Semantic matchmaking as a way for attitude discovery

Michele Ruta, Floriano Scioscia, Saverio Ieva, Giovanna Capurso, Eugenio Di Sciascio
Polytechnic University of Bari
via E. Orabona 4, I-70125, Bari, Italy
name.surname@poliba.it

Abstract—Powerful data analysis techniques are currently applied to 3D motion sensing devices like Microsoft Kinect for posture and gesture recognition. Though effective, they are computationally intensive and require complex training. This paper proposes an approach for on-the-fly automated posture and gesture recognition exploiting Kinect and treating the detection as a semantic-based resource discovery problem. A proper data model and an ontology support the annotation of body postures and gestures. The proposed system automatically annotates Kinect data with a Semantic Web standard logic formalism and then attempts to recognize postures by applying a semantic-based matchmaking between descriptions and reference body poses stored in a Knowledge Base. In addition, sequences of postures are compared in order to recognize gestures. The paper presents details about the prototype implementing the framework as well as an early experimental evaluation on a public dataset, in order to assess the feasibility of both ideas and algorithms.

I. INTRODUCTION

In latest years, low-cost three-dimensional (3D) motion sensing peripherals, such as Microsoft Kinect and Asus Xtion Pro Live, have been popularized by gaming and virtual reality applications. Basically, the hardware equipment does not vary significantly across available devices in the market, and generally includes a standard RGB video camera, one or more microphones and an infrared laser depth sensor (or a time-of-flight camera) for detecting 3D targets in a given environment. Reached resolution and accuracy in motion detection allow envisioning the exploitation of such devices in a wide range of industrial, healthcare and research applications. The limits in sensing precision—dictated by the mass production cost target—could be now balanced by software-side adjustments. Particularly, machine learning techniques allow posture and gesture recognition with enough precision for making the effectiveness off-the-shelf low-cost comparable with much more expensive specialized equipment.

This paper proposes a general framework for automated posture and gesture annotation and recognition, leveraging semantic-based matchmaking. 3D joint position data provided by Microsoft Kinect are annotated through Semantic Web languages according to a geometry-based data model. It allows describing both position and motion primitives, based on a formal ontology providing the required conceptualization. The posture and gesture recognition is treated as a discovery problem, exploiting non-standard inference services supporting non-exact matches [1] to compensate for possible sensing inaccuracies or environmental obstructions. In particular, stereoscopic data retrieved via Kinect are pre-processed to identify key postures, i.e., not transient body positions. They typically correspond to the initial or final state of a gesture. Each key posture is then annotated adopting the Web Ontology Language (OWL 2) standard for the Semantic Web, based on Description Logics (DL) formalism. Hence, non-standard inferences compare the built annotations with pose templates populating the Knowledge Base (KB) and a semantic similarity ranking supports the identification of the best matching posture. Analogously, sequences of key postures are linked together by annotations and compared with gesture template instances in the KB to discover the most plausible gesture.

The theoretical framework has been implemented in a fully functional prototypical tool, and experiments have been carried out on a public dataset [2]. Preliminary results report a satisfactory recognition precision for various kinds of gestures, validating the feasibility and effectiveness of the proposed approach.

The remainder of the paper is organized as follows. Section II discusses related work, while the proposed theoretical framework and approach are described in Section III. The prototype is presented in Section IV and experiments are reported in Section V, before conclusion.

II. RELATED WORK

Sensor-based activity recognition was surveyed extensively in [3]. Data-driven approaches adopt machine learning techniques for model generation or discrimination. However, knowledge-driven methods, e.g., ontology-based ones, have been recently attracting a growing interest.

Machine learning algorithms can be divided into supervised and unsupervised. The former require expensive manual labeling of training data, in order to build the discriminative model subsequently used for recognition. Moreover, when individual behavior or environmental conditions changes the model has to be re-trained. Conversely, probabilistic, graph-based or algebraic algorithms are exploited to generate the recognition models in unsupervised methods. For example, the framework proposed in [4] does not need expensive training stages, because it uses multiple low-dimensional eigenspace to build the models for different activities. Furthermore, proposed semi-supervised learning approaches [5] could be considered the best trade-off between accuracy and model building effort. While recognition performance was good, computational
complexity was rather high (allowing 12 frames per second at best).

