1	6	8	8	8	6
P2	6	6	4	14	6
P3	6	10	20	10	9
P4	9	9	20	12	9
et(1)	5.2	3.3	16.4	6	4.2
P1R	5	3	17	11	5
P2R	17	3	11	11	4
P3R	10	3	16	12	10
P4R	3	3	17	10	3
et(2)	13.7	20	6.7	4.2	15.9
et	18.9	23.3	23.1	10.2	20.1

4.4 The Incremental Clustering Algorithm

While K-means can cluster human actions represented with our features set, it suffers from two major problems: it is sensitive to initialization and it requires the number of clusters, k, be specified a priori. For the proposed framework to be fully autonomous, it has to be able to determine by itself potential number of actions in the sample pool. To account for the two problems, we have proposed an incremental action discovery algorithm

Conceptually, in a real life situation, the algorithm is executed at regular intervals of time after a set of sufficient observations has been added into a sample pool; say, at the end of each day. The system do not have information with regards to the potential number of actions in the sample pool. The algorithm does not attempt to discover all actions within the sample pool at once. It makes its best effort to discover the most likely action. This way it will discard random actions as well as some valid actions. By most likely action, the algorithm finds the most compact cluster using the homogeneity measure defined in the following subsection..

In this section, we describe our proposed incremental approach to discover human actions from unlabeled observations in a noisy dataset. This approach is motivated by the way children learn about their environment in an incremental manner. For example, while there are many activities going on, children do not learn all of them at once. They learn them over time.

Fig. 4.13 gives the algorithm of our proposed incremental approach to discover clusters of actions. It takes the dataset X and a minimum point per cluster $MinPt$ parameter as input. It returns a set of clusters that the algorithm has discovered. Not all data points in the dataset will be clustered.

The basic idea is to start with a sufficiently high value of k, k_{max}. k_{max} can be up to n, the number of data points in the dataset. However, it helps to restrict the computation time by setting a reasonable value for k_{max}. There is no concrete guideline for the choice of k_{max}, however many researchers had referred to Mardia et al. [43] as stating the rule of thumb for setting $k = \sqrt{\frac{n}{2}}$. We have chosen $k_{max} = \sqrt{n}$, which includes the value of k suggested by the said rule of thumb.

For each value of k, K-means is run for a few rounds and the clustering result with lowest total intra-cluster distance is taken as that value of k. This minimizes the problem of sensitivity to

ALGORITHM: Incremental discovery of clusters with *MinPt*

INPUT: A set of observations $\mathcal{X} = \{x_1, \dots, x_n\}$
Minimum points per cluster *MinPt*

OUTPUT: A set of clusters $\mathcal{O} = \{C_1, \dots, C_{NrCt}\}$

ALGORITHM:

1: Set $k = k_{max} = \sqrt{n}$; $\mathcal{O} = \{\ \}$; $i = 1$; Set *MinPt*;

2: Let $\mathcal{X}^* = \mathcal{X}$;

3: **While** $k \geq k_{min}$ **do**

3.1: Perform clustering on $\mathcal{X}^*$ with k, giving clusters
$$\mathcal{O}^* = \{C_1^*, \dots, C_k^*\};$$

3.2: Evaluate the homogeneity of each cluster $\boldsymbol{C_j^*}$ for $j = 1, \dots, k$ only if $\left|C_j^*\right| \geq MinPt$;

3.3: If there is no cluster where $\left|C_j^*\right| \geq MinPt$,
$k = k - 1$;
Skip to Step 2.7;

3.4: Collect the most homogeneous cluster as C_i
$$\mathcal{O} = \{\mathcal{O}, C_i\}$$

3.5: Prune dataset $\mathcal{X}^* = \mathcal{X}^* - C_i$;

3.6: Compute next *k*, $k = \sqrt{n^*}$ where *n** is the number of observations in pruned$\mathcal{X}^*$;

3.7: Increment *i*;

4: Collect un-clustered observations, $\mathcal{O} = \{\mathcal{O}, \mathcal{X}^*\}$

Figure 4.13: The algorithm to incrementally discover actions with *MinPt* parameter.

initialization. For each *k* value, clustering is performed on the dataset and the homogeneity of each of the clusters is evaluated. The choice of clustering algorithm and cluster homogeneity measure will be described in the following subsections. The most homogeneous cluster is collected and the members removed from the dataset. A new value of *k* is then computed from the new population $n*$ of the pruned dataset. Note that it is also possible to add new data to the pruned dataset. This makes the algorithm feasible with dynamic data.

The process gradually lower the value of *k* until $k = 2$ or a specified value, K_{min}. To ensure the clusters have sufficient observations to model the actions, a minimum points or minimum membership parameter (*MinPt*) can be imposed. We suggest to set the *MinPt* parameter based on the size of the dataset as given below.

$$MinPt = \frac{n}{k_{max}} \tag{4.16}$$

where n is the number of data points in the dataset and $k_{max} = \sqrt{n}$.

4.5 Homogeneity of a Cluster

There are a number of measures [37] proposed to evaluate the compactness or validity of a cluster. One key ingredient in many of these cluster validity indexes is the variance. To assess the homogeneity, i.e., cohesiveness and compactness, of individual cluster and rank them accordingly, we have defined and tested two measures: the intra-cluster mean variance ($\bar{\sigma}^2$) and mean joint probability density function ($\overline{P}_f$).

Low value of variance $\bar{\sigma}^2$ indicates compactness of the cluster.

$$mean\,variance\,\overline{\sigma}^2(C_j) \quad = \quad mean\,(var\,(C_j)) \tag{4.17}$$

$$var\,(C_j) \quad = \quad \frac{1}{n_j - 1} \sum_{x_i \in C_j} \left({x_i}^{(j)} - \bar{x}^{(j)} \right)^{\wedge 2} \tag{4.18}$$

$$centroid\,\bar{x}^{(j)} \quad = \quad mean\,(x_i \in C_j) \tag{4.19}$$

where $x_i^{(j)} = [x_{i1} \cdots x_{ip}]$ is a data point in Cluster C_j with dimension p, n_j is the number of points in Cluster C_j, $\bar{x}^{(j)}$ (dimension p) is the mean of all points in Cluster C_j, $\wedge 2$ is element-wise square.

The joint probability density function assumes that observations of a non random action should be normally distributed within its cluster around the cluster centroid with the standard deviation of the cluster. High value of mean joint probability density function $\overline{P}_f$ indicates good cohesion of the cluster based on the assumption of normal distribution. Logarithm in Eq. (4.21) is used to compress the range of the values.

$$\textit{mean joint probability} \tag{4.20}$$

$$\overline{P}_f(C_j) \quad = mean(P(C_j))$$

$$P(C_j) \quad = \quad \sum_{f=1}^{p} ln\left(pdf\left(x_{if}^{(j)}\right)\right) \tag{4.21}$$

$$pdf\left(x_{if}^{(j)}\right) \quad = \quad \frac{1}{\sigma_f^{(j)}\sqrt{2\pi}} e^{-\frac{\left(x_{if}^{(j)}-\mu_f^{(j)}\right)^2}{\left(2\sigma_f^{(j)}\right)^2}} \tag{4.22}$$

where $x_{if}^{(j)}$ is the fth dimension of $x_i^{(j)}$, $\mu_f^{(j)}$ is the mean value of the fth dimension of all points in Cluster C_j, $\sigma_f^{(j)}$ is the standard deviation value of the fth dimension of all points in Cluster C_j.

The main difference between the two criteria is that the variance evaluates the distance of the data points to the centroid, while the probability density function evaluates if the data points are distributed in the form of normal distribution.

In both cases, the "mean" is taken across the dimensions of the data. Since the data points are multidimensional, the mean across all dimensions gives a single value measure.

4.6 Advantages of the Incremental Approach

In a non incremental approach, the clusters are obtained from a single run of the clustering algorithm, once the k-value was found by the cluster validity index. While we can rank the clusters, we maintain the same set of clusters. On the other hand, with the incremental approach, the clusters are obtained from the "best" cluster of each run of the clustering algorithm. The discovered cluster is trimmed from the dataset after each run and the distribution of the data is perturbed. This trimming can potentially lead to the discovery of clusters not found in the large dataset. The

objective of the approach is not to estimate an optimal number of clusters, but instead it attempts to find “good” clusters from the data. This is useful when we have noisy data.

The trimming in each iteration also reduces the overall computation time when compared to the non incremental approach. In the non incremental approach using cluster validity index, the algorithm search through the range of k-value, from $k = 2$ to $k = k_{max}$, with the same number of data points, n, at all k-values. On the other hand, the incremental approach keeps reducing the number of data points, n, in each iteration. It also lowers the k-value based on the reduced number of data points. It does not necessary go through all values of k within the range of $k = 2$ to $k = k_{max}$.

The algorithm starts from the highest k-value allowing it to find a cluster right from the first iteration. The search starts from a high k value, k_{max}. The advantage of starting from high k value is that it is more likely to have clusters with single action than using a small value of k. For example, if there are ten actions in the sample pool, if we start with $k = 2$, it is highly likely that both clusters contain a few actions in them. No decision can be made at this point until further decomposition of the clusters has been performed. If we start with $k = 20$, for example, it is highly likely that there are a number of the clusters that contain single action in them. By starting from high k value, each cluster is being identified and trimmed from the sample pool at each iteration of k value.

Fig. 4.14 shows the comparison between the non-incremental approach using cluster validity index, e.g. Hartigan Index, and our incremental approach. The figure shows the outcome of each iteration of the algorithm. In each iteration, the left hand side shows the outcome from the non-incremental approach, i.e. using cluster validity index, while the right hand side is the outcome from the incremental approach.

A factitious dataset with 53 data points was used. There are apparently three clusters surrounded with noise. Given $n = 53$, k_{max} is approximately 7. The incremental approach started with highest $k = k_{max}$, while the non-incremental approach started with lowest $k = 2$. The non-incremental approach iterated through all values of k from 2 to 7 and computed the cluster validity index in each iteration or at the end of all iterations. An “optimal" k-value is determined based on the cluster validity index value and the resulting clusters from the result with the “optimal" k-value

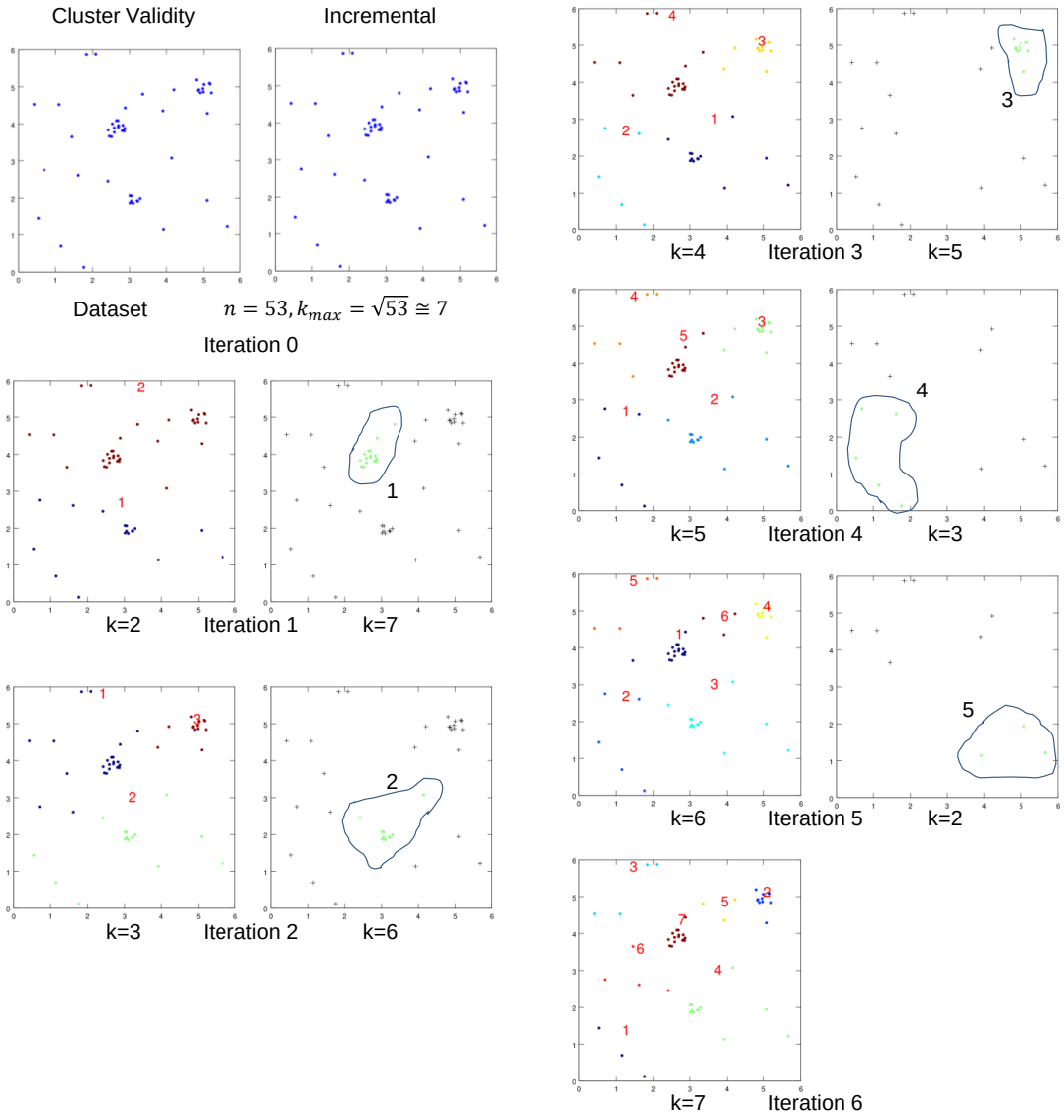


Figure 4.14: Comparison between the incremental approach and non-incremental approach of clustering (cluster validity).

are taken as the outcome. All data points are assigned to a cluster, i.e. there is no noise rejection. In the test run, this approach took six iterations to complete. In contrast, the incremental approach found one cluster in each iteration and complete the clustering in five iterations. In each iteration, the dataset reduced in size.

4.7 Human Action Discovery Performance

We have evaluated the performance of our proposed incremental clustering approach on the datasets described in Appendix A. We also experimented on the value of *MinPt* parameter, and identify the suitable homogeneity measure among the two given in Section 4.5.

