2013 16th International Conference on Advanced Robotics (ICAR)

Edificio Polifuncional "José Luis Massera", Universidad de la República, Montevideo, Uruguay

November 25th to 29th, 2013

ISBN: 978-1-4799-2722-7
Catalog Num.: CFP13ROB-ART

Recognition of Arm Activities Based on Hidden Markov Models for Natural Interaction with Service Robots

Jose I. Figueroa-Angulo*, Jesus Savage-Carmona*, Ernesto Bribiesca-Correa*,
Boris Escalante*, Ronald S. Leder*, Luis E. Sucar†
* Universidad Nacional Autonoma de Mexico.
† Instituto Nacional de Astrofisica, Optica y Electronica.

***Abstract*—This research presents a novel way of representing human motion and recognizing human activities from the skeleton output computed from RGB-D data from vision–based motion capture systems. The method uses a representation of the skeleton which is invariant to rotation and translation, based on Orthogonal Direction Change Chain Codes, as observations for a single Discrete Connected Hidden Markov Model formed by a set of multiple Hidden Markov Models for simple activities, which are merged using a grammar-based structure. The purpose of this research is to provide a service robot with the capability of human activity awareness, which can be used for action planning with implicit and indirect Human–Robot Interaction.** [1]

[2] ***Keywords*—Hidden Markov Models, Activity Recognition, Motion Recognition, Human–Machine Interaction, Pattern Recognition, Machine Learning, Viterbi Path***

I. Introduction

In daily life, human beings perform activities to accomplish diverse tasks at different times throughout the day. These activities are by one or several simpler activities which are performed at different times, and these simple activities have a chronological relationship to each other.

For human activity recognition, there are two techniques for recognizing activities from movement: Template Matching and State–Space [1]. In the Template Matching techniques, the image sequence is converted in a static shape pattern and is compared against a set of reference patterns; it is computationally inexpensive, but it is more sensitive to the variance of the movement duration. On the other hand, the State–Space techniques define a model where each static posture is a state and the states describing several poses are connected by probabilities of transition, and any motion is considered as a graph tour through various states of static poses.

A. *Space–State Human Activity Recognition*

Space–State models have been used widely to predict, estimate, and detect time series over a long period of time. One representative model is the Hidden Markov Model [2], which is a probabilistic technique to study discrete time series, which is recently being used for recognizing human motion. There are two approaches for recognizing human motion and activities using Hidden Markov Models, the first approach, and the most used, is the Maximum Likelihood Probability, where the likelihood probabilities of the elements of a set of isolated Hidden Markov Models are computed from a sequence of observations, and the motion or activity is selected from the Model with the highest likelihood [2]. The other approach is Viterbi Path Labelling, which is based on the computation of the Viterbi Path on a single Hidden Markov Model where a single state or a subset of linked states represent a motion or activity, the sequence of states in the Viterbi path indicates which motion or activity is being performed in a period of time, according to a sequence of observations.

B. *Human Activity Recognition Based on Maximum Likelihood Probability*

Several architectures and types of HMMs have been used for activity recognition from Maximum Likelihood Probability, such as Conditional HMMs, Ergodic HMMs, Linear HMMs, and Maximum Entropy Markov Models. *Conditional HMMs* have been used for human activity recognition and human–object interaction using the skeleton computed from the depth data and the image data of a Microsoft Kinect sensor [3]. *Ergodic HMMs* have been used for recognition of actions which can be used for interaction with video games from a Spherical Histogram for the joints of the skeleton computed from the depth data of a Microsoft Kinect sensor [4]. *Linear HMMs* have been used for recognition of leg motion and hand gestures from labeled body parts from depth data acquired with a PrimeSense camera [5]; recognition of actions which can be used for interaction with video games from a Bag of 3-D Points from the depth data of a Microsoft Kinect sensor [6]; classification of golf swings from the skeleton computed from the depth map of Microsoft Kinect sensor [7]; *Maximum Entropy Markov Models* detection of human activity both in a structured fashion, as well as in an unstructured fashion, using geometrical and location information from Skeleton Joints, and Histograms of Oriented Gradients from image and depth data of a Microsoft Kinect sensor [8], [9].

C. *Human Activity Recognition Based on Viterbi Path Labelling*

Activity recognition with Viterbi Path Labelling has been applied with several architectures and types of HMMs, such as Coupled HMMs, Ergodic HMMs, and Connected Linear

[1]ACKNOWLEDGMENT: This work was supported by PAPIIT-DGAPA UNAM under Grant IN-107609
[2]978-1-4799-2722-7/13/$31.00 ©2013 IEEE

HMMs. *Coupled HMMs* have been used for health monitoring from interactions of respiration and brain activity [10]. *Ergodic HMMs* have been used for recognition of actions on video sequences from Residual Motion Vectors in images sequences [11]; recognition of gestures of the upper body from Pictorial Structures on segmented images [12]; detection of fence climbing on surveillance video using the Star Skeleton of a segmented image blob [13]; modelling of individual and group interactions from the combination of multiple audio and video sources [14]. *Connected Linear HMMs* have been used for transcription of motion from a set of Distributed Body Sensors [15]; recognition of human activity from the location of the hands and the head captured with a Stereo Camera [16]; gait detection and discrimination between walk and jogging activities by analysing the output of a set of body–worn Inertial Motion Units [17]; recognition of human behaviour from image and skeleton from depth data, captured with a Microsoft Kinect sensor [18].