Knowledge representation and logic-based reasoning are exploited by ontology-based approaches for sensor data modeling and activity recognition. A typical advantage of this category is that explicitly formalized knowledge can be shared among researchers, developers and practitioners, also allowing extensions of the core model for specific use cases. For example, in [6] an ontology was engineered to allow automatic annotation of human movements in the classic Benesh notation. Similarly, Video Event Representation Language (VERL) was proposed in [7] to allow a hierarchical representation of events. Furthermore, an Activities of Daily Living (ADL) DL ontology for activity modeling and reasoning in the context of smart homes is described in [8]. The Subsumption standard inference service was used to support activity recognition. Surveillance video analysis is a major application of ontology-based activity recognition [7], [9], as context modeling can improve detection accuracy. Finally, the majority of ontology-based proposals have adopted a top-down approach so far, focusing only on high-level activities and events.

However, user-friendly tools and/or domain-specific customizations are required for large data sets. For example, VERL [7] is complicated and verbose, so exhaustive definition of recognition rules is not practical. Nevertheless, standard inference services are often not enough for effective recognition, as pointed out in [10], because this task cannot be considered simply as classification, but it is more similar to model construction.

The work presented here extends the ontology-based approach described in [11], which helped domain experts expand the core KB through a visual workflow. Subsumption is replaced by the Concept Abduction non-standard DL reasoning task, which actually aims to build a concept (i.e., a model) for missing information whenever a full match cannot be achieved. Moreover, the proposal presented here is complementary to the standard approach, since it allows activity recognition from the bottom up. Finally, auto-tuning algorithms are introduced to compensate for variability in the environment and in user behavior.

III. KNOWLEDGE-BASED MOTION RECOGNITION APPROACH

A. General framework

The proposed framework for motion analysis, annotation and identification is user-independent and does not require a training phase. It exploits the infrared depth data stream from a Kinect sensor to detect and track up to two human subjects simultaneously. For each of them, the adopted NUI (Natural User Interface) API in the Kinect for Windows SDK [12] processes depth data at 30 fps (frames per second). It returns a skeleton structure composed of a hierarchy of 20 body joints, as shown in Figure 1. Each joint is marked with (x, y, z) coordinates in a 3D reference system with origin in the motion sensor itself. Joints undetected by the sensor (e.g., due to occlusion) are marked as inferred by the API, and its coordinates are estimated via proprietary algorithms by the Kinect embedded processing unit. Experience with the Kinect device has shown that inferred joint positions are often inaccurate.

Raw data → labelled data. The framework converts the original Cartesian coordinates of each joint to a spherical reference system centered in the parent joint. Zenith (θ) and azimuth (φ) angles relate each skeleton segment to its parent, e.g., thigh w.r.t. hip in Figure 1. The proposed model considers only angles, disregarding segment length as it is fixed for a given subject. In order to support annotation in Semantic Web languages, the framework maps computed angle data to direction labels, using θ and φ value ranges defined as shown in Figure 2, by means of the Cone-Shaped Directional (CSD) logical framework [13].

Labelled data → semantic annotation. The obtained labelled skeleton model is annotated in a fragment of OWL 2 language corresponding to the ACN (Attributive Language with unqualified Number restriction) DL. Basic DL syntax elements are: concepts (a.k.a. classes), representing sets of objects; roles (a.k.a. properties), linking pairs of objects; individuals, i.e., specific named instances of concepts. They can be combined into concept expressions by means of logical constructors. An ontology (a.k.a. TBox, terminological box) is a collection of inclusion and definition axioms, which model knowledge elicited for the domain. A Knowledge Base (KB) is formed by a TBox and an associated ABox (assertion box), containing descriptions of individuals. The ontology engineered for the proposed framework (not fully shown due to space concerns) includes the following main patterns. – Body. Skeleton joints and segments are represented as subclasses of the SkeletonJoint and SkeletonSegment classes, respectively. The hasParentJoint and hasChildJoint properties link segments to the joints at
their extremities. Subclasses of SkeletonBodyPart model basic body part positions, e.g., RightArmRaised, referring to subclasses of SkeletonSegment through azimuth and zenith CSD properties, reported in Figure 2. Additionally, the hasRightHandNear and hasLeftHandNear properties express the proximity of hands to other joints.

