Framework for Posture and Face Recognition using Kinect an ambient-intelligence method

LIVIU VLADUTU, COSMIN MARIAN
Faculty of Automatic Control and Computer Science
University "Politehnica" of Bucharest
Address Splaiul Independentei no. 313
Sector 6, Bucureşti, 060042
ROMANIA

Abstract: - This electronic document describes one of the first full open-source frameworks for posture and face detection and classification using Kinect sensors, based entirely on an open-sourced software platform. It has been designed with the possibility of automatic monitoring of either an elderly person who is situated in a closed space (a room with furniture, appliances, and other rigid objects), or a patient who suffers from a physiological disorder. It is also based on an original hardware platform (the AmI-Platform), which includes more Kinect sensors, data acquisition stations, and crunches (*data servers).* The articles tackle two of the most important issues for ambient intelligence: accurate determination of the person and his or her facial landmarks (to understand emotions) and fast and reliable posture determination (to take appropriate measures in case of an accident).

Key-Words: - face detection and recognition, postures classification, Kinect, Ambient-Intelligence, Open Source, Ambient Assisted Living, sensor networks, facial landmarks.

Received: February 26, 2022. Revised: November 22, 2022. Accepted: January 15, 2023. Published: February 22, 2023.

1 Introduction

The problem that determined us to start this project from its inception was the context of the aging of Europe (aka the graying of Europe). There are 3 main factors that influence this phenomenon: the higher life expectancy, the decrease in fertility, and a certain decrease in the mortality rate (life expectancy has increased all over Europe, Japan, and North America). We designed a data pipeline for online acquisition, processing, and storage of localization and tracking of an older person in a room, as previously described in [3] and [1], in order to deal with an increased population of elder inhabitants and the need for constant care and supervision (for example, to avoid unwanted falls). As input devices, we used MS Kinect 360, the first generation of motion-sensing input devices developed by Microsoft.

Our context-awareness framework (we called it AmI-Lab) is applied to a major problem in ambient intelligence (from now on, AmI), namely ambient assisted living (a.k.a. AAL).

The European Commission's Information Society Technologies Advisory Group (ISTAG), see [5], is proactively supporting people in their daily activities. With more support in the last 10 years from the European Commission (see AAL Projects in the Reference Section), we aim to improve the quality of life of the elderly in their homes by increasing their autonomy, assisting them in their daily activities, and letting them feel more secure, protected, and supported. Our current approach continues our previous work in providing the AmI site with support for online data acquisition and multi-sensor fusion [1], [3], and will be completed with a comprehensive activity recognition framework—ubiquitous

supervision adapted for people with special needs (older people, persons with disabilities), from a clinical and non-clinical perspective. Typical emotion expressions are seen in various psychiatric conditions, including schizophrenia, depression, and autism spectrum conditions [2], so another target population for this project is represented by people with neurological disorders. The authors conclude that "emotions can be expressed in the face, gesture, posture, voice, and behavior and affect physiological parameters such as the heart rate or the body temperature."

Our paper is organized into the following nine main chapters: The second chapter briefly introduces the organization of the Ambient Intelligence Laboratory from the Faculty of Automation and Computers at the Polytechnic University of Bucharest, showing the physical structure of the laboratory and the main components of the data-acquisition pipeline. also briefly mentions the structure of the pipeline.

Chapter 3 shows our approach to posture representation and the components (segment lines, segment-to-segment angles, and segment-to-plane ones) that are used as features to assert the posture of the persons that are "seen" by the Kinect sensors.

Chapter 4 continues with a summary introduction of our proprietary skeleton viewer, written in Scala, a new programming language that smoothly integrates features of object-oriented and functional languages, enabling Java and other programmers to be more productive.

Chapter 5 follows with a detailed description of how we designed the experiments for both posture and faces recognition.

Chapter 6 introduces the basics of modern face detection and recognition algorithms in both images and video frames and their relation to the AmI-Lab platform.

The first part of chapter 7, "Experimental Results," emphasizes the labeling procedure we've used and the main 4 classification procedures (SVM, random forests, decision, and extra trees), which use an open-source toolkit, Scikit-learn, an ensemble of simple and efficient tools for data mining and data analysis built on Python, NumPy, SciPy, and Matplotlib.

Further on, the chapter concludes with figures showing the performance of the face recognition and detection algorithms.

Chapter 8 discusses related work (such as the state of the art in the last 20 years) for classification and detection of postures and faces.

Finally, chapter 9, together with conclusions and further milestones for the continuation of this project, aims mainly at tighter integration with the AmI-Platform for online skeleton classification and to show other researchers the importance of IoT, smart cities, and sensor network projects.

2 Problem Formulation

There were 2 distinct phases: 2.1. the hardware setup of the laboratory, including wiring, computers, cabled and wireless networks and with measurements coming from sensors (Kinect cameras, Arduino sensor boards, etc.) connected to the data-acquisition computers.