4.7.1 Experiment

In our experiment, we let the algorithms in the framework exhaust all samples in each dataset. The datasets have finite samples. In real life application, the algorithms in the framework do not have to discover all actions at once. In our experiment, we have used minimum points per cluster $MinPt = 25$.

We carried out five experiments to evaluate the action discovery on the dataset of each subject. One experiment used the non-incremental approach based on Hartigan index, while the other four experiments investigate the different homogeneity measures and *MinPt* values using the incremental approach. We the experiments as below:

- H: clustering using Hartigan index,
- Var: clustering using the proposed incremental approach using mean variance as the homogeneity measure without setting the minimum point parameter, i.e. $MinPt = 0$,
- Pdf: clustering using the proposed incremental approach using mean joint probability as the homogeneity measure without setting the minimum point parameter, i.e. $MinPt = 0$,
- Var-25: clustering using the proposed incremental approach using mean variance as the homogeneity measure with the minimum point parameter set to 25, i.e. $MinPt = 25$,
- Pdf-25: clustering using the proposed incremental approach using mean joint probability as the homogeneity measure with the minimum point parameter set to 25, i.e. $MinPt = 25$.

The algorithm was given all the data points, all actions mixed up, without label from the dataset of each subject. In each experiment, we evaluated how well the clustering discovered the actions

and reject the random movements (RA) in each dataset (Person 1 to Person 5). For each clustering result, we calculated the precision and recall. Given that the result of K-means is sensitive to the random initialization, we performed five runs of each experiment to obtain an average evaluation of the clustering performance.

4.7.2 Result and Discussion

Fig. 4.15 shows the average precision of the five experiments, H, Var, Pdf, Var-25 and Pdf-25 as described in Section 4.7.1, on the dataset of each subject. The right most set of columns are the overall average across five subjects. For each subject, the values given are the average across all actions, i.e. nine actions for Person 1 to Person 4, and sixteen actions for Person 5.

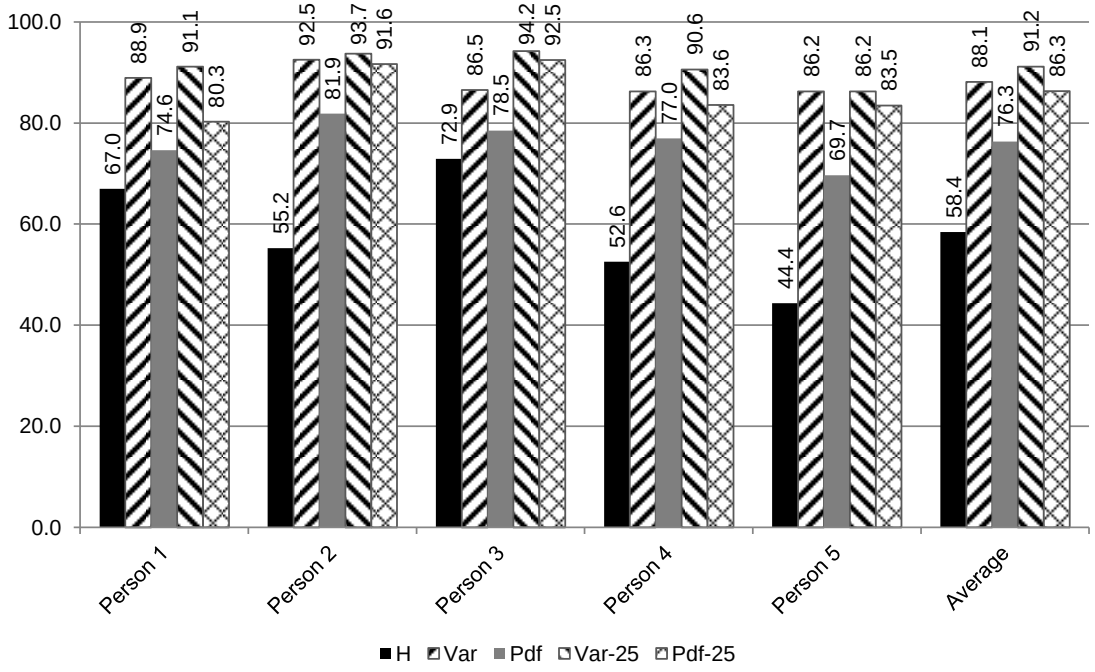


Figure 4.15: Average precision of different approaches for the data of each subject.

We observe a few things from Fig. 4.15:

1. For all subjects, the precision achieved using Hartigan index (H), i.e. the non incremental approach, is the lowest and it is significantly lower than the precision achieved by the incremental approaches (Var, Pdf, Var-25 and Pdf-25).

2. For all subjects, the highest precision is that achieved by the incremental approach using mean variance as the cluster homogeneity measure and with minimum point parameter $MinPt = 25$ (Var-25).

3. For all subjects, the precision achieved by the incremental approach using mean joint probability (Pdf and Pdf-25) as the cluster homogeneity measure is lower than that achieved using mean variance (Var and Var-25) as the homogeneity measure given the same minimum point parameter. The use of probability density function makes assumption on the distribution of the data points in the cluster. The use of variance does not make such assumption and only evaluates the closeness of the data points to their cluster centroid.

4. Setting a value for the minimum point parameter improves precision in both cases of cluster homogeneity measures.

5. The average precision across all five subjects for the result using Hartigan index (H) is 58.4%.

6. The average precision across all five subjects for the result using mean variance and $MinPt = 25$ is 91.2%.

7. The incremental approach with mean variance and $MinPt = 25$ achieved an average 32.8% improvement in the precision over the non incremental approach using Hartigan index.

Fig. 4.16 shows the average recall of the five experiments. We observe similar trend as in the case of precision. The overall average recall for the result using Hartigan index (H) is 73.5%, while the result using mean variance and $MinPt = 25$ achieved an average 92.5% recall. The incremental approach improves the recall by an average of 19%.

Further, we note while the use of Hartigan index can estimate the number of clusters, it does not has the ability to distinguish the clusters between valid actions and random actions. On the other hand, the incremental approach shows great potential to reject random movements. To illustrate this claim, we look into the detail of one run of the incremental clustering algorithm. Table

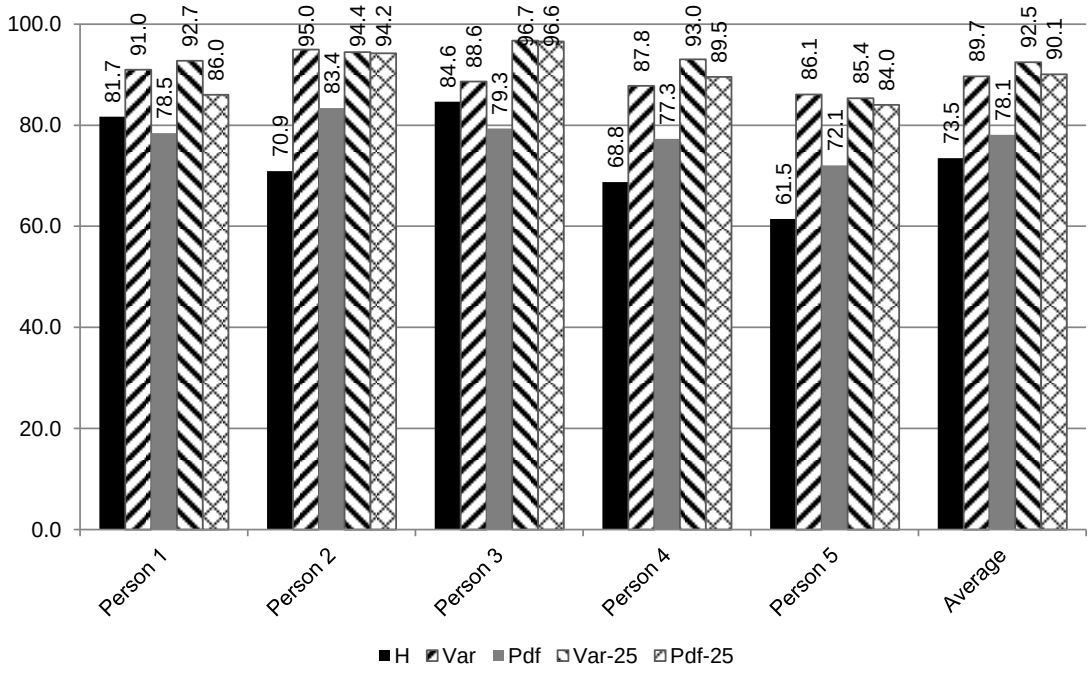


Figure 4.16: Average recall of different approaches for the data of each subject.

4.5 shows the complete result in one run of the incremental approach using mean variance and $MinPt = 25$ for the data of Person 4.

Table 4.5: One complete run of the incremental discovery algorithm using mean variance and $MinPt = 25$ for Person 4

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
k	24	23	22	21	20	19	18	17	16	15	13	11	7	5	3	U
A1	0	0	38	0	1	0	0	0	0	0	0	17	0	0	3	0
A2	0	0	0	1	0	0	0	0	0	55	0	0	0	0	0	0
A3	0	0	0	36	0	0	0	0	0	0	0	0	0	0	0	0
A4	0	30	0	0	0	0	26	0	0	0	0	0	0	0	20	0
A5	0	0	0	0	0	0	0	56	0	0	0	0	0	0	0	0
A6	0	0	0	0	0	0	0	0	33	0	0	0	0	20	0	0
A7	0	0	0	0	56	0	0	0	0	0	0	0	0	0	3	0
A8	56	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A9	0	0	0	0	0	39	0	0	0	0	17	0	0	0	0	0
RA	0	0	0	1	0	0	0	16	0	0	8	9	25	15	13	25

The first row, *i*, is the iteration of the incremental approach as given in Fig. 4.13. The second row is the *k*-value used in each iteration. The remaining rows are the actions as listed in Table

A.1. In the first iteration, $i = 1$, the $k = \sqrt{616} \simeq 24$. In this iteration, one cluster with lowest mean variance was found. This cluster contains all the 56 instances of Action 8 (Working on computer). These 56 instances were trimmed from the dataset, leaving $616 - 56 = 560$ instances. In the next iteration, $i = 2$, $k = \sqrt{560} \simeq 23$. One cluster was found and it contains 30 instances of Action 4 (Relaxing on couch). The incremental process continued until no further cluster can be obtained. A total of fifteen clusters were found, i.e. up to $i = 15$. Twenty five instances of the random movements (RA) were not assigned to any cluster as collected in the right most column, U. We observe a few things from Table 4.5:

1. The clusters obtained in early iterations are homogeneous, i.e. contains only one action.

2. The instances of random movements (RA) are only clustered towards the end of the incremental process.

The above behaviors are desirable for action discovery. One exception to the above observation is at eighth iteration, $i = 8$. At eighth iteration, a significant number of random movements (RA) were clustered with Action 5 (Still). Further investigation into the dataset revealed that these random movements (RA) happened to be sampled from instances when the subject was in still state.

4.8 Summary

In this chapter, we have proposed a set of features suitable for generalization of human action recognition. We suggested a frame rate of fifteen frames per action observation of two seconds window size. We demonstrated the potential of performing unsupervised human action recognition using just the skeleton data from an inexpensive depth sensor.

We addressed the problem of the prior specification of the number of clusters with our proposed incremental approach of clustering. The incremental approach of clustering is used for human action discovery. Unlike conventional approaches that "estimate the number of clusters" in the data, the incremental approach finds "good clusters" in the data. This behavior makes the approach suitable in noisy data.

The approach is implemented with simple clustering algorithm, i.e. K-means. The approach was evaluated on two different datasets comprising of twenty five actions and five subjects in total. The results suggest the use of mean variance as the cluster homogeneity measure. The results show that the proposed approach can potentially reject random movements or noise. The incremental approach also reduces computation time by trimming the dataset in each iteration and recalculating k-value based on the population of the trimmed dataset.

The results show that the approach achieved an overall average precision of 91.2% and recall of 92.5% across the dataset of the five subjects. This represents an improvement of over 30% in precision and 19% in recall when compared to the non incremental approach using cluster validity index.

While the development of the proposed approach was motivated by the purpose of human action discovery, it can be generalized and used in other applications requiring clustering without prior knowledge of the data and with significantly noisy data. It can also be adapted to deal with dynamic data since in each iteration the algorithm assumes a new dataset is made available for it to find one "best" cluster.

There remains the issue of confusion among highly similar activities. We have observed that the problem of sensitivity to random initialization is pronounced in the cases of highly confused actions. By reducing the confusion, the problem of sensitivity to random initialization will be minimized. However, this issue requires extending the feature set beyond solely using vision data.

Chapter 5

Human Action Modeling

When a new action is discovered, i.e., a cluster is found, members of the cluster become examples to learn a template or model for that action. Probability-based algorithms are most commonly used to model human activities and actions. In fact, Hidden Markov Model (HMM) [44] is one of the most popular models [7, 17] to build human activity or action models. HMM has well established mathematical ground and is versatile in its application. It has been used to model human actions since the early 90s [45]. It remains effective even at present time [46] when the objective is for activity or action recognition but not novelty in learning algorithm. In this work, we have chosen Hidden Markov Model (HMM) [47] to model the actions.

5.1 Mixture of Gaussians Hidden Markov Model

Human actions are time series of features, e.g., the change of joint positions over time. The combination of the various joint positions, representing a posture, is considered a state of an action. The changes of joint positions over time is considered state transition. The data, in the form of 3-D coordinates of joint positions is what the system observes in each state of the action. The observations contain errors and uncertainties due to the noise in the sensor data. This scenario can be specified by an HMM.

An HMM is characterized by the following parameters:

$$\lambda = (A, B, \pi) \tag{5.1}$$

where

$A = \{a_{ij}\}, 1 \leq i, j \leq Q$ is the state transition matrix, which defines probability that the next state will be S_j given current state is S_i, where $S = \{S_i\}, 1 \leq i \leq Q$ is the set of states and Q is the number of states.

$\pi = \{\pi_i\}, 1 \leq i \leq Q$ is the initial probabilities, where π_i is the probability of starting in State S_i.

$B = \{b_j(O)\}, 1 \leq j \leq Q$ is the emission probability matrix, where $b_j(O)$ describes the probability density of observing a continuous output O in state S_j.