**Postures and gestures.** BodyPosture is the parent class of all postures modeled in the ontology, which are linked to body part position classes through the hasPosition property. Gestures are performed when the position of one or more body parts changes with a movement, e.g., bending both legs. Therefore the ontology describes a gesture as a sequence of *key postures*: the hasNext property links the preceding posture to the following one. Gestures are divided in SimpleGesture and CompositeGesture classes, which represent movements of just one body part and simultaneous movements of more body parts, respectively. SimpleGesture is further specialized in: LimbGesture, for gestures involving a single limb, e.g., raising the left arm sideways; SegmentGesture, which models finer movements involving body extremities (hands or feet), e.g., waving hand. Complex gestures are subclasses of CompositeGesture, expressed by the conjunction of several SimpleGestures by means of the hasGesture property. This is the case of many functional and expressive gestures in realistic scenarios.

With the above ontology model it is possible to annotate a posture either manually or automatically on the fly.

**Semantic annotation → attitude discovery.** Starting from the automatic annotated description of body postures, the recognition of user behavior is treated as a resource discovery problem. The framework includes a *semantic matchmaker* [1], exploited to detect the key posture template stored in the ABox of the reference KB having the highest degree of semantic similarity w.r.t. the real-time constructed annotation. Sequences of detected key postures are then composed into a gesture annotation, which undergoes a further semantic matchmaking process to identify the most similar gesture template in the KB. Semantic matchmaking is the process of determining the best matches among n resources $S_i$ ($i = 1, \ldots, n$) for a given request $R$, where both request and resources are annotated w.r.t. a common reference ontology [1]. In the proposed approach, $S_i$ are the key posture (respectively, gesture) templates in the KB, while $R$ is the current annotated posture (resp. gesture). Semantic matchmaking relies on the *Open World Assumption* (OWA): the absence of a feature in a description is not considered as a negation. The framework adopts *Concept Abduction* (CA) non-standard inference service to enable a logic-based relevance ranking of available resources w.r.t. a request. If $R$ and $S_i$ are not clashing but $R$ does not subsume $S_i$, CA determines what is missing in $S_i$. The solution $H$ (for *Hypothesis*) to CA represents "why" subsumption does not hold. In this way, it is possible to support approximate matches and to define metrics on $H$ to compute logic-based ranking of resources w.r.t. the request. Furthermore, a *similarity threshold* is introduced: key posture templates having an overall score worse than the threshold are excluded. Otherwise, detected postures are returned along with their scores and $H$ values, which serve as logic-based explanations of results; this is a distinguishing feature of the proposed approach w.r.t. most solutions based on machine learning. The overall degree of semantic similarity of the request w.r.t. and each resource is computed with the following *utility function* (1):

$$f(R, S_i) = 100 \times (1 - \text{penalty}(R, S_i) / \text{penalty}(R, T))$$  \hspace{1cm} (1)

where *penalty* is the CA-induced semantic distance of the annotated posture $R$ from a key posture template instance $S_i$; this value is normalized by the distance of $R$ from $T$ (the universal concept), which is the maximum possible penalty value and depends only on axioms in the ontology.

A toy example should clarify the above process, considering two resources in the key posture Knowledge Base:

$S_1$: *person standing up with straight and parallel legs, left arm straight along left side and right arm outstretched, head up looking straight ahead*. Adopting OWL 2 Manchester Syntax 1 it is expressed w.r.t. the domain ontology as:

$$S_1 \equiv \text{EquivalentTo: StandupPosture and hasPosition only (HeadUp and LeftArmAlongSide and RightArmOutstretched)}$$

$S_2$: *person standing up with straight and parallel legs, left arm raised on left side and right arm raised on right side, head up looking straight ahead*. In Manchester Syntax:

$$S_2 \equiv \text{EquivalentTo: StandupPosture and hasPosition only (HeadUp and LeftArmRaised and RightArmRaised)}$$