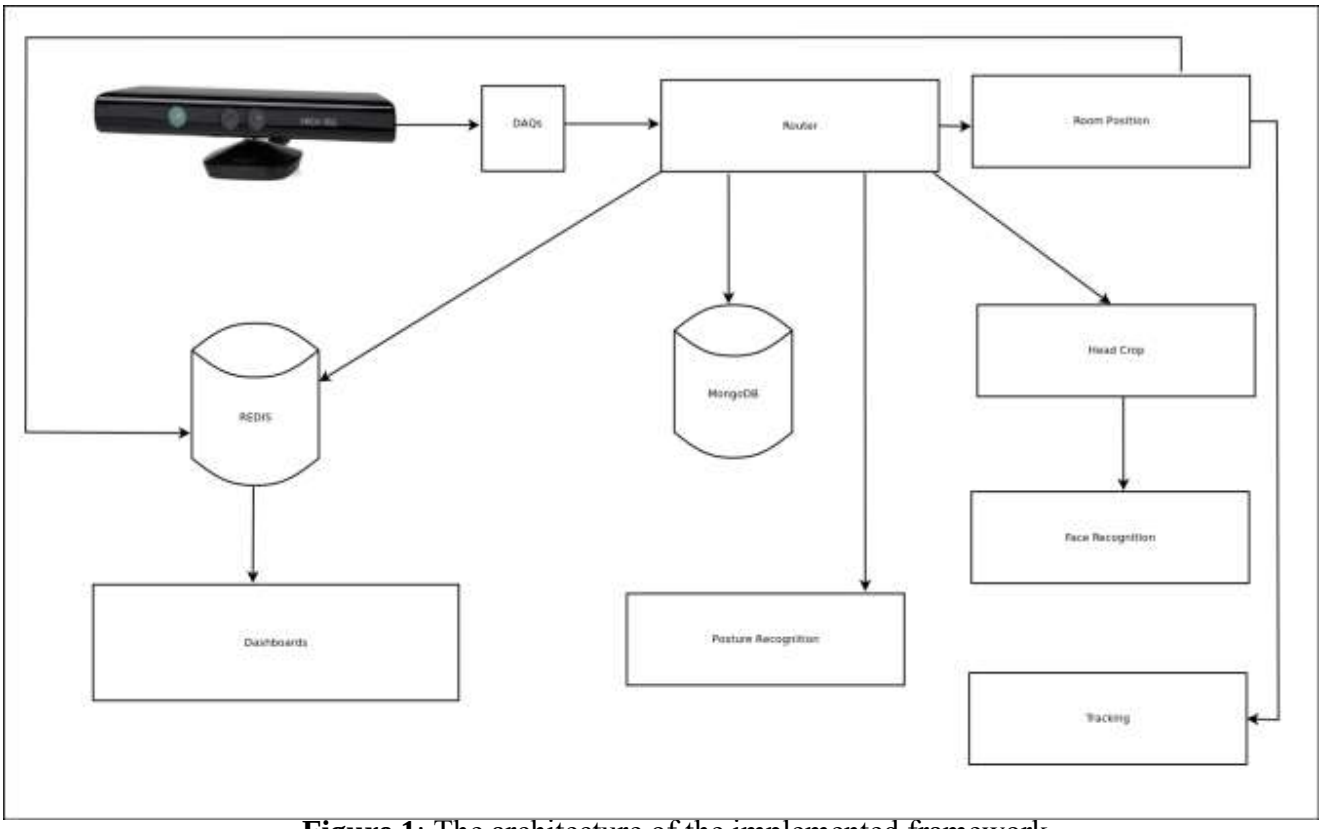

Figure 1: The architecture of the implemented framework.

2.2. the design of the data pipeline and toolboxes.

2.1 The Ambient-Intelligence Laboratory (AmI-Lab)

The experiments described in this paper were carried out at the Polytechnic University of Bucharest's Ambient Intelligence Laboratory (AmI-Lab). The laboratory contains both a hardware platform and a software platform [15], as extensively presented elsewhere: [1]. The hardware platform consists of nine Kinect cameras placed around the room, data acquisition computers, and network and data processing computers.

2.2 The AmI-Platform

The AmI Software Platform is a distributed platform architecture that acts as a data pipeline

2.2.1 Database

The data from sensors is sent to acquisition units for long- and short-term storage and to processing units. We used this platform to easily record experiments and retrieve the collected data in a format that is easy to read and process. The database we used is MongoDB (a NoSQL DBMS), and for communication between the PDUs (Processing Data Units), we are using Kestrel, a distributed messaging queue system open-sourced by Twitter. The architecture is presented in Figure 1.