D. Data Sets

The acquisition of skeleton data from depth images is the most recent approach for marker-less motion capture, it is becoming widespread among the human activity research community because depth sensors, as the Microsoft Kinect sensor, have become more affordable and widely available. As the skeleton captured by these depth sensors has a different format than the most widely used motion capture databases, such as CMU-MMAC [19], CMU Motion Capture Database [20], HDM05 [21], KUG Data Base [22] and TUM Kitchen Data Set [23]; the human activity research community has been creating datasets based on depth sensor data (Table I), using the work of James Shotton [24] on real time human pose recognition, and depth sensors based on PrimeSense technology, such as the Microsoft Kinect sensor.

II. Proposed Approach

In this research, the MSR Daily Activity 3D dataset is used for training and testing the activity recognition system. In the training stage, the code book of key frames is built from the Orthogonal Direction Change Chain Codes of the clustered skeletons of each activity, and the Discrete Connected Hidden Markov Model is built from the Linear Hidden Markov Models which have the largest likelihood probability for each activity. In the testing stage, a sequence of observations symbols is computed from a sequence of Orthogonal Direction Change Chain Codes, using techniques of fuzzy string search on the codebook of key frames; this sequence of observation symbols is used as input to the Discrete Connected Hidden Markov Model to compute the most likely sequence of hidden states, which indicates activities which are being performed during the motion capture.

Dataset		Depth	Color	Skeleton
CAD-60	[8]	●	●	●
G3D	[25]	●	●	●
Kinect+SR400	[26]	●	●	
LIRIS	[27]	●	●	
MSR Action 3D	[6]	●	●	●
MSRDailyActivity3D	[28]	●	●	●
MSRC-12	[29]	●	●	●
RGBD-HuDaAct	[30]	●	●	●
UCF Kinect	[31]	●	●	

TABLE I: Data Sets Based on PrimeSense technology Depth Data

A. Orthogonal Direction Change Chain Code

The digitization stage is based on the Orthogonal Direction Change Chain Code [32], which digitizes three-dimensional curves into a set of codes which represent orthogonal direction changes between three constant length segments of a three-dimensional curve $(\vec{u}, \vec{v}, \vec{w})$ (Equation 1), which are aligned to the corners of a three-dimensional grid with constant-sized cells.

As the orthogonal direction changes are relative, these Chain Codes have some interesting properties: invariance to translation, invariance to rotation, invariance to mirroring and invariance to starting point. The invariance to rotation and translation allow to represent a large set of curves generated by absolute direction changes, such as orthogonal 3-D Freeman codes, using only one Chain Code [33].

There are five different orthogonal direction changes for representing any three-dimensional curve (Figure 1), as explained in the work of Bribiesca [32]:

- The Chain Element "0" represents a direction change which *goes straight* through the contiguous straight-line segments following the direction of the last segment.
- The Chain Element "1" represents a direction change to the *right*.
- The Chain Element "2" represents a direction change *upward* (stair-case fashion).
- The Chain Element "3" represents a direction change to the *left*.
- The Chain Element "4" represents a direction change which is *going back*.

$$chain\,element(\vec{u}, \vec{v}, \vec{w}) = \begin{cases} 0, & if\,\vec{w} = \vec{v}; \\ 1, & if\,\vec{w} = \vec{u} \times \vec{v}; \\ 2, & if\,\vec{w} = \vec{u}; \\ 3, & if\,\vec{w} = -(\vec{u} \times \vec{v}); \\ 4, & if\,\vec{w} = -\vec{u} \end{cases} \tag{1}$$

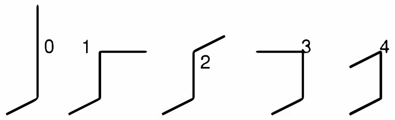


Fig. 1: Orthogonal Direction Change Chain Elements

1) Digitization of a Three-Dimensional Curve: In order to convert a set of three-dimensional lines into a set of line segments of constant length, the first step consists in aligning the vertices, $\vec{p} \in \mathbb{R}^3$ and $\vec{q} \in \mathbb{R}^3$, to the corners of the three-dimensional grid, by rounding the values of $\vec{p}$ and $\vec{q}$, according to the smallest distance on each axis between the vertex and the neighbour values on the grid, $\vec{g_i}$ and $\vec{g_j}$, obtaining the vertices $\vec{p'}$ and $\vec{q'}$ (Equation 2).

$$\vec{g_i} = \{\text{floor}\left(\vec{v}_x\right), \text{floor}\left(\vec{v}_y\right), \text{floor}\left(\vec{v}_z\right)\} \tag{2a}$$

$$\vec{g_j} = \{\text{ceil}\left(\vec{v}_x\right), \text{ceil}\left(\vec{v}_y\right), \text{ceil}\left(\vec{v}_z\right)\} \tag{2b}$$

$$\vec{v'}_x = \begin{cases} \vec{g_i}_x, & if\ \vec{g_i}_x \le \vec{v}_x < \frac{\vec{g_i}_x + \vec{g_j}_x}{2} < \vec{g_j}_x; \\ \vec{g_j}_x, & \text{otherwise} \end{cases} \tag{2c}$$

$$\vec{v'}_y = \begin{cases} \vec{g_i}_y, & if\ \vec{g_i}_y \le \vec{v}_y < \frac{\vec{g_i}_y + \vec{g_j}_y}{2} < \vec{g_j}_y; \\ \vec{g_j}_y, & \text{otherwise} \end{cases} \tag{2d}$$

$$\vec{v'}_z = \begin{cases} \vec{g_i}_z, & if\ \vec{g_i}_z \le \vec{v}_z < \frac{\vec{g_i}_z + \vec{g_j}_z}{2} < \vec{g_j}_z; \\ \vec{g_j}_z, & \text{otherwise} \end{cases} \tag{2e}$$

When the distance between two vertices on the three-dimensional grid is longer than the size of the cell, additional vertices on each axis are added to the line from the components of the Manhattan distance of $\vec{p'}$ and $\vec{q'}$, where $d(\vec{p'}, \vec{q'}) = \left|\vec{p'}_x - \vec{q'}_x\right| + \left|\vec{p'}_y - \vec{q'}_y\right| + \left|\vec{p'}_z - \vec{q'}_z\right|$ [34], to compute a set of constant-length line segments between two vertices.