Similarly, consider the following annotated posture $R$:

$$R \equiv \text{EquivalentTo: Skeleton and hasPosition only (isUp only (Head and SpinalColumnSegment) and isDownwards only (RightUpperArm and RightLowerArm) and isOnTheRight only (RightUpperArm and RightLowerArm) and isDown only (LeftUpperArm and LeftLowerArm and LeftUpperLeg and LeftLowerLeg and RightUpperLeg and RightLowerLeg))}$$

Concept Abduction detects missing features in $S_1$; in $S_1$ the right forearm should be in horizontal position, whereas in $S_2$ the right upper arm should be slightly lowered and the left arm should be down. In OWL:

$$H_{R,S_1} \equiv \text{EquivalentTo: isDownwards only (RightLowerArm)}$$

$$H_{R,S_2} \equiv \text{EquivalentTo: isDown only (LeftUpperArm and LeftLowerArm) and isDownwards only (RightUpperArm)}$$

By applying the utility function, the overall similarity score is 91% for $S_1$ and 60% for $S_2$. $S_1$ is thus the best matching posture. Now, assuming the sequence $C = (S_1, S_2)$ is detected, the system infers the gesture dynamics by applying CA to pairs of subsequent recognized key postures:

$$H_{S_1,S_2} \equiv \text{hasPosition only (isHorizontal only (RightLowerArm and LeftLowerArm) and$$

isDownwards only (RightUpperArm and LeftUpperArm)\nH_{S_1,S_2} EquivalentTo: hasPosition only (isUpwards only (RightLowerArm and LeftLowerArm) and
isHorizontal only (RightUpperArm and LeftUpperArm))\n
H_{S_1,S_2} models the fact that arms start in horizontal position and are then lifted to vertical position in H_{S_1,S_2}. Therefore, the overall gesture annotation G is determined as:\nG EquivalentTo: hasNext only (hasPosition only
(isHorizontal only (RightLowerArm and LeftLowerArm) and
isDownwards only (RightUpperArm and
LeftUpperArm)) and hasNext only (hasPosition only
(isUpwards only (RightLowerArm and LeftLowerArm) and
isHorizontal only (RightUpperArm and LeftUpperArm)))

The framework computes semantic similarity between the detected gesture (playing the role of request in the discovery problem) and gesture templates (resources) in the ABox. The best matching gesture is returned along with the related score and H value.

Since all knowledge about behaviors of interest is encapsulated in the KB, the application domain of the proposed framework can be switched just by updating the ABox. Applications include gaming, fitness, patient rehabilitation, and worker training. In all these use cases, the semantic distance between the real-time annotation and the reference template can be exploited to evaluate user’s compliance or proficiency.

B. Dynamic self-tuning

The motion detection behavior is influenced by two parameters. Since gestures can be performed at variable speed, a Motion Sensitivity (MS) parameter determines the minimum time span a pose must be held for a successful key posture detection. Furthermore, the Precision Sensitivity (PS) parameter is defined as a similarity threshold below which detected postures are not used for gesture recognition. Since higher PS values typically lead to higher precision and lower recall, the framework includes automatic PS tuning to find the best trade-off, rejecting spurious detections while keeping an acceptable recall level.

In general, gestures are not detected at regularly spaced time instants, as the user does not move at a fixed pace. Given the current best posture similarity score at time T, the system will adapt the PS value for time T + 1, in order to filter out transition poses. The basic assumption is that, using the default 30 fps data stream, the similarity score will not change abruptly between consecutive frames. For this purpose, the proposed approach self-tunes PS with the following exponentially weighted moving average (EWMA):

\[ M_T = \alpha S_T + (1 - \alpha)M_{T-1} \]

where \( M_T \) is the next minimum semantic threshold, \( S_T \) is the detected posture similarity score at time T and \( M_{T-1} \) is the previous threshold. The decay factor \( \alpha \) varies in \([0, 1] \). The proposed self-tuning algorithm adopts \( \alpha = 0.05 \), thus working as a low-pass filter.