2.2.2 The tracking system from Data pipeline

As clearly detailed in [1], the software that processes this data is organized as a pipeline made up of individual PDUs (processing data units) that communicate through a system of distributed queues. There are two types of PDUs: data acquisition PDUs, which poll the sensors for new data and record these

measurements into a database of trajectories, and data crunching PDUs, which use face recognition and voice recognition webservices and the recorded data to generate automatic examples by correlating spatio-temporal information. The software is installed on two types of processing nodes: data acquisition nodes (running data acquisition PDUs and the database engine) and data crunching PDUs (running detection and correlation algorithms).

3. Problem of Posture representation

For the purpose of posture recognition, we have chosen to use schematic skeleton data derived from the Kinect depth map images [6] by the NITE/OpenNI middleware. The skeleton provided consists of 15 3D joint positions (e.g., head, neck, right/left foot), forming 16 relevant segments between them (e.g., left/right arm, left/right leg) as described in Fig. 2. For each joint, the X, Y, and Z coordinates are provided in millimeters, relative to the camera coordinate reference system, with origins in the center of the camera and the Z axis pointing out of it.

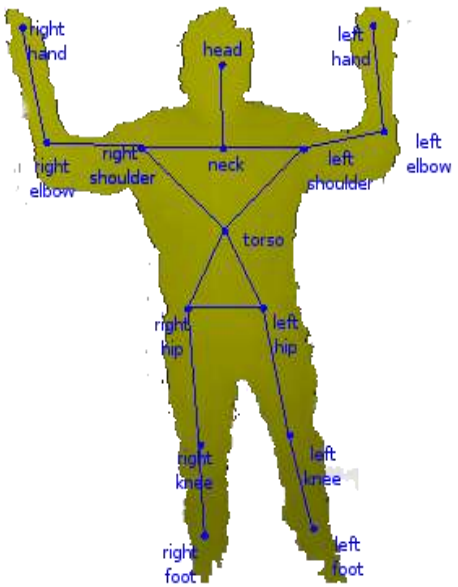

Figure 2. The distribution of the 15 3D joint positions used for acquisition

In order to represent a posture using these data, we have devised a list of features that we can compute based on the joint positions. The features chosen can be separated into two classes: angles between segments and angles between segments and the horizontal plane.

Below, we give the list of angles we have selected. Table 1 displays the 15 features of type angle between segment and segment. Table 2 displays a list of three features of the type of angle between segment and plane. Taking into account all features, we arrive at 19 features.

3.1. Purpose of framework for posture representation

The purpose of our framework is to use this set of features and supervised learning to automatically recognize a set of postures. This framework is independent of the set of postures considered, allowing us to use it in a larger range of possible applications where posture recognition from Kinect skeletons is needed.

For the purpose of evaluating this framework, we have selected a list of 10 postures that are distinguished by the relative positions of the two hands.

3.2. Features used for posture representation

- Right Hand down, Left Hand down (RH-D_LH_D) in Figure 4
- Right Hand lateral, Left Hand down (RH-L_LH_D)
- Right Hand Front, Left Hand down (RH-F_LH_D)
- Right Hand Up, Left Hand down (RH-U_LH_D)
- Right Hand down, Left Hand up, (RH-D_LH_U)
- Right Hand-L, Left Hand up (RH-L_LH_U)
- Right Hand-L, Left Hand-L (RH-L_LH_L)
- Right Hand front, Left Hand front (RH-F_LH_F)
- Right Hand up, Left Hand front (RH-U_LH_F)
- Right Hand up, Left Hand up (RH-U_LH_U)

4 Posture visualization & annotation

To help us visualize the collected data and annotate it in order to run the supervised learning algorithms, we have built a visualization tool using the Scala programming language [7], which runs on the Java platform, and using the Swing UI toolkit and OpenGL for 3D visualization of the skeleton. A screenshot of the window of the application is presented in Figure 3. The user can load a file of data consisting of measurements in JSON format ([1] and [8]) with 3D skeleton data and RGB data captured by the video camera. At the bottom of the window, there is a slider that allows the user to go from frame to frame and inspect and annotate the data.

The window contains a 3D visualization area where the user can rotate the skeleton in 3D in order to examine it, a RGB view that shows the image from the Kinect camera with the skeleton projected in 2D and superimposed over it, and an area that contains buttons with the considered postures. By pressing the corresponding buttons, the user annotates a certain skeleton to have that posture. The posture list is loaded from a text file, making the application itself independent of the considered set of postures. In the bottom left corner, the user has a list of the features considered, showing the value of each for the current selected frame. He can select certain features in order to display a color-coded map of the value of that feature under the slider (see Figure 3).

Once the relevant postures have been annotated, the user can save the postures along with their label in a file that can be loaded later or used in another application. This application proved very useful to us both in terms of easing the annotation process and helping us better understand the data.