We use HMM with Mixture of Gaussians to account for a continuous set in the output observation. The original HMM has the output observation in a set of discrete symbols. Our framework is designed to discover, learn and recognize potentially undefined number of actions. It will significantly degrade the quality of the model if the continuous output is quantized into a set of finite discrete output symbols. The required number of the output symbols will be prohibitively large in order to cover the 3-D volume around a human body while cater for the required resolution to discriminate similar actions. For this reason, Mixture of Gaussians is used to represent the output observations. For M-component Mixture of Gaussians, the emission probability density function is given as

$$b_j(x) = \sum_{k=1}^{M} c_{jk} \mathcal{N}(x; \mu_{jk}, \Sigma_{jk}). \tag{5.2}$$

where x is a p-dimensional continuous-valued data vector, c_{jk} is the mixture weight, $\mathcal{N}$ is the Gaussian or normal density function, μ_{jk} and Σ_{jk} are the mean vector and covariance matrix associated with state S_j and mixture component k.

Replacing B in Eq.5.1 with the parameters in Eq.5.2, the parameters to learn for the HMM model are a_{ij}, π_i, c_{jk}, μ_{jk} and Σ_{jk}. Let the output observation be a time series of length T, $O = \{x_1, ..., x_T\}$, the parameters are estimated using the following series of equations in Baum-

Welch algorithm [44].

The forward variable is defined as

$$\alpha_t(i) = P(x_1, ..., x_t, q_t = S_i|\lambda). \tag{5.3}$$

where q_t indicates the state at time step t.

$\alpha_t(i)$ can be solved by

$$\alpha_1(i) = \pi_i b_i(x_1),\ 1 \leq i \leq Q \tag{5.4}$$

and

$$\begin{aligned} \alpha_{t+1}(j) = \left[\sum_{i=1}^{Q} \alpha_t(i)\, a_{ij}\right] b_j(x_{t+1}) \qquad & 1 \leq t \leq T-1 \\ & 1 \leq j \leq Q. \end{aligned} \tag{5.5}$$

The backward variable is defined as

$$\beta_t(i) = P(x_{t+1}, ..., x_T, q_t = S_i|\lambda), \tag{5.6}$$

and $\beta_t(i)$ can be solved by

$$\beta_T(i) = 1,\ 1 \leq i \leq Q \tag{5.7}$$

and

$$\begin{aligned} \beta_t(j) = \sum_{j=1}^{Q} a_{ij} b_j(x_{t+1}) \beta_{t+1}(j) \qquad & 1 \leq t \leq T-1 \\ & 1 \leq j \leq Q. \end{aligned} \tag{5.8}$$

Using the forward and backward variables defined above, the probability of being in state S_i at time t and state S_j at time $t+1$, given the model and the observation sequence O, is defined as

$$\begin{aligned}\xi_t(i,j) &= P(q_t = S_i, q_{t+1} = S_j | O, \lambda) \\ &= \frac{\alpha_t(i)\, a_{ij} b_j(x_{t+1})\, \beta_{t+1}(j)}{\sum_{i=1}^{Q}\sum_{j=1}^{Q} \alpha_t(i)\, a_{ij} b_j(x_{t+1})\, \beta_{t+1}(j)}.\end{aligned} \tag{5.9}$$

The desired parameters of the model can then be estimated through an iterative update process using the following equations:

$$\hat{\pi}_i = \sum_{j=1}^{Q} \xi_t(i,j) \tag{5.10}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1}\sum_{j=1}^{Q} \xi_t(i,j)} \tag{5.11}$$

$$\hat{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T-1}\sum_{k=1}^{M} \gamma_t(j,k)} \tag{5.12}$$

$$\hat{\mu}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot x_t}{\sum_{t=1}^{T-1} \gamma_t(j,k)} \tag{5.13}$$

$$\hat{\Sigma}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot \left(x_t - \mu_{jk}\right)\left(x_t - \mu_{jk}\right)'}{\sum_{t=1}^{T-1} \gamma_t(j,k)} \tag{5.14}$$

where prime denotes vector transpose and $\gamma_t(j,k)$ is the probability of being in state j at time t and the kth mixture component accounting for x_t, i.e.,

$$\begin{aligned}&\gamma_t(j,k) \\ &\quad = \left[\frac{\alpha_t(j)\,\beta_t(j)}{\sum_{j=1}^{Q} \alpha_t(j)\,\beta_t(j)}\right] \times \\ &\quad\quad \left[\frac{c_{jk}\mathcal{N}\left(x_t, \mu_{jk}, \Sigma_{jk}\right)}{\sum_{m=1}^{M} c_{jm}\mathcal{N}\left(x_t, \mu_{jk}, \Sigma_{jk}\right)}\right].\end{aligned} \tag{5.15}$$

With the Mixture of Gaussians parameters, the resulting HMM model in Eq.5.1 is elaborated as Eq.5.16 where B has been replaced by μ, c and Σ:

$$\lambda = (A, \mu, c, \Sigma, \pi) \tag{5.16}$$

The objective of the learning phase is to determine the parameters in Eq. 5.16 for each action model.

5.2 The Learning Process

To learn the HMM, it is necessary to predefine the values of the number of states Q and the number of mixture of Gaussians M. To autonomously determine these values, we use an exhaustive search within a range of values and select the model that has the best performance in cross-validation test.

Fig. 5.1 summarizes the steps in the learning process. At the beginning, the cluster members are split into training and cross-validation sets as illustrated in Fig. 5.2. We use 8:2 training to cross-validation ratio. 80% of the members in the cluster are used to train the model, while 20% are used to perform holdout cross-validation of the model. Through exhaustive search of model parameters, the model with best cross-validation performance is chosen as the learned model.

Exhaustive search will find the best solution within the search space, however it is extremely computation demanding. It is necessary to constrain the range of the search values. We will suggest a suitable range of values based on our experimental outcomes.

5.3 Human Action Recognition Performance

We evaluate the effectiveness of the Mixture of Gaussians HMM in modeling the actions in our datasets and the learning strategy described in Fig. 5.1.

ALGORITHM: Learning of action model

INPUT: A cluster of observations $C = \{O_1, \ldots, O_n\}$

OUTPUT: HMM model of the action $\lambda = (A, \pi, c, \mu, \Sigma)$

ALGORITHM:

1: Set values of Q_{min}, Q_{max}, M_{min}, M_{max} and P_{train};

2: LL = -Inf;

3: Split members of C into training set C_{train} and cross-validation set C_{cv} with $|C_{train}| = P_{train}|C|$ and $|C_{cv}| = (1 - P_{train})|C|$;

4: For M_{min} to M_{max} %search number of mixtures

4.1: For Q_{min} to Q_{max}; %search number of states

4.1.1: Train HMM on C_{train} to give $\hat{\lambda} = (\hat{A}, \hat{\pi}, \hat{c}, \hat{\mu}, \hat{\Sigma})$;

4.1.2: Test the HMM on C_{cv} and computer average log likelihood LL_{cv};

4.1.3: If $LL_{cv} > LL$, $\lambda = \hat{\lambda}$;

Figure 5.1: The steps involved in the learning phase. Q_{min} and Q_{max} are the minimum and maximum value of the number of states respectively. M_{min} and M_{max} are the minimum and maximum value of the number of mixtures respectively. P_{train} is the proportion of data to be used as training set.

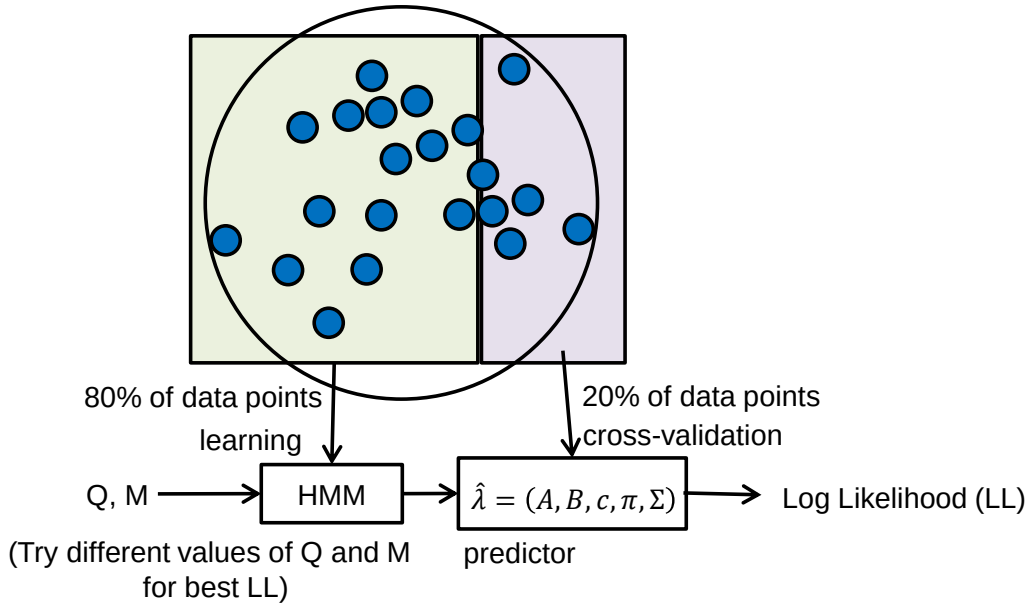


Figure 5.2: Splitting a cluster into learning and cross-validation sets.

5.3.1 Experiment

We model each action in the learning set as described in Appendix A for both datasets, CAD-60 and KOS-H16. The experiments were conducted without the random actions. This is a supervised learning scenario. The model learned for each action is then tested with the samples in test set as described in Appendix A.

In our experiment, we have used minimum number of mixtures $M_{min} = 1$, maximum number of mixtures $M_{max} = 7$, minimum number of states $Q_{min} = 2$ and maximum number of states $Q_{max} = 15$. Given that we have 15 frames per feature in one sample, we set $Q_{max} = 15$. Five runs of the experiment were carried out, and we report the average results.

5.3.2 Result and Discussion

Fig. 5.3 shows the average precision and recall of the HMM models to recognize unseen observations. The detailed results are given in Table 5.1 and 5.2. The learning and recognition performance has achieved over 90% for both precision and recall for all the five subjects. If we look at the detailed results in Table 5.1 for Persons 1 to 4, we can see that the recognition performance achieved 100% precision and recall rate in majority of the actions. This performance is in par with that achieved by Sung et al [13] using the more complex two layer maximum entropy Markov model (MEMM) on the same dataset.

Table 5.1: Precision and Recall of the HMM Modeling Result for Person 1 to 4.

	Person 1		**Person 2**		**Person 3**		**Person 4**		**Average**	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
brushing teeth	96.0	99.2	82.2	98.3	87.9	100	79.6	95.8	86.5	98.3
cooking (chopping)	100	95.0	100	77.5	98.9	83.3	100	75.0	99.7	82.7
cooking (stirring)	100	100	100	100	100	99.2	100	96.7	100	99.0
relaxing on couch	100	99.2	100	99.2	100	100	100	96.7	100	98.7
still (standing)	100	100	100	100	100	100	100	100	100	100
talking on couch	100	98.3	100	95.8	100	99.2	100	95.8	100	97.3
talking on the phone	100	100	100	100	100	97.5	100	100	100	99.4
working on computer	100	99.2	100	88.3	100	90.8	100	89.2	100	91.9
writing on whiteboard	100	100	100	94.2	100	100	100	96.7	100	97.7
Average:	**99.6**	**99.0**	**98.0**	**94.8**	**98.5**	**96.7**	**97.7**	**94.0**	**98.5**	**96.1**

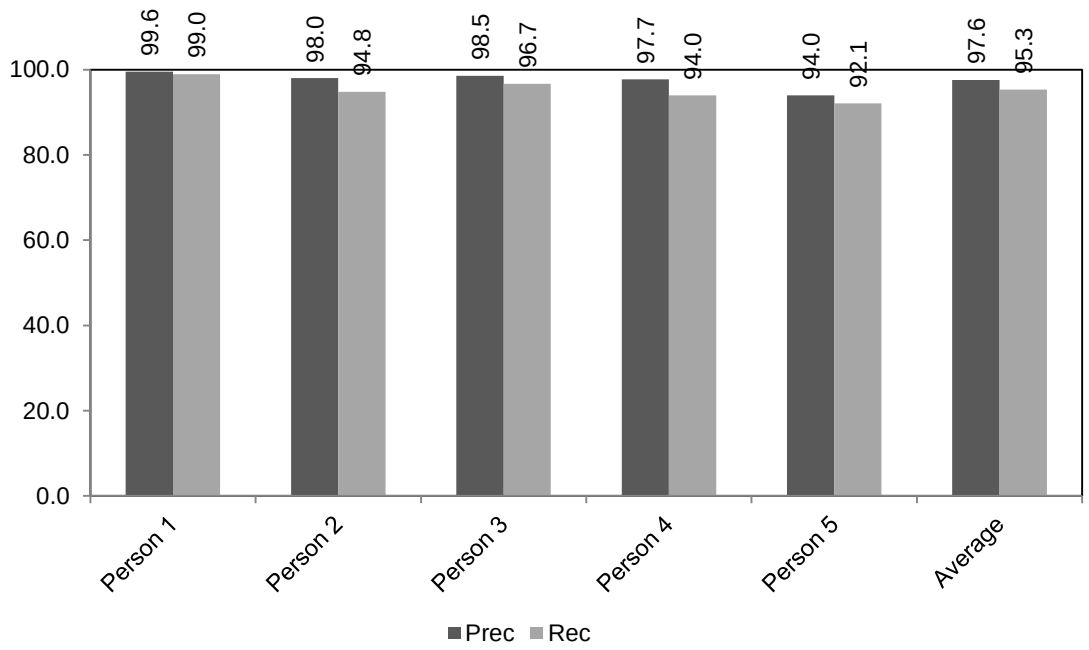


Figure 5.3: Average precision and recall of learning and recognition for each subject.

Table 5.2: Precision and Recall of the HMM Modeling Result for Person 5.