Finally, the Chain Codes are computed by taking three consecutive line segments, starting from the first line segment, and applying the rules of orthogonal direction changes to compute the corresponding chain element (Figure 2).

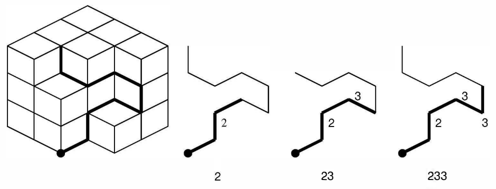


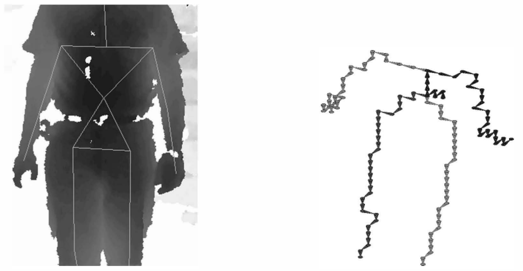
Fig. 2: Example of Chain Code Sequence

B. Digitization of Three-dimensional joint data

The three-dimensional joint data is captured by a three-dimensional vision system, such as the Microsoft Kinect, which acquires the joint data by analysing the depth map captured by the sensor. The skeleton data is digitized to Chain Codes (Figure 3) for generating a set of key frames which represent the motion of the arms.

There are a couple of factors which have significant impact in the digitization of the skeleton data to Chain Codes: the noise of the sensor and the angle of orientation of the body. The former affects the length of each limb, resulting in Chain Codes of variable length; while the latter affects the proportion of orthogonal segments along a Chain Code, which has negative effects in the algorithms which match Chain Codes.

1) Length of the Parts of the Body: The length of the body parts of the skeleton which are captured by the Microsoft Kinect sensor experiment variation in their measures either by noise on the sensor of the camera, loose clothing on the subject who is being recorded or limitations on the precision of the algorithm which computes the location of each joint in the skeleton, in order to keep a fast capture rate. When this noisy data is converted to a set of Chain Codes, the length each element of the set varies according to the length of the corresponding body part.



(a) Depth Map Skeleton (b) Digitized Skeleton (42mm Segments)

Fig. 3: Digitization of a Depth Map Skeleton

In order to correct those variations, from the original skeleton is built a skeleton made of unitary vectors and is scaled by a reference length which in this case is equal to 170 mm, the length of the head and neck joints of the skeleton which is captured by the Microsoft Kinect sensor and processed by OpenNI.

2) Orientation of the Body: An issue with the matching of Chain Codes from skeletons captured by three-dimensional vision sensors is the orientation angle of the torso of the subject: the visual analysis of skeletons made of orthogonal direction vectors shows that the same pose has varying proportions of orthogonal direction vectors, according to the orientation angle of the subject to the camera, which changes the apparent pose of each limb [35].

In order to avoid that issue, the orientation of the skeleton is normalized by applying two rotations to the skeleton, the first rotation is computed using the normal vector $\vec{N}$ of the triangle formed by the joints of the torso $\vec{p_1}$, the left shoulder $\vec{p_2}$ and the right shoulder $\vec{p_3}$ which is aligned to the axis $\vec{Z}$ of the camera by computing the rotation matrix $\mathbf{R}$ which transforms the normal vector $\vec{N}$ into the vector $\vec{Z}$, using the Rodrigues rotation formula (Equation 16) [36]. The whole skeleton $\mathbf{S}$ is rotated by the matrix $\mathbf{R}$ to get a skeleton, $\mathbf{S_Z}$, which is aligned to the the axis $\vec{Z}$ of the camera (Equation 17).

Once the normal vector of the body has been aligned towards the line of vision of the camera, the second rotation is computed using the up vector of the body, which is computed from the up vector $\vec{U} = \vec{T_L} + \vec{T_R}$, where $\vec{T_L} = \vec{p_1} + \vec{p_2}$ and $\vec{T_R} = \vec{p_1} + \vec{p_3}$, and $\vec{p_1}, \vec{p_2}, \vec{p_3}$ are the joints of the torso, the left shoulder and the right shoulder; the up vector $\vec{U}$ is aligned to the vertical axis of the world $\vec{Y}$, by computing the rotation matrix $\mathbf{R}$ which transforms the up vector $\vec{U}$ into the vector $\vec{Y}$, using the Rodrigues rotation formula (Equation 30). The whole skeleton $\mathbf{S_Z}$ is rotated by the matrix $\mathbf{R}$ to get a skeleton, $\mathbf{S_{ZY}}$, which is aligned to the vertical axis of the world (Equation 31).This skeleton $\mathbf{S_{ZY}}$ is digitized to get a set of Chain Codes which describe each limb and analyse motion using this data.