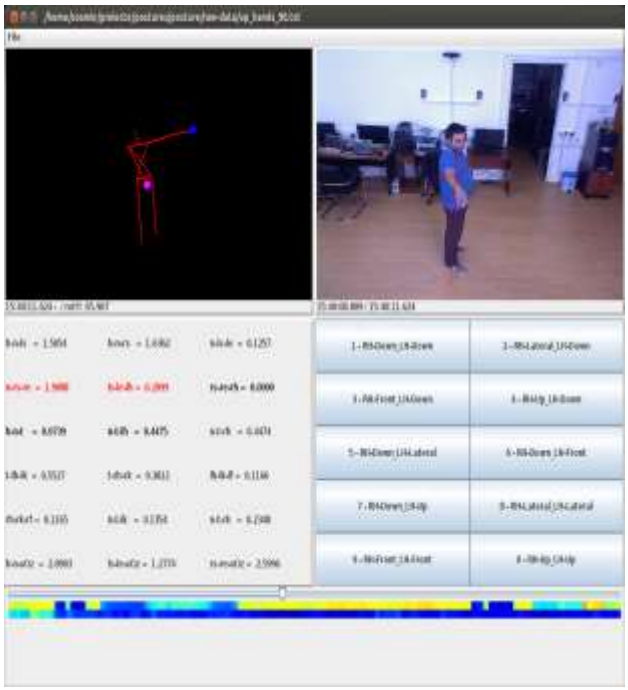

Figure 3. Screenshot of the viewer

5. Design of the experiment

The experiment was run in the Ambient Intelligence Laboratory of the University "Polytechnic" of Bucharest, Faculty of Automatic Control and Computer Science. The data was collected in several sessions. In each session, the subject positioned himself in the viewing field of a Kinect camera at variable angles (0, 45, 90, 135, 180 degrees) from the direction in which the camera was pointing and switched from posture to posture. The data provided by the Kinect, including the 3D skeleton, the 2D skeleton, and the RGB data with timestamps, was saved in several files [1]. The files contained on each line a measurement in JSON format with fields describing the measurement type (image_rgb or skeleton), the timestamp, and corresponding data (e.g., for image_rgb, the resolution and the compressed

image as JPEG with the Base64 encoding, and for skeleton, the 3D and 2D positions of joints).

In total, we have collected 9 raw data files consisting of 3867 measurements, of which 3736 were skeleton data and 104 were RGB images, totaling 69.3 MB. The approach we used is also compatible with other Scala-Spark big data frameworks and the latest data processing pipelines.

The next step was to load each data file in the viewer application and annotate the relevant skeletons with the correct postures. It is important to note that some postures were not considered to match any of the postures in our set, so they were not annotated. The annotated data was saved to files and merged into one single file. Each line of the file contained a labeled measurement in the JSON format containing the value of each feature, the label associated with it, and the timestamp. A total of 2527 postures were annotated in a file of 1.6 MB. The next step was to write a Python script that loads the data and uses the scikit-learn Python toolkit in order to run several learning algorithms on the labeled data and evaluate their performances using cross validation [9] with a split of 75% of the data for training and 25% of the data for validation. Scikit-learn is an open-source, simple, and efficient tool for data mining and data analysis built on NumPy, SciPy, and Matplotlib; see [10], [11].

We have found this tool very easy to use for the tasks we face, allowing us to write only 105 lines of Python code for loading the data, applying and validating the learning algorithms on the data, and displaying the results. The Scala code for our viewer will be uploaded to the folder "viewer-scala" in the repo [15].

The use of a supervised learning technique allowed us to reach our end goal of having a framework for automatically learning postures by just giving examples of skeletons that have a certain posture from a given set of postures.

6. Face detection and recognition implementation

Face detection and recognition is of primordial importance for us since the events that are about to occur (like falling) or just happened are to be observed not only in a posture alteration but also in how the facial landmarks are suddenly modified (in a grimace), showing both pain and surprise as well as anger. We used the 68-point mappings (for mouth, eyes, and eyebrows) that were obtained by training a shape predictor on the labeled HYPERLINK "https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/" iBUG 300-W dataset [12].

One can find in the literature books and tutorials on computer vision using OpenCV, the powerful library that we used in conjunction with dlib, numpy, pickle (for serialization), and face recognition developed using machine learning and OpenCV; however, I do strongly recommend the books by [13], [13], and [17].

Figure 5: All the students participating in the Face detection & recognition

Algorithms work quite fast, and for recognition we used >320 images with approximately 175 for testing. There were 8 classes for face classification: the 6 students from Figure 5, "first_author" (FA in Table 9), and a class "unknown," where we tested some faces from the "lfw" database ([16]; see References).