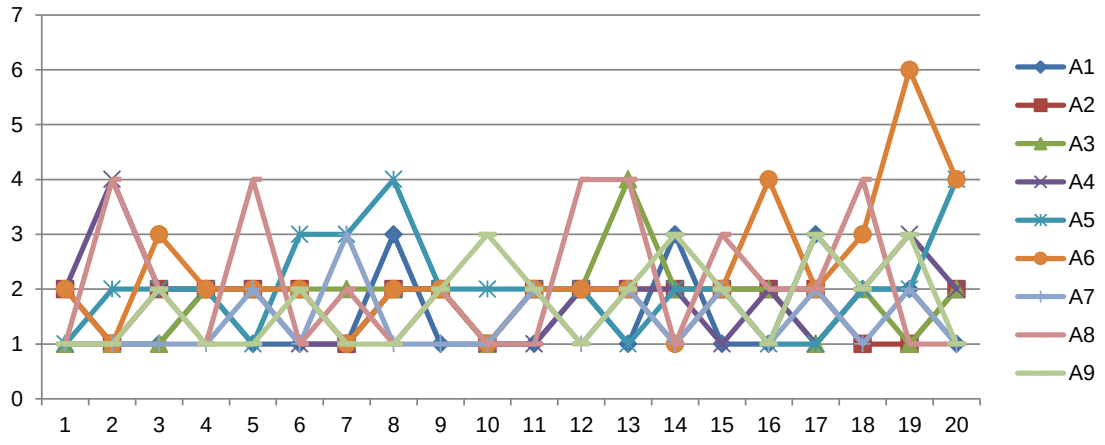
	Prec	Rec
bowing	98.1	89.2
drinking (left)	100	95.0
drinking (right)	100	94.2
sit	86.0	89.2
sit down	65.7	100
stand	100	80.0
stand up	89.0	88.3
talking on phone (left)	90.5	98.3
talking on phone (right)	100	97.5
walking	95.4	95.0
wave bye (left)	100	92.5
wave bye (right)	100	93.3
wave come (left)	99.1	87.5
wave come (right)	99.1	80.0
wave go (left)	96.1	95.8
wave go (right)	84.3	97.5
Average:	**94.0**	**92.1**

Fig. 5.4 shows the values of M and Q in all runs of the experiments. While the algorithm was

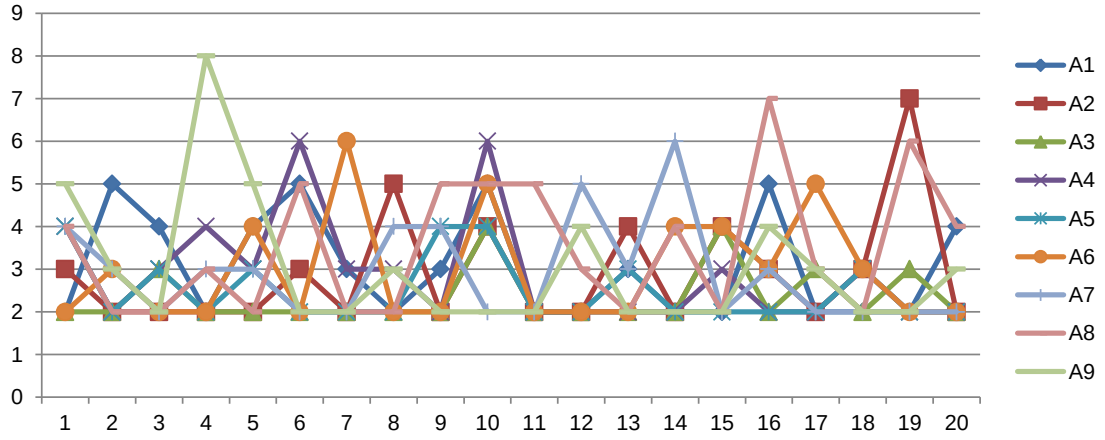
run with the range $1 \leq M \leq 7$ and $2 \leq Q \leq 15$, majority of M and Q values lie in the range of $1 \leq M \leq 4$ and $2 \leq Q \leq 6$.

5.4 Summary

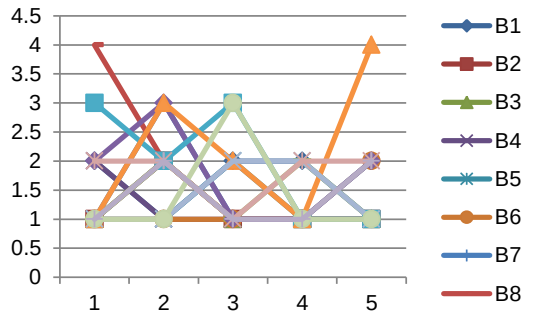
Mixture of Gaussians Hidden Markov Model can effectively model the actions in our datasets with overall average precision of 97.6% and recall of 95.3%. The proposed learning algorithm effectively found the required number of Gaussian components and number of states. We observe that the majority values of M and Q are in the range of $1 \leq M \leq 4$ and $2 \leq Q \leq 6$. This knowledge will allow us to reduce computation time of the learning phase by limiting the search range of M and Q.



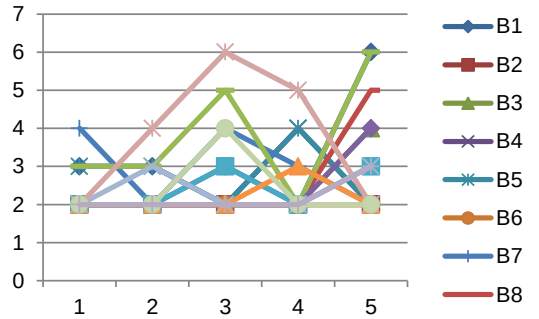
(a) *M* parameter for the twenty runs (P1 to P4 each five runs) of learning algorithm.



(b) *Q* parameter for the twenty runs (P1 to P4 each five runs) of learning algorithm.



(c) *M* parameter for the five runs of learning algorithm on P5 data.



(d) *Q* parameter for the five runs of learning algorithm on P5 data.

Figure 5.4: *M* and *Q* parameters in all runs of the learning algorithm.

Chapter 6

A Prototype Implementation on A Mobile Robot

We have build a prototype for the application of the auto-HaR framework based on a mobile robot. In this chapter, we describe the details of the auto-HaR implementation for this prototype that have not been covered in the other chapters.

6.1 Hardware Descriptions

The ultimate goal of developing the auto-HaR is to enable the application of human action recognition technology in everyone's home. One of the important criteria is to keep the cost low. The hardware comprises of four items as shown in Fig. 6.1 and listed below:

1. A Microsoft Kinect Xbox 360 sensor
2. An iRobot Create Mobile platform with charging dock
3. A computing unit
4. A body frame

The total cost of the hardware is less than JPY150,000.00.

Figure 6.1: Hardware components: iRobot Create mobile base, laptop, Microsoft Kinect Xbox 360 and acrylic body frame.

The iRobot Create is a low-cost mobile base. It has sensors and actuators to facilitate its safe navigation in the normal household setting. It is the same mobile base being used in the popular consumer robotic vacuum cleaner, iRobot Roomba. The mobile base is used to enable the use of a single sensor to observe human activities. It provides the human following capability.

The Microsoft Kinect Xbox 360 sensor is the sensor used to capture the input data required for the auto-HaR framework. Kinect has a combination of sensors as shown in Fig. 6.2 allowing it to deal with both visual and audio data. It is being used in gaming consoles. In our application, only visual data is used. The specifications of Kinect relevant to our application are:

- Viewing angle of 43° vertical by 57° horizontal field of view
- Color VGA motion camera 640 x 480 pixel resolution @30 FPS
- Depth Camera 640 x 480 pixel resolution @30 FPS
- Minimum viewing distance- 0.8m
- Maximum viewing distance- 3.5m

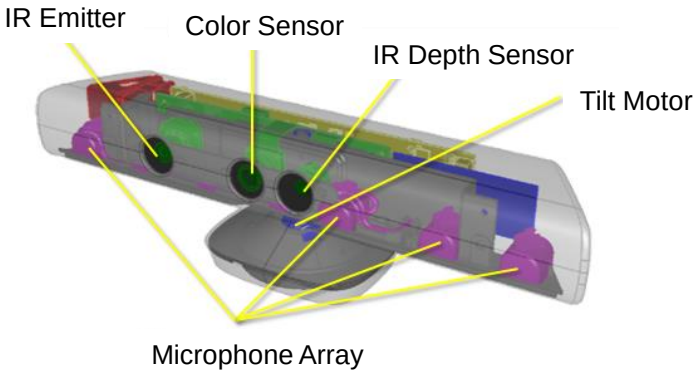


Figure 6.2: Internal of Microsoft Kinect [3].

The computing unit is basically a computer to do all the processing to deal with the sensor data, execute the algorithms in the auto-HaR framework and control the mobile robot. We have used a Lenovo Thinkpad E440 with the following specifications:

- Intel Core i5-4200M CPU @ 2.50GHz with integrated graphics
- 4GB DDR3 RAM
- 256GB SSD HDD
- Windows 7 Professional 64-bit OS

6.2 Software Descriptions

For the prototype, we have used two open source programming platform:

1. Processing [48]. Processing is an integrated development environment built on Java programming language with focus on visual output based on sketches.
2. GNU Octave [49]. GNU Octave is a high-level interpreted language, primarily intended for numerical computations. It adopt similar syntax to the Matlab allowing easy adaption of Matlab codes.

Figure 6.3: Photos of the prototype mobile robot with its charging dock (right).

The Processing is used to interface with the Kinect to capture data, provide user interface and control the mobile robot. It then triggers the Octave to execute algorithms in auto-HaR on the collected data. We describe the implementation details of the software system in this section.

6.2.1 Overview

Fig. 6.4 shows the software structure for the prototype mobile robot.

The Kinect provides sensor data to the auto-HaR and robot control modules. The robot control module uses the Kinect to detect the presence of the human and, track and follow the human using its navigation behavior.

The auto-HaR framework uses the Kinect data to perform its operation as described in Chapter 3. It receives trigger signal from the robot control module when there is human subject to track and performs the autonomous action recognition. A task scheduler is used to initiate the discovery phase at scheduled time, e.g. at the end of a day.

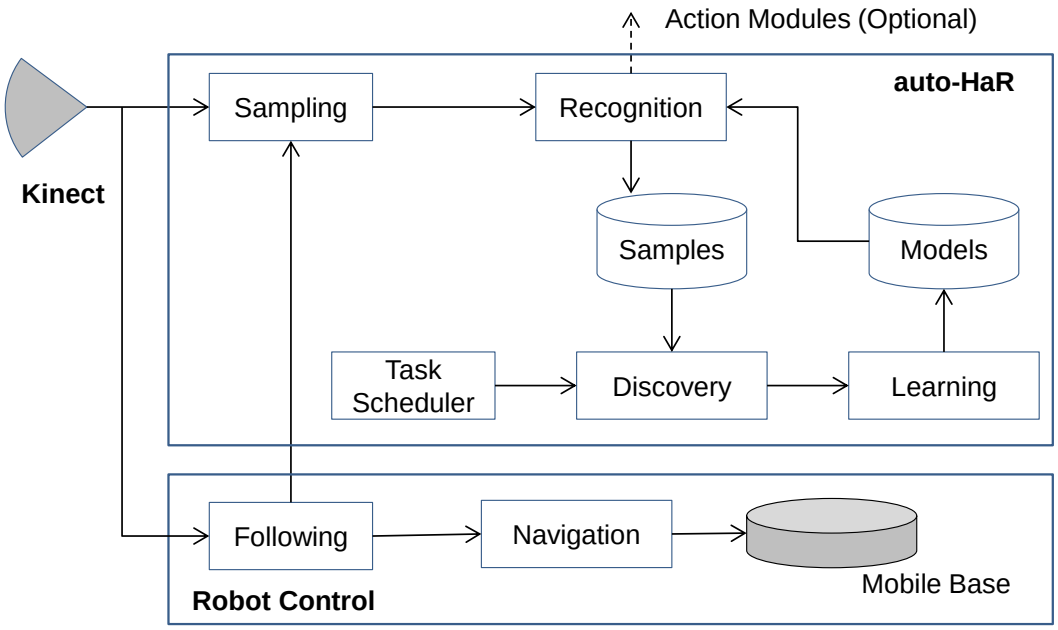


Figure 6.4: Software structure.

6.2.2 Human Following

To use a single sensor to observe the daily activities of a person requires that the sensor goes wherever the person goes. This is the task of the human following module in the robot controller. Human following is achieved by keeping a desired distance between the sensor and the person.

The OpenNI SDK can provide the 3D coordinates of the Center of Mass (CoM) of a person in the view of the Kinect. This is provided without having to track the skeleton. The z-coordinate gives the depth or outward distance from the sensor. By keeping the z-coordinate of the CoM within a desired range, the mobile robot will constantly follow a person.

Fig. 6.5 shows the operating range of Kinect. It is desirable to maintain the person within the normal range of 0.8 to 4 meters. To account for limited space in the premises, we keep the person between 1 to 2 m.

6.2.3 Sampling: Input Segmentation

As described in Section 4.2, we have used sliding window to segment the continuous input data into discrete action instances as shown in Fig. 6.6. We have used the reduced frame rate and

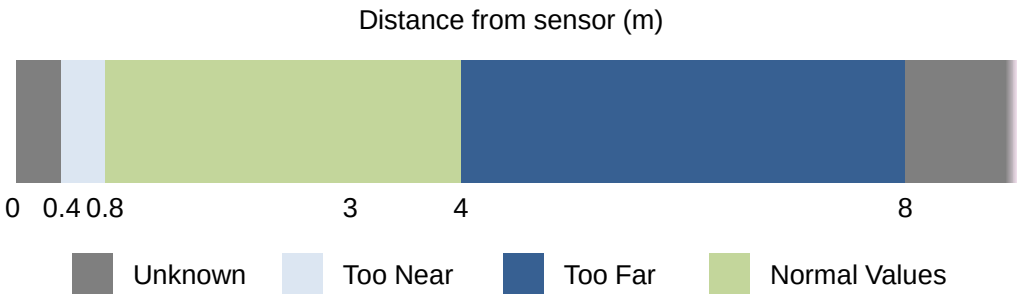


Figure 6.5: Operating range of Kinect.

sample fifteen frames in two seconds. Given the sensor data is at thirty frame per second (30 fps), there are sixty frames in two seconds. The sampling module practically record one frame in every four frames received. The recorded data format is shown in Fig. 6.7. Each frame contains the 3D coordinates of fifteen joint positions.