$$\vec{j} = (x, y, z) \tag{3}$$
$$\mathbf{S} = \{j_1, j_2, j_3 \cdots j_{13}, j_{14}, j_{15}\} \tag{4}$$
$$\vec{p_1} = S_{j_{\mathbf{9}}} \tag{5}$$
$$\vec{p_2} = S_{j_3} \tag{6}$$
$$\vec{p_3} = S_{j_6} \tag{7}$$
$$\vec{T_L} = \vec{p_2} - \vec{p_1} \tag{8}$$
$$\vec{T_R} = \vec{p_3} - \vec{p_1} \tag{9}$$
$$\vec{T_N} = \vec{T_L} \times \vec{T_R} \tag{10}$$
$$\vec{Z} = (0, 0, 1) \tag{11}$$
$$\theta = \cos^{-1}\left(\vec{T_N} \cdot \vec{Z}\right) \tag{12}$$
$$\vec{r} = \frac{\vec{Z} \times \vec{T_N}}{\|\vec{Z} \times \vec{T_N}\|} \tag{13}$$
$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{14}$$
$$\mathbf{N}(\vec{r}) = \begin{bmatrix} 0 & -\vec{r_z} & \vec{r_y} \\ \vec{r_z} & 0 & -\vec{r_x} \\ -\vec{r_y} & \vec{r_x} & 0 \end{bmatrix} \tag{15}$$
$$\mathbf{R} = \mathbf{I} + \sin\theta * \mathbf{N}(\vec{r}) + (1 - \cos\theta) * \mathbf{N}(\vec{r})^2 \tag{16}$$
$$\mathbf{S_Z} = \mathbf{S} * \mathbf{R} \tag{17}$$
$$\mathbf{S_Z} = (j_1, j_2, j_3 \cdots j_{13}, j_{14}, j_{15}) \tag{18}$$
$$\vec{p_1} = S_{Z_{j_{\mathbf{9}}}} \tag{19}$$
$$\vec{p_2} = S_{Z_{j_3}} \tag{20}$$
$$\vec{p_3} = S_{Z_{j_6}} \tag{21}$$
$$\vec{T_L} = \vec{p_2} - \vec{p_1} \tag{22}$$
$$\vec{T_R} = \vec{p_3} - \vec{p_1} \tag{23}$$
$$\vec{T_U} = \vec{T_L} + \vec{T_R} \tag{24}$$
$$\vec{Y} = (0, 1, 0) \tag{25}$$
$$\theta = \cos^{-1}\left(\vec{T_U} \cdot \vec{Y}\right) \tag{26}$$
$$\vec{r} = \frac{\vec{Y} \times \vec{T_U}}{\|\vec{Y} \times \vec{T_U}\|} \tag{27}$$
$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{28}$$
$$\mathbf{N}(\vec{r}) = \begin{bmatrix} 0 & -\vec{r_z} & \vec{r_y} \\ \vec{r_z} & 0 & -\vec{r_x} \\ -\vec{r_y} & \vec{r_x} & 0 \end{bmatrix} \tag{29}$$
$$\mathbf{R} = \mathbf{I} + \sin\theta * \mathbf{N}(\vec{r}) + (1 - \cos\theta) * \mathbf{N}(\vec{r})^2 \tag{30}$$
$$\mathbf{S_{ZY}} = \mathbf{S_Z} * \mathbf{R} \tag{31}$$

To eliminate the ambiguity which is posed by the properties of invariance of the Orthogonal Direction Change Chain Codes, before the computation of the Chain Codes from the set of constant length segments, three orthogonal segments are appended at each end (Table II). These orthogonal segments indicate the position of each limb at the ends (hands or feet), relative to the joint at the middle of the joint sequence (neck or hip).

Chain Codes	x	y	z	Chain Codes	x	y	z
211	$+$	$-$	$-$	433	$+$	$+$	$-$
213	$+$	$-$	$+$	431	$+$	$+$	$+$
233	$-$	$-$	$-$	411	$-$	$+$	$-$
231	$-$	$-$	$+$	413	$-$	$+$	$+$

TABLE II: Relative Position Chain Codes

These Chain Codes work on the premise that the orthogonal segments which are used to generate the digitized curve from the Orthogonal Direction Change Chain Codes have the values of $\{(0, -1, 0), (0, 0, -1)\}$, for the left end, and the values of $\{(0, 1, 0), (0, 0, 1)\}$, for the right end.

A benefit of these additional Chain Codes is that if the set of reference Chain Codes is organized in a prefix tree, the lookup of Chain Codes is directed to the branches which have the same prefix, resulting in a reduction of the lookup time.

C. Motion Analysis

The skeleton by itself is useful for analysing motion, since each limb has 3 degrees of freedom, which accounts for a large number of combinations of motions and angles which can be used to extract relevant information about how a limb is moving.

1) Splitting the skeleton: To make easier the analysis of the motion and as the subjects of interest can be the arms or legs, the skeleton is split into upper limbs and lower sections, according to the Table III. The purpose of those joint sequences is for computing a Chain Code which covers both sides of each section in a consecutive manner.

	Joint Sequence
Upper Section	Left Hand, Left Elbow, Left Shoulder, Right Shoulder, Right Elbow, Right Hand
Lower Section	Left Foot, Left Knee, Left Hip, Right Hip, Right Knee, Right Foot

TABLE III: Joints of upper and lower sections

2) Fast Levenshtein Distance: One way of measuring the similarity between two strings of characters is by computing the amount of single character edits which are required to change a string into the other. This measure, also known as editing distance, can be computed by the Levenshtein function [37], which calculates the editing distance between two strings by counting the minimum number of insertions, deletions and substitutions between characters using dynamic programming techniques. Similar strings have a short Levenshtein distance between them, while the dissimilar strings have a long Levenshtein distance.

The Levenshtein function can be used to compute the editing distance between strings of any length, however the cost to compute it has order $\mathcal{O}(mn)$, where m, n are the lengths of the strings. And when the closest match of a string is searched in a set of strings, this cost is multiplied by the number of elements o in the set of strings, resulting in a total computing cost of $\mathcal{O}(mno)$.

One approach to reduce the computing cost of string is by arranging the strings in prefix order, allowing the Levenshtein distance table to be reused for similar strings as well as limiting the growth of the distance table by appending or removing rows at the bottom. A data structure which is very useful for such purpose is the *trie* or *prefix tree* [38], which is a n-ary tree where the position on the tree defines the associated key, all the descendent nodes share a prefix, and the leaves store the values.