The classification results (97%) are promising, meaning that the algorithms need to be further studied and faster implementations are to be written in C++ (see last chapter), using OpenCV and gcc (work in progress). On all acquisition stations, we've used OpenCV (version 3.4.3) and Ubuntu 18.04 and 22.04 (LTS) as operating systems. The code for face detection and a new folder, "FaceRecognition," have been uploaded to GitHub in the "FaceDetect" folder [15].

That folder also contains the code for recognition in frames and videos, along with some of the bash scripts for checking the data and installing OpenCV.

The code also works for collections of frames (video-sequences) but we don't enter into this right now due to the lack of space.

7. Experimental Results

As we already mentioned, from 3736 postures gathered we annotated 2527 postures with the following labels (see Table 3). A number of 1209 postures were found not to match any of the classes. For the purpose of supervised learning, we have chosen to test four learning algorithms already implemented in scikit-learn. For each algorithm, we computed a score given by the ratio of the correctly classified examples from the validation set.

Another measurement we performed was to build and graphically display a confusion matrix [19]. The confusion matrix shows which postures were marked as other postures and how many times.

Below, we present the results we got from the four algorithms:

Support Vector Machines-Hearst (1998); For this algorithm, we obtained a score of 0.875. From the selected algorithms, this was the one having the worst performances.

The results are presented in Table 4.

Random Forest Classifier: This algorithm obtained a score of 0.96, making it one of the best-performing algorithms from the tested algorithms for our problem. The results are in Table 5.

Extra Tree Classifier: This algorithm had a score of 0.958, having a very close performance to the one obtained by the Random Forest Classifier. Results are in Table 6.

Decision Tree Classifier: **The score obtained by this classifier was 0.93 (Table 7).**

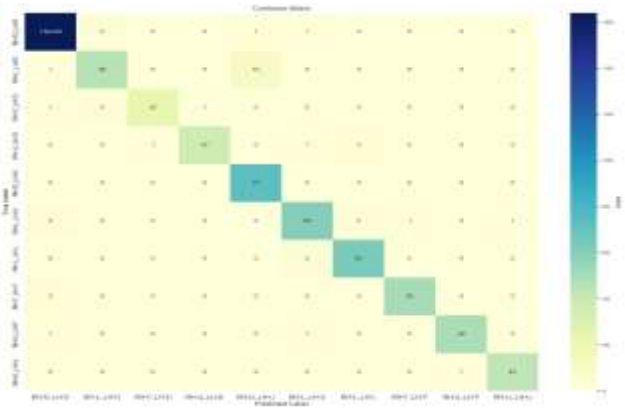

Figure 4. Decision Tree Classifier for postures (confusion matrix)-based on info in Table 5.

Figure 4 above shows a screen capture of the confusion matrix, yet another way to quantify the classification errors. Looking at the

performance of the results, we can conclude that we obtained good results for the framework we proposed and for the relatively short list of postures we considered. Even if the list of postures considered was short, we can notice that the postures themselves were very similar to each other, yet we obtained prediction scores that were over 95%.

Comparing the four algorithms, we can notice that the ensemble learning methods [21] had very good performance, being a good match for the problem and the set of features we have. To conclude this section, we present a table with the performance of the four algorithms tested (Table 8).

8. Related Work

Since the first introduction of the first-generation Kinect, back in November 2010, there have been many research groups dealing with the problem of body part posture classification using the schematic skeleton representation returned by a dedicated chip (within the Kinect sensor) for the range camera developed by Primesense, an Israeli developer. We are using the simplified expression "Kinect sensor," although in fact it represents a set of integrated sensors for range, visible light, sound, and infrared light.

The main applications for posture classification are:

1. human gesture recognition for AAL (ambient assisted living) scope (our aim);

2. sign-language understanding (using mainly hand-shape recognition), [22], and [23];

3. posture analysis for ergonomic training [24];

Apart from a previous work of our group [3], we are aware of some other similar recent works in kinect-based posture classification for AAL and senior monitoring applications, i.e., [25] and the EvAAL competition (2013). While one approach focuses on event detection (falling) when transitioning from sitting to standing and vice versa [25], the other primarily summarizes a competition of fast-setups (60 minutes) for indoor localization and tracking [26]. We have described a general framework for posture analysis (a complete set based on proprietary code for the pipeline of data acquisition, storage, and analysis) that can be easily extended to more postures and greater completeness.

While [27] only works with displaying a human skeletal joint model and using a LED cube, [28] analyzes more postures (22 of them), including leg postures, and uses PCA as feature extraction for a SVM classifier. Our work is, as far as we know, the only one that offers a complete kit of solutions (face and posture acquisition and classification) for human subjects that might need constant monitoring.

9. Conclusions & future work

At this moment, all the classification is of the type "supervised," meaning all the data for classification (i.e., matrices of angles for a posture) are known to which category they belong.