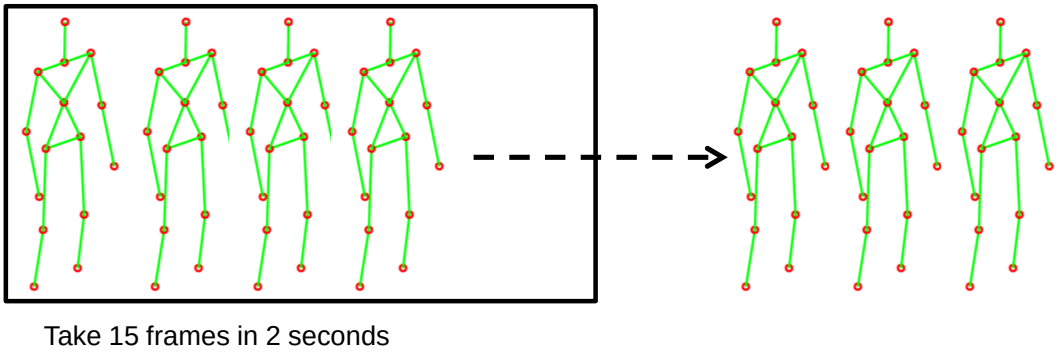


Figure 6.6: Sliding window sampling.

The complication comes in deciding the timing to sample the next instance. As the algorithm does not know the start and begin of an action, if it sample an instance every two seconds as shown in Fig. 6.8, the sampling process can miss actions performed in between the two sample instance.

If we sample an instance after every frame as shown in Fig. 6.9, an action can be sampled into many instances. Some of these instances will be from the beginning of the action, while some towards the end. The high number of peripheral movements will increase the confusion in discovery phase. Further, there will be a large number of data to store.

In the prototype, we have used a random timing to sample three instances every two seconds

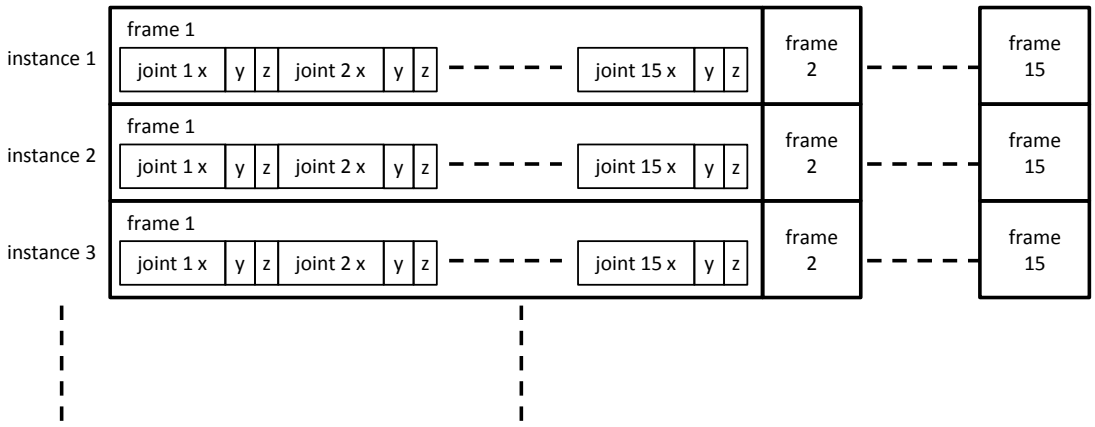


Figure 6.7: Raw data format.

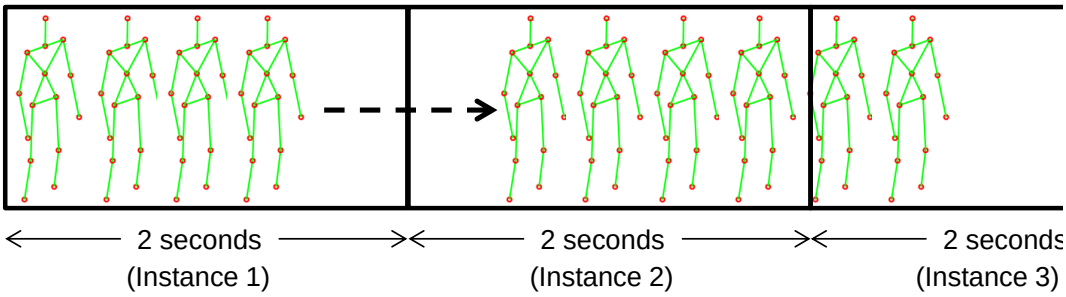


Figure 6.8: Sampling every two seconds.

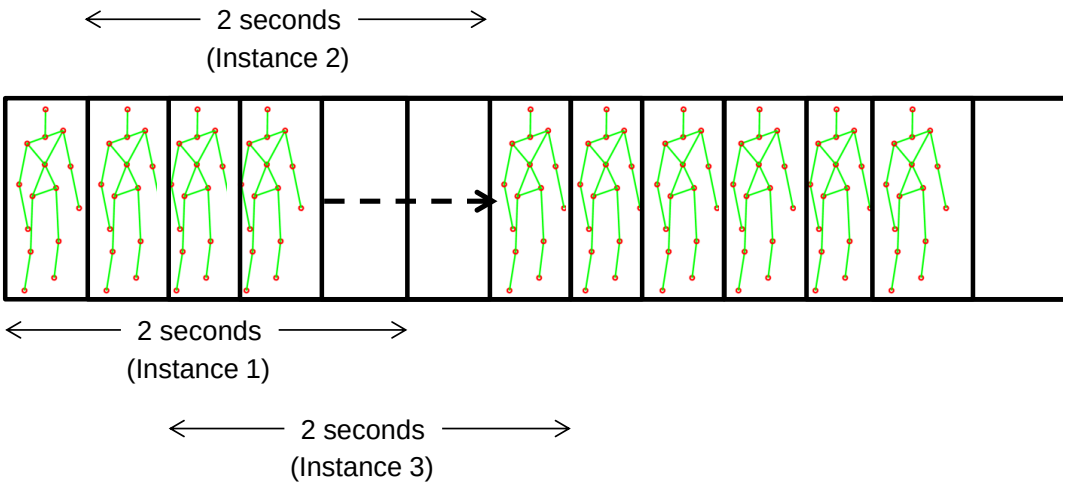


Figure 6.9: Sampling every frame.

as shown in Fig. 6.10. This is a compromise between the two extreme cases described above.

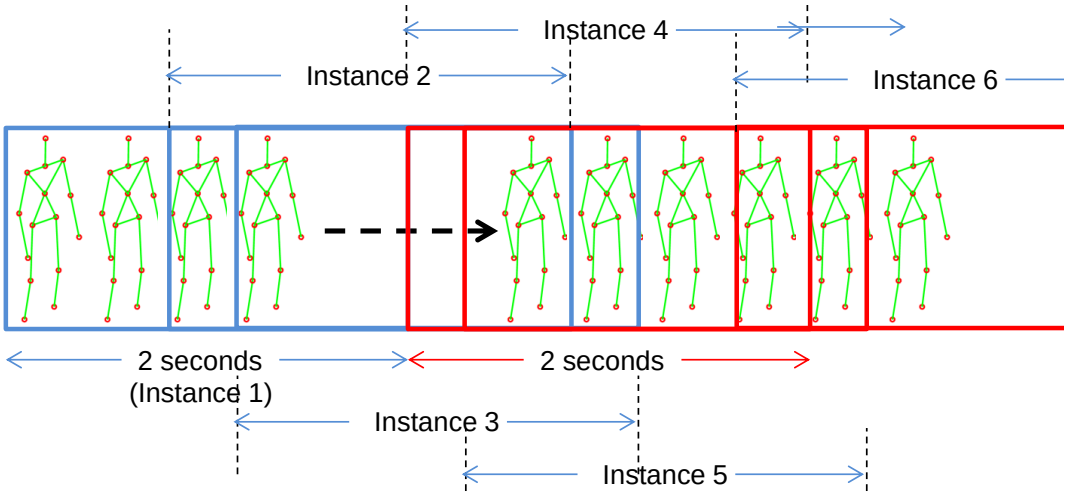


Figure 6.10: Sampling three instances at random timing every two seconds.

6.2.3.1 Coordinate Frame Transformation

To ensure the features extracted are view invariant, the coordinates of the joints are transformed from the camera coordinate frame $\{C\}$ to a coordinate frame on the body $\{B\}$ as illustrated in Fig. 6.11.

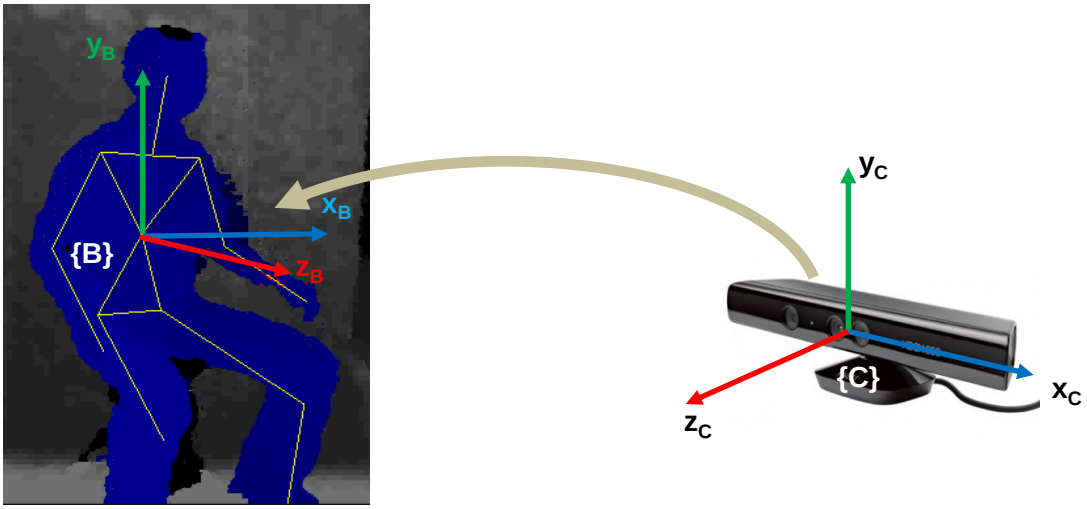


Figure 6.11: Coordinate frame transformation.

We have chosen to place the body coordinate frame $\{B\}$ at the torso joint of the first frame of an action instance. By doing so, we avoided computing the transformation matrix in every frame. Also, by fixing the $\{B\}$ at first frame, we can observe the translation movement of the joints;

otherwise the person will appear to be performing the action on the same spot, i.e. at stationary location. This is illustrated in Fig. 6.12.

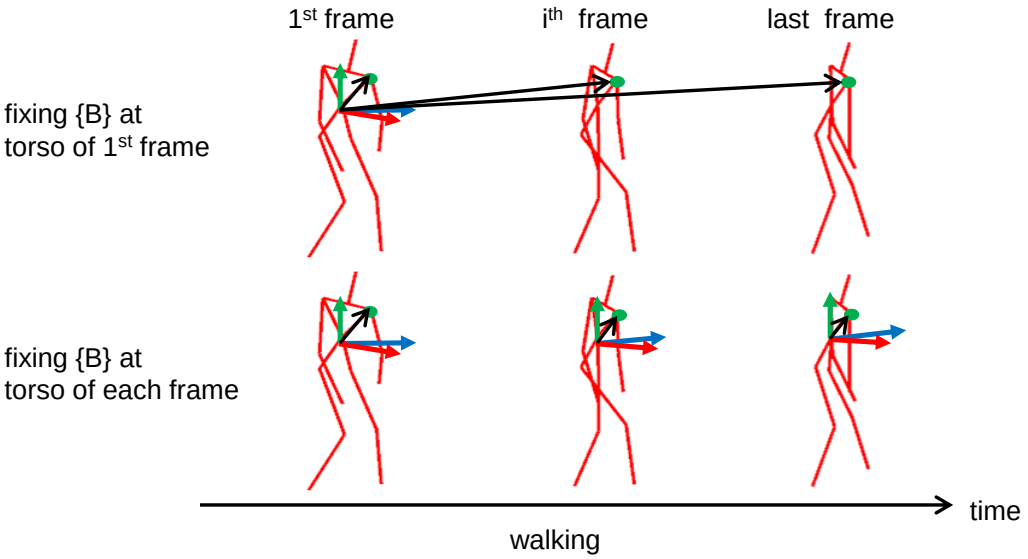


Figure 6.12: Reference frame at torso joint of first frame in comparison with fixing the reference frame at torso joint of every frame.

To simplify the transformation, we maintain the y-axis in $\{B\}$ parallel to the y-axis in $\{C\}$, i.e. pointing upward. Also, the x-z plane in $\{B\}$ is defined parallel to that in $\{C\}$. In this way, the required transformation comprises (only) of a rotation around y-axis in $\{C\}$ to align $\{B\}$ such that its z-axis is pointing to the front of the person and a translation from the origin of $\{C\}$, i.e. $< 0,0,0 >$, to the torso joint position in the first frame. This is illustrated in Fig. 6.13.

To transform the coordinates of a point in $\{C\}$, $\vec{p^C}$, to its coordinates in $\{B\}$, $\vec{p^B}$, we need to determine the homogeneous transformation matrix, T_C^B. The following steps describe the operations involved to transform a point from $\{C\}$ to $\{B\}$.