The search of a string on the trie starts by initializing a global minimum cost, which is used as criteria to keep going deeper on the trie. The next step consists in computing the Levenshtein distance between the current character on the string and all the nodes on the next depth level, the global minimum cost is updated to the minimum Levenshtein distance computed previously. The updated global minimum cost is used as flag to keep searching on the internal nodes whose minimum costs are lesser than the global minimum cost. The search is repeated until there are no minimum Levenshtein distances which are smaller than the global minimum cost [39], [40].

D. Learning Model

The purpose of this work is to analyse human behaviour by recognizing the activities which are performed by a person. Human activity has the properties of being both complex and dynamic, since a person can be performing any action, which can be a pose or a motion, and suddenly change to another action.

1) Hidden Markov Models: The learning model for activity analysis is based on Hidden Markov Models, which are statistical Markov Models in which the signal or process to model is assumed to be a Markov Process with unobserved states [2]. In this research, the hidden variable is an activity which is being performed in a period of time, and the observed variable is a symbol from the code book of key frames. The sequence of key frame symbols of each limb (observations) is used as input, either as a training sample or to figure out which activity is performed from a set of observations. The Hidden Markov Model for a simple activity is trained using Viterbi Learning [41] on a set of sequences of observations of variable length, which represent repetitions of the same activity. The purpose of the training is to recognize different motions and poses which can be performed with the arms.

2) Connected Hidden Markov Model: The learning model proposed for this work is a large Hidden Markov Model which is formed by connecting of several small Hidden Markov Models [42], which can recognize a single activity (Figure 4), to a common initial state B_m and a common final state E_m. The common initial state B_m has equal emission probabilities for each symbol and equal transition probabilities to any of the initial states of the motion and pose recognition models, and the common final state E_m has a transition which returns to the common initial state B_m to restart the recognition process (Figure 5). This configuration allows to detect transitions between activities by returning to the common initial state B_m after a change in the sequence of observations which has low probability to be emitted in a certain set of states.

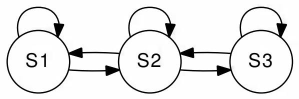


Fig. 4: Hidden Markov Model for Single Activity Recognition

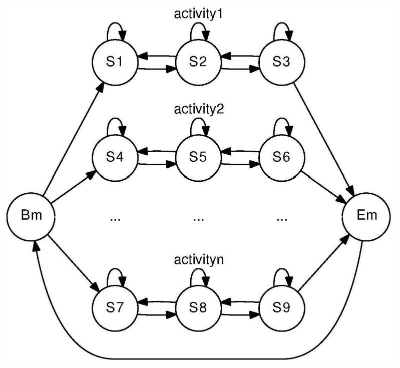


Fig. 5: Connected Hidden Markov Model for Continuous Activity Recognition

The first step for building the Connected Hidden Markov Model is to train every individual HMM for activity recognition with Viterbi Learning, using several combinations of states and a training set of motion sequences, which are performed by a group of persons. The Viterbi Learning algorithm is selected for training the Hidden Markov Models because the topology of each individual Hidden Markov Model is already defined and the computing of the transition probabilities is performed deterministically over the topology of the model, which ensures that all the connected states are not isolated.

The selection of the best HMM starts by computing the Likelihood Probability of a set of sequences of observations of every activity in the testing set, using a set of HMMs which recognize the same activity, and whose amount of states ranges from 3 to 16 states. At the end of this testing stage, the HMMs which have the Likelihood Probability for a target activity are selected for building the Connected Hidden Markov Model.

Once all the best individual Hidden Markov Models are selected, the construction of the Connected HMM starts by removing the transitions to the states B and E of each individual activity recognition model, the next step is to connect the common initial state B_m to the first state of each individual Hidden Markov Model, and the last step is to connect the final state of each individual motion/pose recognition model to the common final state E_m.

The activity of a limb is labelled by computing the optimal sequence of states ,$\vec{Q}$, using the Viterbi algorithm on a sequence of observations, $\vec{O}$, which is obtained by classifying the Chain Codes of digitized skeleton joints against the code book of key frames. The sequence of states goes from the state B_m, through all the states which belong to a certain activity, and the state E_m to return to the beginning of the Connected HMM, where depending on the changes on the sequence of observations $\vec{O}$, the sequence of states $\vec{Q}$ can go through the states which described the former motion or can go through the states of other activity.

III. Tests

The purpose of the tests is to prove that simple activities can be recognized using three dimensional joint data, digitized as Three-Dimensional Chain Codes, as input for a set of Hidden Markov Models which recognize motion as a sequence of discrete key frames.

A. Input Data

The tests were performed using the Microsoft Research Daily Activity 3D Data set ($\mathrm{MSRDaily}$) [28], which was captured by using a Microsoft Kinect device. The data set is composed by 16 activities, *a*) drink; *b*) eat; *c*) read book; *d*) call cellphone; *e*) write on a paper; *f*) use laptop; *g*) use vacuum cleaner; *h*) cheer up; *i*) remain still; *j*) toss paper; *k*) play game; *l*) lay down on sofa; *m*) walk; *n*) play guitar; *o*) stand up; and *p*) sit down which are performed by 10 persons, who execute each activity twice, once in standing position, and once in sitting position. There is a sofa in the scene. Three channels are recorded: depth maps (.bin), skeleton joint positions (.txt), and RGB video (.avi). There are $16 * 10 * 2 = 320$ files for each channel. The whole set is formed by $320 * 3 = 960$ files. For the purpose of this work, only the skeleton joint positions were used as input for the activity recognition system.

The training of the Hidden Markov Models for each activity in the $\mathrm{MSRDaily}$ data set was done by selecting the activities of the first 6 subjects, and a validation test was performed with this training set; the last 4 subjects were used as input for tests with unknown data. All the skeletons were normalized and oriented to the axes Y and Z, using the algorithms specified at Section II-B2.