Since we have a large number of postures to classify, we used cross-validation in order to check how well the classifier can generalize when dealing with an independent set.

The main contributions of this work to the scientific community are:

A. the design of a proprietary, flexible pipeline for the acquisition and classification of human postures; we haven't used any commercial software whatsoever; all the code was designed by the authors using diverse programming languages (see above) or taken from the open-source community (from GitHub or SVN);

B. The classifier is easily extensible to include more families of postures (such as lying down with different hand and leg positions) or being bent sideways (with different hand positions). It all depends on the scenario that can be used for experiments. We have recorded all the experiments in a laboratory without any support to attenuate a (simulated) fall. These are the **main conclusions** about the importance of AmI-based elderly monitoring:

- Mobility monitoring: The system can monitor the movement and physical activity of the elderly, helping to detect any changes in their behavior and alert caretakers if necessary. The desktops for acquiring signals can be replaced by cheaper tablets (see previous experience, [29]) or laptops or smartphones;
- Ease of Use: cheap sensors (like Kinect) were designed to be user-friendly, making it easier for the elderly to use and interact with the system. This can be especially important for seniors, who may have limited technical skills or mobility.
- Continuous Monitoring: The use of sensors and other devices can allow for continuous monitoring of the elderly, providing a real-time understanding of their well-being and allowing for quick intervention if necessary.
- Improved Safety: By using AmI in conjunction with commodity hardware, elderly individuals can enjoy a safer and more secure environment with features like fall detection, emergency response systems, and real-time monitoring.
- Increased Independence: AmI systems can help the elderly maintain their independence and quality of life by enabling them to continue living in their own homes for longer periods of time. This can also reduce the burden on caregivers and nursing homes.
- Indoor localization: The system can provide real-time information about the location of the elderly within their home, helping caretakers respond quickly if needed.

Also as future work:

a) Systems that think like humans in interpreting ambient information (posture, face expression, sound, and trajectories) to understand a subject's mood and health/behavior in real-time.

b) Systems that act like humans in taking the appropriate, immediate actions as a response to a) above.

c) We aim to use this framework to create different "real-life like" scenarios (in context with its awareness framework), which can be linked to our recent IoT project;

d) Another idea is to try dimensionality reduction before classification using Principal Components Analysis (PCA) or Independent Component Analysis; this system can be leveraged on top of an existing marketplace in order to allow researchers from around the world to share sensor recordings among them;

• We have chosen Figshare (www.figshare.com) as the marketplace to share our sensor with other researchers from the global community so that they can test their data analysis and classification algorithms and codes as well.

• A version of the algorithms for face detection & recognition is in the works using gcc (versions 7.3, 8 and above as C++ compilers) instead of Python;

A new paper might include a modern approach for real-time detection and classification of objects using deep learning.

Table 1 [SEGMENT TO SEGMENT (FEATURES DEFINITION)]

Feature's index	Part 1	Part 2	Part 3
1.	head	Neck	left_shoulder
2.	head	Neck	right_shoulder
3.	neck	left_shoulder	left_elbow
4.	neck	right_shoulder	right_elbow
5.	left_shoulder	left_elbow	left_hand
6.	right_shoulder	right_elbow	right_hand
7.	head	Neck	torso
8.	neck	torso	left_hip
9.	neck	Torso	right_hip
10.	Torso	left_hip	left_knee
11.	Torso	right_hip	right_knee
12.	left_hip	left_knee	left_foot
13.	right_hip	right_knee	right_foot
14.	neck	torso	left_knee
15.	neck	torso	right_knee

Table 2 [The 3 segment-to-plane angles component]

Feature's index set2	Segment_End Point 1	Segment_End Point 2	Plane
1.	head	neck	xOz
2.	left_shoulder	left_elbow	xOz
3.	right_shoulder	right_elbow	xOz

Table 3
[THE 1209 MISCLASSIFIED POSTURES]

Posture's Index	Number of Misclassified	Significance of the posture
P1	718	R-hand Down-Left hand Down
P2	269	R-hand Down-Left hand Front
P3	251	R-hand Down-Left hand Lateral
P4	230	R-hand Down-Left hand Up
P5	149	R-hand-Front-Left hand Down
P6	185	R-hand Front-Left hand Front
P7	186	R-hand Lateral-Left hand Down
P8	219	R-hand Lateral-Left hand Lateral
P9	168	R-hand Up-Left hand Down
P10	152	R-hand Up-Left hand Up

Table 4
[RESULTS OBTAINED USING SVM CLASSIFIER]