First obtain the following joint coordinates in $\{C\}$ from the *first frame* of an action instance captured from Kinect.

$$torso\ joint\ \vec{t}(1) = \begin{bmatrix} t_x(1) \\ t_y(1) \\ t_z(1) \end{bmatrix} \tag{6.1}$$

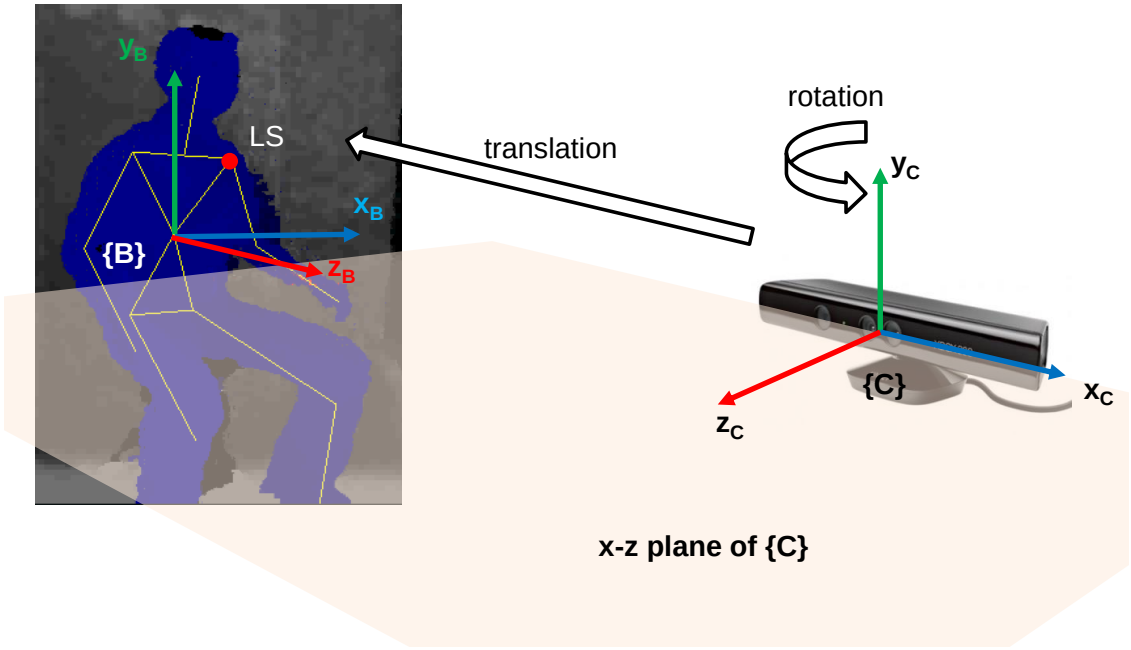


Figure 6.13: Coordinate frame transformation.

$$left\ shoulder\ joint\ \vec{ls}(1) = \begin{bmatrix} ls_x(1) \\ ls_y(1) \\ ls_z(1) \end{bmatrix} \tag{6.2}$$

$$right\ shoulder\ joint\ \vec{rs}(1) = \begin{bmatrix} rs_x(1) \\ rs_y(1) \\ rs_z(1) \end{bmatrix} \tag{6.3}$$

Second, form the transformation matrix (in homogeneous form, i.e. 4x4) to translate the origin from $\{B\}$ to the origin of $\{C\}$. Note the translation is the negation of the torso joint vector in the first frame.

$$T(trans)_C^B = \begin{bmatrix} 1 & 0 & 0 & -t_x(1) \\ 0 & 1 & 0 & -t_y(1) \\ 0 & 0 & 1 & -t_z(1) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.4}$$

Third, obtain the unit vectors of the x-axes of the two reference frames, $\hat{x_B}$ is parallel to the shoulder of the subject.

$$\hat{x_{B'}} = \frac{1}{\left\| \left(\vec{ls}(1) - \vec{rs}(1) \right)_{y=0} \right\|} \left(\vec{ls}(1) - \vec{rs}(1) \right)_{y=0} \quad and \quad \hat{x_C} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{6.5}$$

Forth, compute the rotation angle θ using the left and right shoulder joints,

$$\theta = sign\left((\hat{x_{B'}} \times \hat{x_C})_y \right) cos^{-1} (\hat{x_{B'}} \cdot \hat{x_C}) \tag{6.6}$$

Fifth, form the homogeneous transformation matrix for rotation,

$$T(rot)_C^B = \begin{bmatrix} cos\theta & 0 & sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -sin\theta & 0 & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.7}$$

Sixth, compose the homogeneous transformation matrix T_C^B,

$$T_C^B = T(rot)_C^B * T(trans)_C^B \tag{6.8}$$

Finally, given a point P whose coordinates in $\{C\}$ is given by

$$\vec{p^C} = \begin{bmatrix} p_x^C \\ p_y^C \\ p_z^C \end{bmatrix} \tag{6.9}$$

Pad the vector and compute its coordinates in $\{B\}$,

$$\begin{bmatrix} \vec{p^B} \\ 1 \end{bmatrix} = \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \\ 1 \end{bmatrix} = T_C^B [\begin{bmatrix} \vec{p^C} \\ 1 \end{bmatrix} \tag{6.10}$$

6.2.3.2 Local Vector Features

In Section 4.2 we explained the use of local vectors to represent range of movements as the features for an action instance. The local vector is computed from the difference between the 3D coordinates of two joints as illustrated in Fig. 6.14.

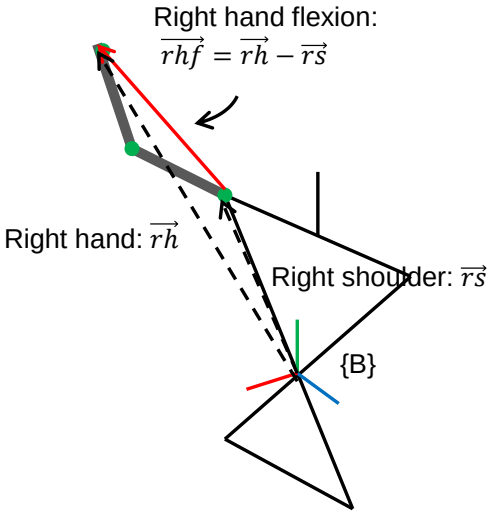
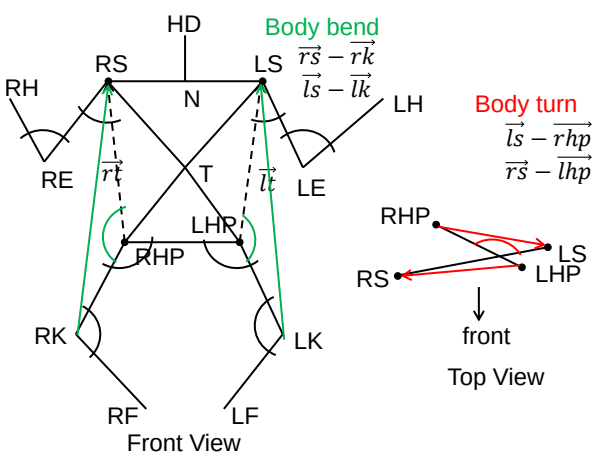


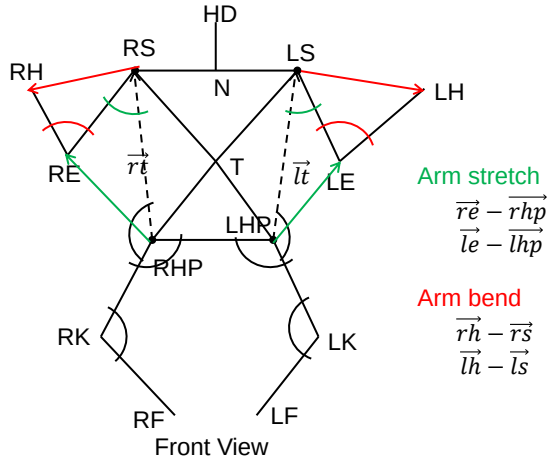
Figure 6.14: Right hand flexion.

The feature set comprises of twelve range of movements and two hand to head interaction vectors as shown in Fig. 6.15. In total, we have fourteen 3-D vectors in each frame giving $14 \times 3 =$

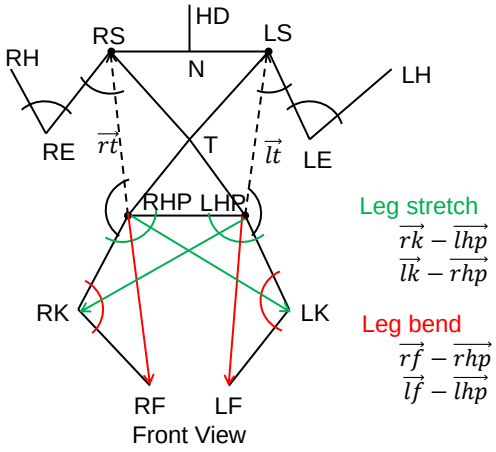
42 features per frame. The spatial information (posture) of the human action is captured by the coordinates in each frame whereas the temporal information (motion) is captured through the changes across the sampled frames.



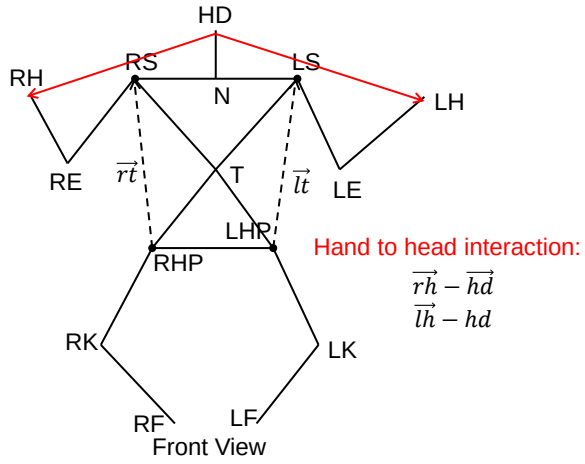
(a) Body bend and turn.



(b) Arm stretch and bend.



(c) Leg stretch and bend.



(d) Hand to head interacts.

Figure 6.15: The fourteen vectors in feature extraction.

With 15 frames per observation, we have a total of $42 \times 15 = 630$ features per action observation or instance. The format of the data after the processing is as shown in Fig. 6.16.

This is the data being fed to the auto-HaR framework. The implementations of the discovery and learning, and recognition modules follow the algorithms described in Chapter 4 and 5.

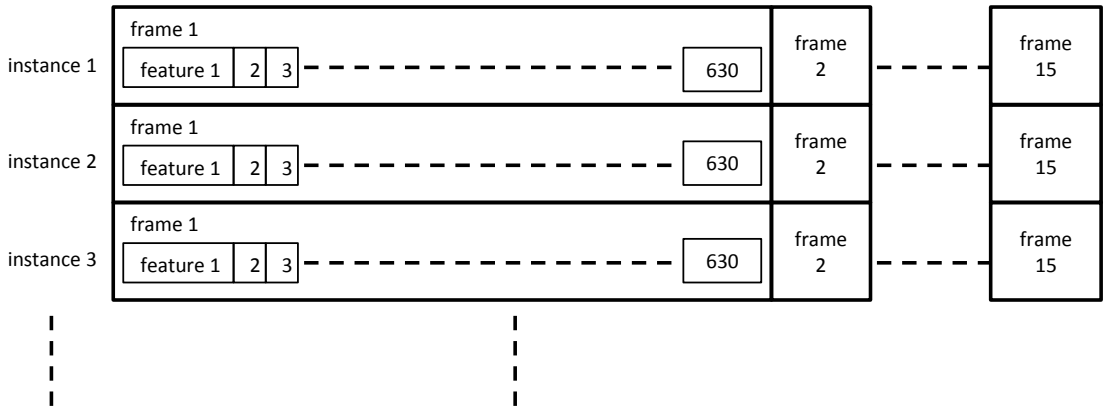


Figure 6.16: Processed data format.

6.3 Evaluation of Real Time Operation

The exercise to implement the proposed framework in a real life robot allowed us to identify issues to be addressed in real time operation. It allowed us to to identify necessary further works to apply the framework in real life applications.

A thorough evaluation of the real life system is beyond the scope of this dissertation. A thorough evaluation will require committing a significant amount of time for a human subject to live with the mobile robot, and to manually label the action instances for evaluation purpose. We have however, conducted short real time experiments with the mobile robot observing a subject.

In the experiment, the subject performed three simple actions, walking, standing and sitting, in front of the mobile robot. By keeping the action set minimal, we keep the laborious annotation process to minimal. We note here that the annotation is meant for evaluation purpose. The framework does not require the annotation to operate. We let the robot observe the subject for a multiple periods of five to ten minutes. The results from the auto-HaR framework were recorded with the corresponding frame number. The results were visually inspected by drawing the corresponding frames to determine the discovered and recognized action instances, as shown in Fig. 6.17.

We label transitions between actions and those action samples with significant errors in skeleton data as noise. However, with the random sliding window approach, there are action instances that contain some transition between actions. Fig. 6.18 shows a few frames with skeleton data

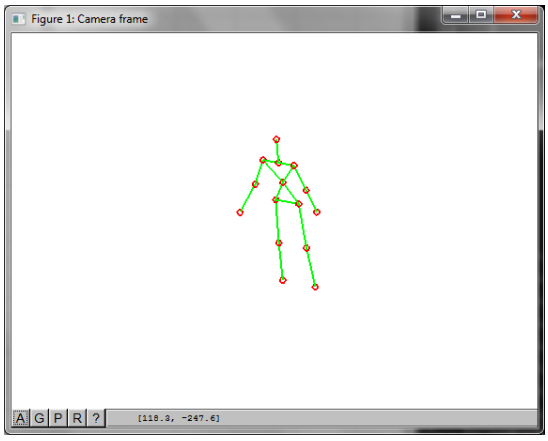


Figure 6.17: Visual inspection of the frames in an action instance.

noise.

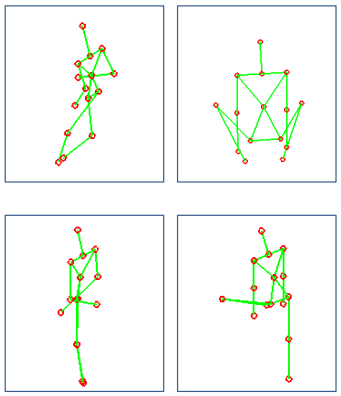


Figure 6.18: Frames with skeleton data noise.

With the time constraint in this study, we have annotated a short period of observation that have resulted in 336 action instances being used for the discovery, while 97 instance were recognized. Table 6.1 shows the number of each action instance that we had identified from the 336 instances in the short observation period.

The discovery phase discovered five clusters when asked to discover after the short period of observation. Fig. 6.19 shows the result of the discovery phase in a confusion matrix form.

Cluster 1 captured 36 instances of standing (Action 1) out of the 39 total members in it. Cluster 1 is a good representation of the standing action. Based on majority membership, we assign this

Table 6.1: Actions in the real time study.

	Label	Action	Qty
1.	Action 1	Standing	113
2.	Action 2	Walking	73
3.	Action 3	Sitting	100
0.	Noise	Noise	50

			Actual Actions			
	Assignment	Clusters	1	2	3	0
	0	U	0	11	9	8
Clusters	1	1	36	1	1	1
	1	2	42	5	21	15
	2	3	0	18	6	4
	3	4	0	10	17	5
	2	5	2	12	7	4

Figure 6.19: Confusion matrix in discovery phase.

cluster as Action 1 for the purpose of evaluating the recognition in the later stage. Likewise, Cluster 2 has been assigned as Action 1. However, Cluster 2 contained significant number of other actions. Following the same approach, Cluster 3 and 5 have been assigned as Action 2 (walking), while Cluster 4 has been assigned Action 3 (sitting). Using this assignment, we have computed the precision and recall in the discovery phase. The result is given in Table 6.2.