For this work, a discrete Hidden Markov Model is used to recognize activities, therefore, a code book of symbols is needed as input for the model. The symbols are generated from a reference set of skeletons, which is computed by applying Linde-Buzo-Gray Vector Quantization [43] to the set of normalized skeletons. From this set of skeletons, a code book is generated for the key frames of the motion of the arms.

A control group of Hidden Markov Models for the activities of the arms is computed with Viterbi Training, using observations based on the average Euclidean distance between the joints of each skeleton of the training set and the joints of a code book of skeletons.

The activity recognition using Chain Codes is performed on skeletons which are digitized at a set of decreasing three-dimensional grid resolutions (17mm, 42mm, 68mm). For each resolution, the key frames are generated by digitizing the reference set of skeletons. For each activity, a Hidden Markov Model is computed with Viterbi Learning, using the with the observations based on the similarity measures between the Chain Codes of the training set and the Chain Codes of the code book of key frames.

For both groups, each activity is trained on a set of Hidden Markov Models with increasing amount of states, ranging from 3 to 16 states, which have the topology specified in the Section II-D2.

The first test is the Single Model Test, whose purpose is to find the amount of states where each Hidden Markov Model has the highest likelihood probability, to select the model which is more capable of recognizing an activity by using the testing set as input. From the results of this test, the Hidden Markov Models which have the highest likelihood probability are used to build the Connected Hidden Markov Models for Activity Labelling.

The second test is applied to the Connected Hidden Markov to find out if it is able to label a set of testing data, which is formed by a set of 10 routines, formed by activities of the $\mathrm{MSRDaily}$ data set, that are performed by each person of the testing set (Tables IV, V). The test is performed by computing the Viterbi Path of each input routine, the path is segmented using the lengths of each activity. Within each sub-path, the states which are consecutive or related are counted, and that count is divided by the length of the sub path to get a percentage of the amount of observations which are labelled correctly by a subset of the Hidden Markov Model, which ranges from $[0.0 \cdots 1.0]$.

Test 1		Test 2		Test 3	
activity	example	activity	example	activity	example
remain still	2	remain still	2	remain still	2
walk	1	sit down	2	walk	2
remain still	2	eat	1	lay down on sofa	1
sit down	2	remain still	1	sit down	2
remain still	1			call cellphone	1
				stand up	1
				call cellphone	2

Test 4		Test 5		
activity	example	activity	example	
remain still	2	remain still	2	
toss paper	2	sit down	2	
remain still	2	eat	2	
cheer up	2	drink	2	
walk	1	stand up	1	
		walk	1	

TABLE IV: Set of Test Activities for Viterbi Path Labelling Test (Standing Start)

Test 6		Test 7	
activity	example	activity	example
remain still	1	remain still	1
lay down on sofa	2	stand up	2
remain still	2	walk	2
stand up	1	sit down	1

Test 8		Test 9	
activity	example	activity	example
remain still	1	remain still	1
use vacuum cleaner	1	stand up	2
remain still	1	walk	2
use laptop	1	sit down	2
standup	2	remain still	1
walk	1		
sit down	2		
write	1		

Test 10		
activity	example	
remain still	1	
stand up	2	
walk	2	
sit down	2	
remain still	1	

TABLE V: Set of Test Activities for Viterbi Path Labelling Test (Sitting Start)

Regarding this test, two details must be pointed out. The first detail is that the activities of the MSRDaily data set start either from a standing position or a sitting position; and the other detail is that the routines are built by concatenating the motion data without any motion segmentation. Thus, for both cases, the Viterbi Path can show states which are not related to the activities indicated in the routine.

IV. Results

The results on the Activity Labelling Tests show that a Connected Hidden Markov Model which uses Orthogonal Direction Change Chain Codes as observations, has a slightly inferior accuracy at recognizing activities than a Control Connected Hidden Markov Model which uses Average Euclidean Distance of Joints as observations (Tables VI, VII, VIII, IX).

	Viterbi Path Labelling Accuracy(%)			
Routine	Test Subject 1	Test Subject 2	Test Subject 3	Test Subject 4
1	28.70%	1.30%	23.07%	42.59%
2	31.04%	12.65%	38.40%	50.10%
3	38.79%	24.79%	27.61%	23.93%
4	39.11%	3.30%	29.18%	32.86%
5	31.55%	8.83%	26.56%	33.78%
6	29.87%	10.30%	32.62%	35.02%
7	9.87%	13.95%	9.91%	30.71%
8	1.94%	6.98%	8.73%	25.14%
9	15.79%	13.30%	4.22%	65.86%
10	4.90%	7.67%	2.20%	52.18%

TABLE VI: Viterbi Path Labelling Accuracy (Arms, Normalized Skeleton, Euclidean Distance Classifier, 256 Symbols)

	Viterbi Path Labelling Accuracy(%)			
Routine	Test Subject 1	Test Subject 2	Test Subject 3	Test Subject 4
1	18.24%	4.36%	21.35%	20.22%
2	16.67%	9.31%	15.97%	10.18%
3	15.46%	18.70%	21.79%	19.26%
4	25.76%	14.34%	22.05%	20.23%
5	28.87%	12.58%	15.70%	22.76%
6	21.39%	8.66%	24.22%	4.63%
7	30.13%	11.82%	7.55%	27.66%
8	19.17%	8.65%	15.29%	21.17%
9	11.13%	6.76%	2.53%	5.15%
10	19.33%	11.50%	11.14%	16.89%

TABLE VII: Viterbi Path Labelling Accuracy (Arms, 17mm ODC3, Levenshtein Distance Classifier, 256 Symbols)