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
P1	166	0	0	0	1	1	0	0	0	0
P2	7	28	2	0	10	0	0	0	0	0
P3	3	0	22	1	0	0	0	0	0	0
P4	0	0	2	34	0	0	3	0	0	0
P5	7	4	0	0	62	0	0	0	0	0
P6	8	0	0	0	0	57	1	0	0	0
P7	0	0	5	0	0	8	52	0	0	0
P8	5	2	0	1	0	0	0	45	0	0
P9	3	0	0	0	0	1	0	0	47	0
P10	0	0	0	0	0	0	0	0	4	40

Table 5
[RESULTS OBTAINED USING RANDOM FORREST CLASSIFIER]

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
P1.	164	2	0	0	1	1	0	0	0	0
P2.	1	46	0	0	10	0	0	0	0	0
P3.	1	2	22	1	0	0	0	0	0	0
P4.	0	0	1	37	0	1	3	0	0	0
P5.	0	0	0	0	73	0	0	0	0	0
P6.	2	0	0	0	0	60	2	1	0	1
P7.	0	0	0	0	0	2	63	0	0	0
P8.	3	0	0	0	0	0	0	50	0	0
P9.	1	0	0	0	0	1	0	0	49	0
P10.	0	0	0	0	0	0	0	0	1	43

Table 6
[RESULTS OBTAINED USING EXTRA TREE CLASSIFIER]

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
P1.	165	0	0	0	0	1	0	1	0	0
P2.	0	47	2	0	10	0	0	0	0	0
P3.	2	0	23	1	0	0	0	0	0	0
P4.	0	0	2	36	0	1	0	0	0	0
P5.	1	0	0	0	72	0	0	0	0	0
P6.	4	0	0	0	0	60	1	0	0	1
P7.	0	0	0	0	0	2	63	0	0	0
P8.	2	0	0	0	0	0	0	51	0	0
P9.	2	0	0	0	0	0	0	1	48	0
P10	0	0	0	0	0	0	0	0	3	41

Table 7

[RESULTS OBTAINED USING DECISION TREE CLASSIFIER]

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
P1.	162	2	0	0	4	1	0	0	0	0
P2.	2	43	1	0	1	0	0	0	0	0
P3.	2	2	22	1	0	0	0	0	0	0
P4.	0	0	4	35	0	1	3	0	0	0
P5.	0	0	0	0	70	0	0	3	0	0
P6.	3	0	0	0	2	57	2	0	1	1
P7.	0	0	0	0	0	1	62	0	0	0
P8.	2	1	0	0	0	0	0	50	0	0
P9.	2	0	0	0	0	0	0	1	45	0
P10	0	0	0	0	0	0	0	0	1	43

Table 8

[Performance of all the 4 classification algorithms for postures]

Algorithm Index	Type	Classification precision
1.	Support Vector Machines	87.5%
2.	**Random Forrest**	**96%**
3.	Extra-tree classifier	95.8%
4.	Decision-tree classifier	93%

Table 9

[RESULTS OBTAINED FOR FACES CLASSIFIER]

	Alex	Andrei	Cosmin	Denis	Diana	Gabriel	FA	Unk
Alex	11	3	0	0	0	0	0	0
Andrei	1	37	0	0	10	0	0	0
Cosmin	1	0	27	0	0	0	0	0
Denis	0	0	0	19	0	1	3	0
Diana	0	0	0	0	29	0	0	0
Gabriel	1	0	0	0	0	24	2	1
FA	0	0	0	0	0	0	11	0
Unk	0	0	1	0	0	0	0	11

Acknowledgment:

The authors would like to express their gratitude to the European Commission for the support of the ERRIC FP7 project.

This project insured the financial support not only for the hardware equipment from within the AmI-Lab but also for the authors monthly scholarships. Last, but not least, we would like to express our gratitude to Prof. Adina Florea (director of the project), Andrei Ismail, Diana Tatu, Cosmin-Gabriel Samoila, Denis Ilie and Mihai Trascau for the important contribution they made in the design of AmI-Lab project and for helping us with the experiments.

References:

[1] Ismail A. Marian C. and Florea A., "A Framework for Consistent Experimentation in AmI", *AgTAmI 2013 workshop, The 19th International Conference on Control Systems and Computer Science (CSCS)*, 2013.

[2] Grabowski K, Rynkiewicz A. et.al, "Emotional expression in psychiatric conditions: New technology for clinicians" in *Psychiatry and Clinical Neurosciences"- Japanese Society of Psychiatry and Neurology*, 2018.

[3] Ismail A., Florea A. "Multimodal Indoor Tracking of a Single Elder in an AAL Environment", Ambient Intelligence-Software and Applications: *4th International Symposium on Ambient Intelligence (ISAmI 2013-Advances in Intelligent Systems and Computing)*, Heidelberg, Springer-Verlag.