Table 6.2: Precision and Recall in discovery phase during real time operation.

		Prec	Rec
1	Standing	63.9	97.5
2	Walking	56.6	52.
3	Sitting	53.1	27.9
	Average:	**57.9**	**59.3**

We observe that the first cluster in the discovery phase has high purity. However subsequent clusters contain significant impurities. The average precision and recall based on the real time study are 57.9% ad 59.3% respectively.

Fig. 6.20 shows the confusion matrix during the recognition phase. Ninety seven action instances were recognized by the five action models learned from the discovery phase outcome.

Based on the assignment given in the discovery phase, we computed the precision and recall

Discovered Models	Assignment	Models	Actual Actions 1	2	3	0
	1	1	9	6	3	4
	1	2	14	3	12	5
	2	3	3	4	3	1
	3	4	5	3	4	5
	2	5	2	4	6	1

Figure 6.20: Confusion matrix in recognition phase.

of the recognition in Table 6.3.

Table 6.3: Precision and recall in recognition phase during real time operation.

		Prec	Rec
1	Standing	41.1	69.7
2	Walking	33.3	40
3	Sitting	23.5	14.3
	Average:	**32.6**	**41.3**

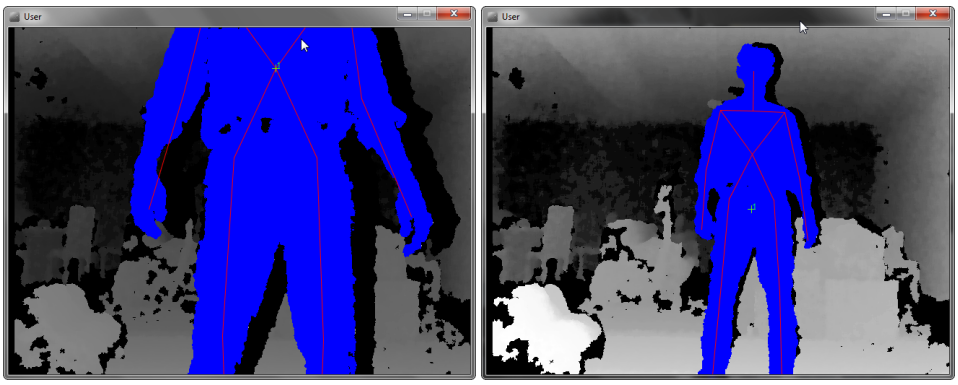
Given the impurities of the clusters in discovery phase, the low recognition rate is anticipated. The real time performance based on the short observation period has an average recognition precision of 32.6% and recall of 41.3%.

We attribute the poor performance in real time operation to two major factors: the segmentation of an action instance, and the noise in the skeleton data. When evaluating the framework using the two datasets in Appendix A, the segmentation was performed on data that have minimal transitional movements and skeleton data noise. The noise introduced in the datasets in Appendix A are movements and postures that differ from the valid actions. They do not contain skeleton data noise where in an instance of fifteen frames, one or two frames may have skeleton data noise. They do not contain transitional movement between two valid actions.

Given the above observations, further works to improve the segmentation strategy and to deal with noise in skeleton data will be essential.

We have also observed the limitation in the existing depth sensor, Kinect. The Kinect starts to detect a human skeleton at a distance of 900mm when the subject has a height of 1700mm. At this distance, however there are significant noise in the data for the head and feet joints. It can reliably

detect the whole body of the same subject at a distance of nearly 2000mm. Fig. 6.21 shows the field of view of the Kinect at different distances given a subject of 1700mm height.



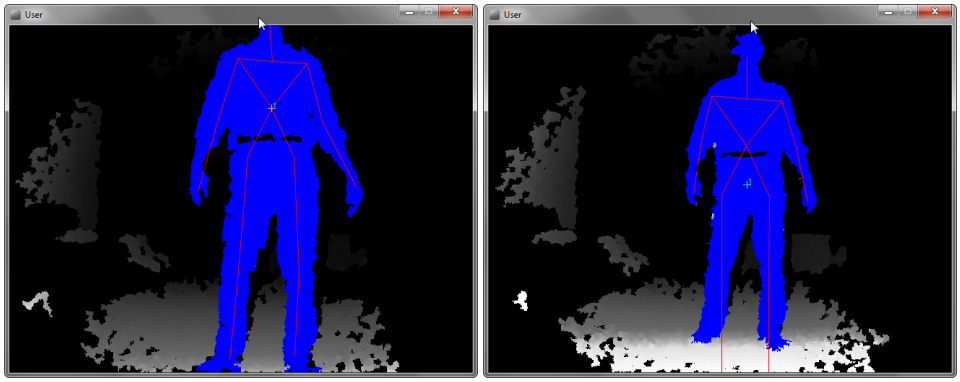
(a) 900mm (b) 2000mm

Figure 6.21: Field of view of Kinect at different distance, with a subject of height 1700mm.

The required distance for reliable skeleton detection is not desirable in a normal home, where space may be constrained. The wide angle lens (Zoom lens) available for the Kinect can reduce the distance by nearly half. Fig. 6.22 shows the field of views at different distances for the same subject when using the Zoom lens. However, with the Zoom lens, the depth data are significantly deteriorated. The skeleton data noise are significantly increased, and the response time is significantly increased. Microsoft Kinect has a near mode operation that can detect skeleton at a distance as low as 400mm. However, it does not increase the width of the view and only a few joints are detected.

6.4 Summary

Real life implementation faces a different set of problems for the auto-HaR framework. With the datasets in Appendix A, we have shown the ability of the auto-HaR to perform all the stages in human action recognition by itself. We showed the ability of incremental approach of clustering to enable discovery of human actions from undefined set of actions. In the short experiment in real life situation, the framework showed its potential to function as designed. We could see it discovered actions by its own, learn the actions and perform recognition. However, we see two



(a) 900mm (b) 1200mm

Figure 6.22: Field of view of Kinect with Zoom Lens at different distance, with a subject of height 1700mm.

important issues requiring attention to ensure reliable real time operation. The segmentation of action instances need to be able to handle transitional actions. The noise from skeleton data should be dealt with.

Further more, we observe the limitations in the consumer depth sensor. Improvement in the technology of the consumer depth sensor, especially in increasing the size of the viewing field, will significantly increase the usability of the auto-HaR framework in real life applications.

Chapter 7

Conclusions

A framework to autonomously discover, learn and recognize human actions has been developed. A novel incremental approach of clustering has been developed to enable human action discovery. The algorithms were evaluated on two datasets and showed average precision and recall higher than 80% in all phases. While the framework can deal with simple actions with with high precision and recall, it requires further work to improve its performance when dealing with highly resembling actions. The sampling process can deal with stationary and continuous actions such as standing and walking, enhancement is required to deal with situations when changing from one action to another as well as more complex actions.

While there are further problems to be resolved, this study provides the ground work for further research works to realize the use of human activity recognition technologies in normal home setting using low-cost devices. The incremental approach of clustering developed in this dissertation can be adapted to deal with dynamic data in other applications.

7.1 Summary

This dissertation has presented the development of a framework for human action recognition, *auto-HaR*, for applications in a normal home setting where embedded sensors are not available, and human operation to label or train the system is not desirable. It is a step forward in realizing

the adoption of human activity recognition technologies in our everyday life.

The proposed framework clearly divides the process of human action recognition into three stages or phases: discovery, learning and recognition. In doing so, the process can incorporate supervised and unsupervised learning approaches in different stage making it possible to realize the whole process autonomously. The framework uses data from low-cost depth sensor, Kinect. Only the 3D coordinates of the joints are required. By doing so, the framework can be implemented at low cost and the use of only depth data moderates the concern on privacy intrusion of using vision sensor in human activity monitoring in our everyday life.

The major challenge lies in the discovery phase, which is less developed than the learning and recognition phases.

To facilitate the discovery of an undefined number of actions, we investigated a feature set based on human range of movement. We have shown that correlation-based feature selection is not desirable for the generation of HAR. The set of features based on human range of movement includes the necessary local vectors that represent all possible movements of human limbs. In the investigation, we found it is sufficient to use fifteen frames of a two second segment to represent an action instance. There was no noticeable degradation in clustering performance using fifteen frames as compared to using sixty frames in two seconds at full frame rate of the sensor.

A novel incremental approach of clustering based on the well established K-means has been developed to address the requirements of the discovery phase. In particular, it deals with the uncertainty of random movements or noise, and the undefined number of actions to be discovered. Our clustering approach has the notion of "looking for good clusters" rather than "finding optimal k-value". An optimal k-value is irrelevant in the situation when the clusters are being discovered incrementally and possibly indefinitely from potentially noisy data.

In human action discovery phase, an average precision of 80.4% and recall of 83.8% were achieved when the algorithm was evaluated on our dataset of sixteen actions and a third party dataset of nine actions. In the experiment, random movements were included in the dataset. The algorithm and features at current state appear to have difficulty in distinguishing closely resembling actions, e.g. talking over phone and drinking.

The output from the discovery phase are clusters of discovered actions. Each cluster contains examples for one action. They can be used to learn model of the action using supervised approach. Human actions or activities are time series with errors and uncertainties. Variants of Hidden Markov Model (HMM) have been widely used to model human activities with good account of success. We have used Mixture of Gaussians HMM to model the action with multi-dimensional features.

In the evaluation of the use of Mixture of Gaussians HMM for the action model, an average precision of 97.6% and recall of 95.3% were achieved. This indicates the suitability of the model to represent the actions in the datasets. We have also determined that suitable range of the values for the number of mixtures (M) is from 1 to 4, and the number of states (Q) is from 2 to 7. This knowledge enables the algorithm to constrain its search when it autonomously determine the suitable M and Q value for each new action.

The final phase of the auto-HaR framework uses the models found in the learning phase to recognize known actions in new observations. Our experimental results show that recognition rate of the overall framework is at an average precision of 94.3% and recall of 91.4%.

While the framework has shown good performance on the two datasets used, more works are required to improve the performance in real-time operation. We have identified a number of future research works in Section 7.3.

7.2 Limitations

At the moment, the framework has the following limitations:

1. It works with a single person. This is analogous to the situation where the system is expected to monitor and support a home user. We consider this to be an acceptable situation and that each person has his/her unique way of performing an action.

2. It operates on best effort basis. It does not learn all actions at one time. It is expected to incrementally discover new actions and learn the models. One day it may learn how the person walk, the next day it may learn how the person sit. This resembles the way children

learn about their environment.

3. It intends to discover atomic activities or actions.

4. With the current set of features, it cannot discriminate highly similar actions such as scratching one's ear and talking on phone. When relying solely on visual information, human themselves have problem in discriminating actions with similar posture and motion dynamic.

5. The learned action models do not have a linguistic label. Each model is an action and enumerated. Linguistic labels are for the purpose of communication. For the purpose of understanding human activities and intention, for example, it is sufficient to know that Action 1 usually takes place in the morning, at the bed and that it usually leads to Action 2. It does not matter whether Action 1 is being labeled as "wake up"or "okimasu (wake up in Japanese language)", it does not matter if Action 2 is being labeled as "brush teeth"or "hamigaki (brush teeth in Japanese language)".

6. The sensor used is only suitable for indoor. The target applications are in the normal homes.

7. Segmentation of continuous observation into action instances in real-time requires better strategy.

7.3 Research Outlook

Further research works are required to improve the robustness of the HaR, as well as to extend the recognition to different levels of activities. The incremental approach of clustering can be adapted to deal with problems other than human activity recognition. We identify the following potential extension to this dissertation:

- The learned action models do not have linguistic label. For the purpose of communication, the system can learn the linguistic labels through interaction with human. A suitable human-machine interaction interface can be developed to address this issue.

- Alternative forms of input, such as audio and contextual data, can be provided to enhance the discriminative ability of the auto-HaR.
- Parallel auto-HaR frameworks can be developed to process the multimodal inputs.
- Segmentation of feature space can be used to enable the action model to be constructed in modular forms, e.g. action of hands plus action of legs.
- Hierarchical based model can be developed to infer high level activities from the actions.
- The sliding window sampling approach to segment input stream into action instances requires better or alternative strategy. The proposed framework uses sliding window sampling to segment the data from the continuous observation into action instances of two second windows. When the sampling was performed on the dataset, there were minimal peripheral movements due to transition from one action to another. In real-time scenario, there are significant amount of peripheral movements especially at the beginning and ending of an action, and in transitions from one action to another.
- The incremental approach of clustering can be applied to dynamic data and deal with data mining problems.

Bibliography

[1] K. Yamazaki, R. Ueda, S. Nozawa, M. Kojima, K. Okada, K. Matsumoto, M. Ishikawa, I. Shimoyama, and M. Inaba, "Home-assistant robot for an aging society," *Proceedings of the IEEE*, vol. 100, no. 8, pp. 2429–2441, 2012.

[2] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *International Journal of Computer Vision*, vol. 79, no. 3, pp. 299–318, 2008.

[3] Microsoft, "Kinect for windows sensor components and specifications," http://msdn.microsoft.com/en-us/library/jj131033.aspx, online accessed on 2014-05-30.

[4] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Unstructured human activity detection from rgbd images," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2012, pp. 842–849.

[5] J. Aggarwal and M. S. Ryoo, "Human activity analysis: A review," *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, p. 16, 2011.

[6] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer vision and image understanding*, vol. 104, no. 2, pp. 90–126, 2006.

[7] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine recognition of human activities: A survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1473–1488, 2008.

[8] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 6, pp. 790–808, 2012.

[9] R. Poppe, "A survey on vision-based human action recognition," *Image and vision computing*, vol. 28, no. 6, pp. 976–990, 2010.

[10] Z. Zhang, "Microsoft kinect sensor and its effect," *MultiMedia, IEEE*, vol. 19, no. 2, pp. 4–10, 2012.

[11] L. Xia, C.-C. Chen, and J. Aggarwal, "Human detection using depth information by kinect," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2011, pp. 15–22.

[12] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, "Real-time human pose recognition in parts from single depth images," *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[13] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Human activity detection from rgbd images," in *AAAI workshop on Plan, Activity, and Intent Recognition*, 2011.

[14] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3d points," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2010, pp. 9–14.

[15] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012, pp. 1290–1297.

[16] L. Chen, H. Wei, and J. Ferryman, "Readingact rgb-d action dataset and human action recognition from local features," *Pattern Recognition Letters*, 2013.