	Viterbi Path Labelling Accuracy(%)			
Routine	Test Subject 1	Test Subject 2	Test Subject 3	Test Subject 4
1	31.37%	2.24%	22.70%	14.81%
2	40.06%	3.34%	22.43%	16.17%
3	29.28%	25.96%	23.67%	13.25%
4	51.87%	19.54%	23.27%	27.98%
5	33.73%	22.37%	18.42%	22.88%
6	34.41%	3.58%	22.08%	7.24%
7	27.24%	8.72%	11.95%	17.98%
8	4.74%	6.32%	10.76%	10.98%
9	4.86%	1.39%	2.17%	12.94%
10	10.95%	11.50%	3.67%	9.67%

TABLE VIII: Viterbi Path Labelling Accuracy (Arms, 42mm ODC3, Levenshtein Distance Classifier, 256 Symbols)

	Viterbi Path Labelling Accuracy(%)			
Routine	Test Subject 1	Test Subject 2	Test Subject 3	Test Subject 4
1	14.46%	6.24%	6.99%	22.22%
2	5.96%	9.31%	6.08%	9.58%
3	18.66%	28.21%	18.34%	16.44%
4	27.05%	15.36%	15.70%	14.78%
5	37.50%	23.10%	10.86%	19.73%
6	22.34%	5.67%	17.24%	6.51%
7	27.50%	13.18%	20.13%	21.44%
8	13.57%	9.23%	11.98%	18.21%
9	4.35%	5.79%	1.45%	6.19%
10	25.00%	14.83%	23.90%	18.94%

TABLE IX: Viterbi Path Labelling Accuracy (Arms, 68mm ODC3, Levenshtein Distance Classifier, 256 Symbols)

V. Conclusions

In this research, it was presented the first steps in the development of a system for natural interaction with robots, by recognizing human activities from data of three-dimensional sensors, such as the Microsoft Kinect sensor, using Orthogonal Direction Change Chain Codes for digitization of three-dimensional joint data, and Hidden Markov Models for activity recognition. The Orthogonal Direction Change Chain Codes provide a way of digitizing joint data which is invariant to rotation, translation and mirroring, which simplifies the matching against a set of key frames, which represent positions in the motion range of a limb.

A Connected Hidden Markov Model to recognize activities with repetitive motion was proposed for motion recognition. The results of the tests showed that the Connected Hidden Markov Model is capable of recognizing activities in standing and sitting positions. The follow-up for this work is to integrate spatial information to the motion analysis to enhance the classification of motionless activities as well as performing a thorough research on techniques of fuzzy string search, to enhance the accuracy of the classification Chain Codes.

References

[1] J. Aggarwal and Q. Cai, “Human motion analysis: A review,” *Computer Vision and Image Understanding*, vol. 73, no. 3, pp. 428 – 440, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1077314298907445

[2] L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” in *Proceedings of the IEEE*, 1989, pp. 257–286.

[3] M. Glodek, F. Schwenker, and G. Palm, “Detecting actions by integrating sequential symbolic and sub-symbolic information in human activity recognition,” in *Proceedings of the 8th international conference on Machine Learning and Data Mining in Pattern Recognition*, ser. MLDM’12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 394–404. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31537-4_31

[4] L. Xia, C.-C. Chen, and J. Aggarwal, “View invariant human action recognition using histograms of 3d joints,” in *The 2nd International Workshop on Human Activity Understanding from 3D Data (HAU3D)*, 2012.

[5] A. Jalal, S. Lee, J. T. Kim, and T.-S. Kim, “Human activity recognition via the features of labeled depth body parts,” in *Proceedings of the 10th international smart homes and health telematics conference on Impact Ananlysis of Solutions for Chronic Disease Prevention and Management*, ser. ICOST’12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 246–249. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30779-9_36

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

[7] L. Zhang, J.-C. Hsieh, and J. Wang, “A kinect-based golf swing classification system using hmm and neuro-fuzzy,” in *Computer Science*

and Information Processing (CSIP), 2012 International Conference on, aug. 2012, pp. 1163 –1166.

[8] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Human activity detection from rgbd images," Carnegie Mellon University, Department of Computer Science, Cornell University, Ithaca, NY 14850, Tech. Rep., 2011.

[9] ——, "Unstructured human activity detection from rgbd images." in *ICRA*. IEEE, 2012, pp. 842–849. [Online]. Available: http://dblp.uni-trier.de/db/conf/icra/icra2012.html#SungPSS12

[10] I. Rezek, P. Sykacek, and S. Roberts, "Learning interaction dynamics with coupled hidden markov models," *Science, Measurement and Technology, IEE Proceedings -*, vol. 147, no. 6, pp. 345 –350, nov 2000.

[11] S. Aslam, C. F. Barnes, and A. F. Bobick, "Video action recognition using residual vector quantization and hidden markov models." in *IPCV'10*, 2010, pp. 659–666.

[12] C.-M. Oh, M. Z. Islam, and C.-W. Lee, "Pictorial structures-based upper body tracking and gesture recognition," in *Proc. 17th Korea-Japan Joint Workshop Frontiers of Computer Vision (FCV)*, 2011, pp. 1–6.

[13] E. Yu and J. K. Aggarwal, "Detection of fence climbing from monocular video," in *Proceedings of the 18th International Conference on Pattern Recognition - Volume 01*, ser. ICPR '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 375–378. [Online]. Available: http://dx.doi.org/10.1109/ICPR.2006.440

[14] D. Zhang, D. Gatica-Perez, S. Bengio, and I. McCowan, "Modeling individual and group actions in meetings with layered hmms," *Multimedia, IEEE Transactions on*, vol. 8, no. 3, pp. 509 – 520, june 2006.