[4] Ten S., Mirror Image (2010), "How Kinect depth sensor works – stereo triangulation?", *https://mirror2image.wordpress.com/2010/11/30/how-kinect-works-stereo-triangulation/*

[5] European Commission, "I. A. Group. Scenarios for ambient intelligence in 2010", Ambient Assisted Living (AAL) Projects: http://www.aal-europe.eu/projects-main/

[6] Shotton J, Fitzgibbon A et. al. "Real-Time Human Pose Recognition in Parts from Single Depth Images", *IEEE CVPR*, pp. 1297-1304 2011

[7] Odersky M, Spoon L and Venners B. *Programming in Scala: Updated for Scala 2.12 3rd Edition*, Walnut Creek California, Artima Press, 2016.

[8] Crockford, D, "The application/json media type for javascript object notation (JSON)",RFC 4627, Network Working Group, 2006.

[9] Kohavi R, "A study of cross-validation and bootstrap for accuracy estimation and model selection." *IJCAI. Vol. 14.* No. 2. pp. 1137-1143, 1995.

[10] Pedregosa F. et al, "Scikit-learn: Machine Learning in Python", *The Journal of Machine Learning Research 12*, pp. 2825-2830, 2011.

[11] Cournapeau D., Scikit-Learn repo: http://scikit-learn.org

[12] Sagonas, C., Antonakos E., Tzimiropoulos G et al. "300 faces In-the-wild challenge: Database and results". *Image and Vision Computing (IMAVIS), Special Issue on Facial Landmark Localisation "In-The-Wild*", 2016.

[13] Rosebrock, A."Face Alignment with OpenCV and Python", PyImageSearch, 2017.

[14] Rosebrock, "Deep Learning for Computer Vision with Python, Practitioner Bundle, 1st Edition (1.2.2)", PyImageSearch, 2017.

[15] Vladutu L., Marian C., Ismail A., Trascau M. https://github.com/vliviu

[16] Huang Gary B, "Labeled Faces in the Wild": http://vis-www.cs.umass.edu/lfw/

[17] Kaehler A, Bradski G, *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*, O'Reilly Media, 2017.
[18] Rosebrock, A."*Practical Python and OpenCV*", PyImageSearch, 2016.
[19] Stehman, S. V., "Selecting and interpreting measures of thematic classification accuracy" *Remote sensing of Environment 62.1,* pp. 77-89, 1997.
[20] Hearst MA, "Support vector machines.", *Intelligent Systems and their Applications,* IEEE 13.4: pp. 18-28, 1998.
[21] Dietterich, T. G., "Ensemble methods in machine learning", in *Multiple classifier systems.* Berlin Heidelberg, Springer pp. 1-15, 2000.
[22] Keskin C., Kıraç F., Kara, Y. E., Akarun L.,"Real Time Hand Pose Estimation Using Depth Sensors", *Advances in Computer Vision and Pattern Recognition,* pp 119-137, 2013.
[23] Pugeault N., Bowden R., "Spelling It Out: Real–Time ASL Finger-spelling Recognition", *(ICCV 2011 Workshops) IEEE International Conference on Computer Vision,* pp. 1114-1119, 2011.
[24] Raya S. & Teizer J.,"Real-time construction worker posture analysis for ergonomics training", *Advanced Engineering Information*, Vol. 26, Issue 2, pp. 439-455. 2012.
[25] Parajuli, M., Dat Tran, Wanli Ma & Sharma, D., "Senior health monitoring using Kinect", using Kinect, IEEE Fourth Int'l Conference on Communications & Electronics (ICCE), 2012.
[26] EvAAL, Evaluation of localization and activity recognition systems for ambient assisted living: The experience of the 2012 EvAAL competition, *Journal of Ambient-intelligence and Smart Environments,* 5,pp.119 – 132, 2013.
[27] Zheng X. 3D Human Postures Recognition Using Kinect, Conference: *Proceedings of the 2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics*-Volume 01, pp. 344 – 347, 2012.
[28] Zhang Z., Liu Y., Li A., Wang M."A novel method for user-defined human posture recognition using Kinect", *7th IEEE International Congress on Image and Signal Processing*, pp. 736 -740, 2014.
[29] Albu F, Hagiescu D, Puica M, Vladutu L," Intelligent tutor for first grade children's handwriting application", Madrid, 2nd-4th of March, 2015. 1 *(INTED2015 ISBN: 978-84-606-5761-3)*, pp. 3708-3717.

Contribution of Individual Authors & collaborators to the Creation of a Scientific Article (Ghostwriting Policy)

Liviu Vladutu (LV), Cosmin Marian (CM) and Andrei Ismail carried out the programs design, implementation and optimization, experiments for the data pipeline. LV has drafted and corrected the paper.
Adina Florea was the Director of the project, supervising all phases.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

The project was financed by the ERRIC FP7 project, number 264207, *FP7-REGPOT-2010-1 (http://www.erric.eu)*

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.