[17] E. Kim, S. Helal, and D. Cook, "Human activity recognition and pattern discovery," *IEEE Pervasive Computing*, vol. 9, no. 1, pp. 48–53, 2010.

[18] D. Wyatt, M. Philipose, and T. Choudhury, "Unsupervised activity recognition using automatically mined common sense," in *AAAI*, vol. 5, 2005, pp. 21–27.

[19] T. Huynh, M. Fritz, and B. Schiele, "Discovery of activity patterns using topic models," in *10th International Conference on Ubiquitous computing*. ACM, 2008, pp. 10–19.

[20] M. Stikic, D. Larlus, S. Ebert, and B. Schiele, "Weakly supervised recognition of daily life activities with wearable sensors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2521–2537, 2011.

[21] R. Hammid, S. Maddi, A. Johnson, A. Bobick, I. Essa, and C. L. Isbell, "Unsupervised activity discovery and characterization from event-streams," *arXiv preprint arXiv:1207.1381*, 2012.

[22] B. Chikhaoui, S. Wang, and H. Pigot, "Adr-splda: Activity discovery and recognition by combining sequential patterns and latent dirichlet allocation," *Pervasive and Mobile Computing*, 2012.

[23] P. Cui, F. Wang, L.-F. Sun, J.-W. Zhang, and S.-Q. Yang, "A matrix-based approach to unsupervised human action categorization," *IEEE Transactions on Multimedia*, vol. 14, no. 1, pp. 102–110, 2012.

[24] M. Charikar, C. Chekuri, T. Feder, and R. Motwani, "Incremental clustering and dynamic information retrieval," in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*. ACM, 1997, pp. 626–635.

[25] M. Ester, H.-P. Kriegel, J. Sander, M. Wimmer, and X. Xu, "Incremental clustering for mining in a data warehousing environment," in *VLDB*, vol. 98, 1998, pp. 323–333.

[26] J. Lin, M. Vlachos, E. Keogh, and D. Gunopulos, "Iterative incremental clustering of time series," in *Advances in Database Technology-EDBT 2004*. Springer, 2004, pp. 106–122.

[27] Z. Li, J.-G. Lee, X. Li, and J. Han, "Incremental clustering for trajectories," in *Database Systems for Advanced Applications*. Springer, 2010, pp. 32–46.

[28] S. Young, I. Arel, T. P. Karnowski, and D. Rose, "A fast and stable incremental clustering algorithm," in *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*. IEEE, 2010, pp. 204–209.

[29] M. Halkidi, M. Spiliopoulou, and A. Pavlou, "A semi-supervised incremental clustering algorithm for streaming data," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2012, pp. 578–590.

[30] OpenNI., "OpenNI | The standard framework for 3D sensing," http://openni.org/, 2010, [Accessed: 2012-04-30].

[31] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping multidimensional data*. Springer, 2006, pp. 25–71.

[32] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 281-297. California, USA, 1967, p. 14.

[33] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 8, pp. 1226–1238, 2005.

[34] B. Mackenzie, "Range of Movement (ROM)," http://www.brianmac.co.uk/musrom.htm, 2010, [Accessed: 2012-06-29].

[35] V. M. Zatsiorsky, *Kinematics of human motion*. Human Kinetics, 1998.

[36] K. Schindler and L. Van Gool, "Action snippets: How many frames does human action recognition require?" in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2008, pp. 1–8.

[37] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1650–1654, 2002.

[38] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics-theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[39] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 2, pp. 224–227, 1979.

[40] J. A. Hartigan, *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[41] W. J. Krzanowski and Y. Lai, "A criterion for determining the number of groups in a data set using sum-of-squares clustering," *Biometrics*, pp. 23–34, 1988.

[42] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[43] K. V. Mardia, J. T. Kent, and J. M. Bibby, "Multivariate analysis," p. 365, 1980.

[44] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[45] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using hidden markov model," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 1992, pp. 379–385.

[46] L. Xia, C.-C. Chen, and J. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2012, pp. 20–27.

[47] K. P. Murphy, "Markov and hidden markov models," in *Machine learning: a probabilistic perspective*. The MIT Press, 2012, pp. 589–630.

[48] "Processing 2," http:///processing.org/.

[49] "Gnu octave," http://www.gnu.org/software/octave/.

Publications

Refereed Journal Papers

1. W.H. Ong, L. Palafox, and T. Koseki, “An Unsupervised Approach for Human Activity Detection and Recognition”, International Journal of Simulation Systems, Science and Technology (IJSSST), Vol.14, No.5, 2013

2. W.H. Ong, L. Palafox, and T. Koseki, “An Incremental Approach of Clustering for Human Activity Discovery”, IEEJ Transactions on Electronics, Information and Systems, Vol. 134, No.11, 2014 (to appear)

3. W.H. Ong, L. Palafox, and T. Koseki, “Autonomous Learning and Recognition of Human Action based on An Incremental Approach of Clustering”, IEEJ Transactions on Electronics, Information and Systems (under review)

Refereed Conference Papers

1. W.H. Ong, L. Palafox, and T. Koseki, "Unsupervised Human Activity Detection with Skeleton Data from RGB-D Sensor," in Proceedings of The Fifth International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN), pp. 30-35, 5-7 June 2013

2. W.H. Ong, L. Palafox, and T. Koseki, "Investigation of Cluster Validity Indices for Unsupervised Human Activity Discovery," in Proceedings of The 2013 International Conference

on Artificial Intelligence (ICAI'13), Vol.1, pp. 315-321, 22-25 July 2013

Other Publications

1. W.H. Ong, L. Palafox, and T. Koseki, "Investigation of Feature Extraction for Unsupervised Learning in Human Activity Detection,"Bulletin of Networking, Computing, Systems, and Software, North America, 2, Jan. 2013 (presented at The Second International Workshop on Networking, Computing, Systems, and Software, Okinawa, Japan in Dec 2012)

2. W.H. Ong, and T. Koseki, "Unsupervised Activity Detection Based On Human Range of Motion Features,"Seoul National University-University of Tokyo (SNU-UT) Joint Seminar, Mar. 2013

Appendix A

Datasets

We recorded our example observations of a single subject with sixteen actions and we refer this set of observations as KOS-H16. We have also used the example observations of different actions of four subjects from Cornell Activity Dataset, CAD-60 [13]. We refer the datasets for the four subjects in CAD-60 as Person 1 (P1), Person 2 (P2), Person 3 (P3) and Person 4 (P4). We refer the datasets for the single subject in KOS-H16 as Person 5 (P5).

From the two set of example observations, we sampled 80 instances of each action for each subject using sliding window of two seconds. We sampled 160 instances from the random actions for each subject. The samples are split into learning and test set at 7:3 ratio. Fifty-six (56) instances are used in the discovery and model learning, i.e., the learning set. Twenty-four (24) instances are used as unseen observations to test the performance of the learned models, i.e., the test set. For the random actions, the learning set has 112 instances and the test set has 48 instances for each subject. Therefore, for each subject, we have a learning set of 56 instances of 9 actions in addition to 112 instances of random movements. The instances from different actions are mixed together and the framework does not know the labels. This gives a total of $56 \times 9 + 112 = 616$ instances in the learning set.

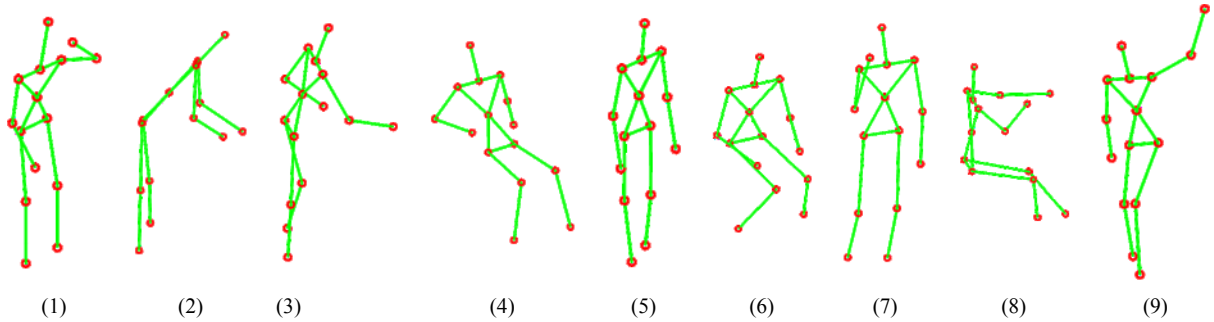


Figure A.1: Snapshots of one random frame of the skeleton of the nine actions by Person 1 to 4 (1) brushing teeth, (2) cooking (chopping), (3) cooking (stirring), (4) relaxing on couch, (5) still (standing), (6) talking on couch, (7) talking on phone, (8) working on computer, (9) writing on whiteboard.

A.1 Cornell Action Dataset CAD-60

CAD-60 consists of 12 daily activities: rinsing mouth, brushing teeth, wearing contact lens, talking on phone, drinking water, opening pill container, cooking (chopping), cooking (stirring), talking on couch, relaxing on couch, writing on whiteboard and working on computer. The data was collected from four subjects: two males (we refer them as Person 1 and Person 4) and two females (we refer them as Person 2 and Person 3). One of the females is left-handed (Person 3). Still (standing) and random activity samples by each subject are also included in the dataset. All of the data were collected in a normal household setting with no occlusion of body from the view of sensor. The CAD-60 dataset contains RGB images, depth images and 3-D coordinates of fifteen joint positions (skeleton data) for each frame. We use only the skeleton data to obtain the required features as described in Section 4.2. Including still, CAD-60 dataset has 13 activities in total. Among them, four of the activities have less than 10 examples per subject and are insufficient for the discovery phase to find. We have used the remaining 9 activities and the random activities to test our framework. These activities are atomic, i.e., actions in our context, and are listed in Table A.1.

Table A.1: List of actions

1.	A1	Brushing teeth
2.	A2	Cooking (chopping)
3.	A3	Cooking (stirring)
4.	A4	Relaxing on couch
5.	A5	Still (standing)
6.	A6	Talking on couch (sitting)
7.	A7	Talking on the phone
8.	A8	Working on computer
9.	A9	Writing on whiteboard
10.	RA	Random

A.2 The Sixteen Actions KOS-H16

KOS-H16 dataset has a single subject referred as Person 5 (P5). The dataset contains actions with significant movements, e.g., waving hand. There are sixteen actions in this dataset, as illustrated in Fig. A.2 and listed in Table A.2. “sit”refers to the stationary state of sitting on the chair. “stand” refers to the stationary state of standing on feet. “sit down”refers to the transition action from stand to sit, whereas “stand up”refers to the transition from sit to stand. There are three waving gestures. “wave bye”is the gesture of waving sideway. “wave come”is the gesture of waving the hand inward from high to low position. “wave go”is the gesture of waving the hand outward from low to high position. “(left)”indicates the activity being carried out with left hand, whereas “(right)”indicates a right handed activity.

Table A.2: List of actions by Person 5

1.	B1	Bowing	10.	B10	Walking
2.	B2	Drinking (left)	11.	B11	Wave bye (left)
3.	B3	Drinking (right)	12.	B12	Wave bye (right)
4.	B4	Sit	13.	B13	Wave come (left)
5.	B5	Sit down	14.	B14	Wave come (right)
6.	B6	Stand	15.	B15	Wave go (left)
7.	B7	Stand up	16.	B16	Wave go (right)
8.	B8	Talking on phone (left)	17.	RA	Random
9.	B9	Talking on phone (right)			

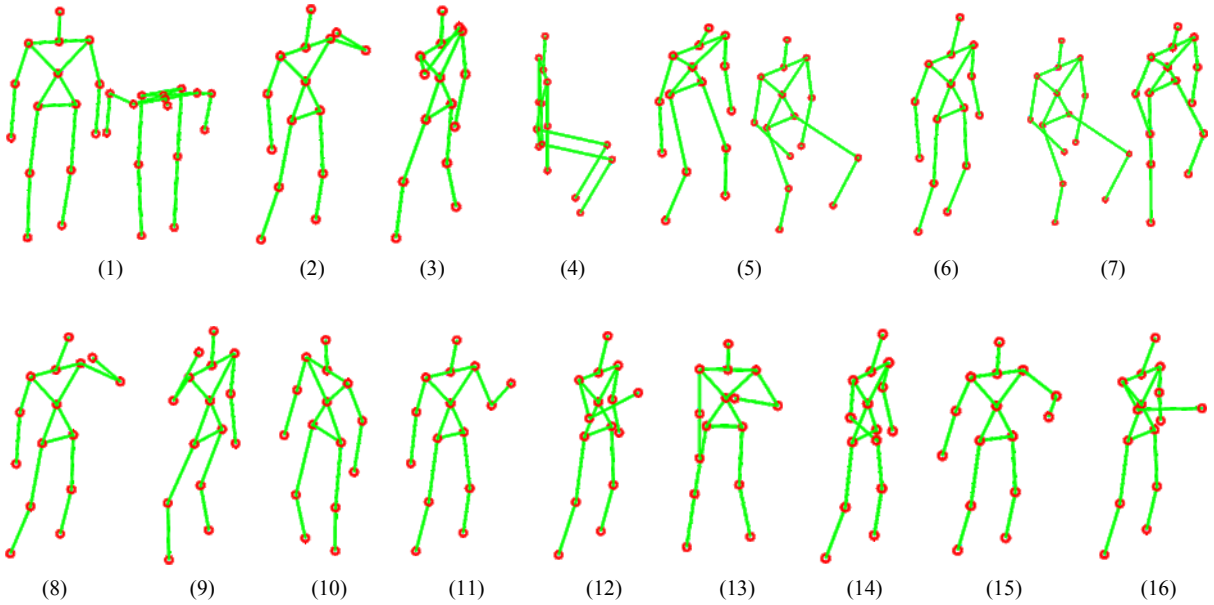


Figure A.2: Snapshots of one* random frame of the skeleton of the sixteen actions by Person 5 (1) bowing, (2) drinking (left), (3) drinking (right), (4) sit, (5) sit down, (6) stand, (7) stand up, (8) talking on phone (left), (9) talking on phone (right), (10) walking, (11) wave bye (left), (12) wave bye (right), (13) wave come (left), (14) wave come (right), (15) wave go (left), (16) wave go (right). *Two frames from the beginning and end of an observation are taken for actions (1), (5) and (7).