[15] E. Guenterberg, H. Ghasemzadeh, V. Loseu, and R. Jafari, "Distributed continuous action recognition using a hidden markov model in body sensor networks," in *Proceedings of the 5th IEEE International Conference on Distributed Computing in Sensor Systems*, ser. DCOSS '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 145–158. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02085-8_11

[16] Y. A. Ivanov and A. F. Bobick, "Recognition of visual activities and interactions by stochastic parsing," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 22, no. 8, pp. 852–872, 2000.

[17] A. Mannini and A. M. Sabatini, "Gait phase detection and discrimination between walking–jogging activities using hidden markov models applied to foot motion data from a gyroscope," *Gait & Posture*, vol. 36, no. 4, pp. 657 – 661, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0966636212002342

[18] M. Nergui, Y. Yoshida, N. Imamoglu, J. Gonzalez, and W. Yu, "Human behavior recognition by a bio-monitoring mobile robot," in *Proceedings of the 5th international conference on Intelligent Robotics and Applications - Volume Part II*, ser. ICIRA'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 21–30. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33515-0_3

[19] F. De la Torre, J. K. Hodgins, J. Montano, and S. Valcarcel, "Detailed human data acquisition of kitchen activities: the cmu-multimodal activity database (cmu-mmac)," in *Workshop on Developing Shared Home Behavior Datasets to Advance HCI and Ubiquitous Computing Research, in conjuction with CHI 2009*, 2009.

[20] CMU, "Cmu graphics lab motion capture database," http://mocap.cs.cmu.edu/, 2004.

[21] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber, "Documentation mocap database hdm05," Universität Bonn, Tech. Rep. CG-2007-2, June 2007.

[22] B.-W. Hwang, S. Kim, and S.-W. Lee, "A full-body gesture database for automatic gesture recognition," in *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, april 2006, pp. 243 –248.

[23] M. Tenorth, J. Bandouch, and M. Beetz, "The TUM Kitchen Data Set of Everyday Manipulation Activities for Motion Tracking and Action Recognition," in *IEEE International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS), in conjunction with ICCV2009*, 2009.

[24] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1297–1304. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2011.5995316

[25] V. Bloom, D. Makris, and V. Argyriou, "G3d: A gaming action dataset and real time action recognition evaluation framework," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, june 2012, pp. 7 –12.

[26] D. Summers-Stay, C. Teo, Y. Yang, C. Fermuller, and Y. Aloimonos, "Using a minimal action grammar for activity understanding in the real world," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, oct. 2012, pp. 4104 –4111.

[27] C. Wolf, J. Mille, L. E. Lombardi, O. Celiktutan, M. Jiu, M. Baccouche, E. Dellandrea, C. E. Bichot, C. Garcia, and B. Sankur, "The liris human activities dataset and the icpr 2012 human activities recognition and localization competition," LIRIS Laboratory, Tech. Rep. RR-LIRIS-2012-004, Mar. 2012.

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

[29] S. Fothergill, H. M. Mentis, P. Kohli, and S. Nowozin, "Instructing people for training gestural interactive systems," in *CHI*, J. A. Konstan, E. H. Chi, and K. Höök, Eds. ACM, 2012, pp. 1737–1746.

[30] B. Ni, G. Wang, and P. Moulin, "Rgbd-hudaact: A color-depth video database for human daily activity recognition," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, nov. 2011, pp. 1147 –1153.

[31] S. Masood, C. Ellis, A. Nagaraja, M. Tappen, J. L. Jr., and R. Sukthankar, "Measuring and reducing observational latency when recognizing actions," in *The 6th IEEE Workshop on Human Computer Interaction: Real-Time Vision Aspects of Natural User Interfaces (HCI2011), ICCV Workshops*, 2011.

[32] E. Bribiesca, "A chain code for representing 3d curves," *Pattern Recognition*, vol. 33, no. 5, pp. 755 – 765, 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S003132039900093X

[33] E. Bribiesca and C. Velarde, "A formal language approach for a 3d curve representation," *Computers &amp; Mathematics with Applications*, vol. 42, no. 12, pp. 1571 – 1584, 2001. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0898122101002632

[34] E. Krause, *Taxicab Geometry: An Adventure in Non-Euclidean Geometry*. Dover Publ., 1987. [Online]. Available: http://books.google.com.mx/books?id=IW7ICV0QXWwC

[35] J. Figueroa, J. Savage, E. Bribiesca, and E. Succar, "Recognition of static gestures using three-dimensional chain codes using dominant direction vectors," in *Intelligent Environments (Workshops)*, 2012, pp. 231–241.

[36] O. Rodrigues, "Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des coordonnées provenant de ces déplacements considérés indépendamment des causes qui peuvent les produire," *J. Math. Pures Appl.*, vol. 5, no. 1, pp. 380–440, 1840.

[37] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," Tech. Rep. 8, 1966.

[38] D. E. Knuth, *The art of computer programming, volume 3: (2nd ed.) sorting and searching*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 1998.

[39] S. Hanov, "Fast and easy levenshtein distance using a trie," http://stevehanov.ca/blog/index.php?id=114, 1 2011.

[40] M. A. Vasconcelos, "Fast and easy levenshtein distance using a trie (in c++)," http://murilo.wordpress.com/2011/02/01/fast-and-easy-levenshtein-distance-using-a-trie-in-c/, 2 2011.

[41] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*, united states ed ed. Prentice Hall, Apr. 1993. [Online]. Available: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20\&amp;path=ASIN/0130151572

[42] J. Savage, "A hybrid system with symbolic ai and statistical methods for speech recognition," Ph.D. dissertation, University of Washington, Seattle, WA, USA, 1995, uMI Order No. GAX96-09599.

[43] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *Communications, IEEE Transactions on*, vol. 28, no. 1, pp. 84 – 95, jan 1980.