Preliminary tests with the proposed framework have shown that the presence of inferred joints reduces the detection accuracy significantly. Therefore the basic skeleton annotation step has been modified to omit segments containing inferred joints. This exploits the OWA explained in Section III-A in order to create annotations containing only information endowed with enough confidence. The semantic matchmaking algorithm, however, does not change. Furthermore, as similarity scores are influenced by the occlusion level, \( \alpha \) is set as variable in (2). If at least two leg joints and both hands are inferred, the occlusion is categorized as light and \( \alpha = 0.0125 \) in order to filter fluctuations even more. Figure 3a provides an example of the evolution of posture semantic similarity score and the corresponding EWMA. In areas (b) and (c) a light occlusion is detected: then the EWMA tends to flatten even though the posture score is highly volatile. Conversely, when there is no occlusion, such as in section (a), EWMA follows the score value more closely.

A severe occlusion occurs, instead, when the motion sensor marks as inferred at least one of the following sets of joints: (i) both hands and one arm joint; (ii) two torso joints; (iii) two leg joints and one foot joint; (iv) a total of at least six joints. In those cases, the quality of available information is severely degraded, therefore the semantic similarity threshold should be increased sharply, in order to consider only very close matches as positive detections. The following sigmoid function is applied to calculate an offset \( S_p \) to be added to (2):

\[ S_p = c \cdot S_{freq} / (S_{freq} + 1) \]

where \( S_{freq} \) is the number of occurrences of severe occlusions in the last 10 frames. The function is plotted in Figure 3c. Since sigmoid is between 0 and 1, a scale factor c is introduced. In case of severe occlusion, the decay factor is kept at \( \alpha = 0.05 \), resulting in the following formula:

\[ M_T = \alpha S_T + (1 - \alpha)M_{T-1} + S_p \]

Figure 3b plots an example of posture score, EWMA and related offset \( S_p \), in case of severe occlusions. the corrected EWMA quickly saturates to 95% to filter out false positives. Finally, in case of unclassified occlusions, \( \alpha \) is computed with the regression function plotted in Figure 3d in order to adapt dynamically to the inferred joint frequency \( O_{freq} \):

\[ \alpha = 0.0099 + 0.6107e^{-2.7508 \cdot O_{freq}} \]

IV. Prototype

In order to show the benefits of the proposed approach, a software prototype has been implemented, extending the existing tool [11]. The tool allows users to compose semantic annotations for body postures and gestures via the GUI in Figure 4, without requiring specific expertise in Semantic Web languages. Features are as follows:

A. When a subject is facing the Kinect device, her movements are tracked and skeletal data are retrieved and displayed on the top-left camera panel.
B. The user can activate real-time posture and gesture recognition, or load a pre-recorded stream. The system allows recording the raw tracking data: for each frame, the exported file contains the spatial coordinates of all skeleton joints.

C. The user can add a key posture or a gesture to the knowledge base and delete or rename previously saved templates.

D. The loaded reference ontology and the current posture annotation are shown in the top-right panel. The user can edit the posture description through drag-and-drop of classes and properties from the ontology, before saving it.

E. The user can manually set the MS and PS thresholds illustrated in Section III-A or activate PS auto-tuning as described in Section III-B.

F. The system allows to evaluate the overall detection performance on a specific dataset; results are exported to file.

G. Gestures are detected as explained in Section III-A and are added to the timeline in the bottom panel.

V. EXPERIMENTS

An experimental campaign has been carried out to evaluate the performance improvements associated to the adoption of the PS auto-tuning. The public Gaming Action (G3D) [2] dataset has been selected, providing the synchronized video, depth and skeleton data streams of seven different gestures. Only five gestures—reported in Table I along with their key postures—have been considered for the experiments, because for the other two gestures manual dataset inspection has shown frequently incorrect execution.

Recognition performance has been evaluated under the following settings: (1) enabling auto-calibration and excluding inferred joints from annotations; (2) disabling auto-calibration and excluding inferred joints from annotations; (3) enabling auto-calibration only; (4) using the older framework [11], as a baseline reference. PS was set to 75% in configurations 2 and 4. MS value was set to 500 ms in all cases except the Miscellaneous sequence, where it was lowered to 260 ms. Precision, recall and F-score have been selected as performance metrics; consecutive detections of the same posture have been considered as a single true positive, if the posture is correct.

Results are reported in Figure 5: legend keywords Partial/Total explain the exclusion/inclusion of inferred joints in annotations, while ON/OFF are referred to the auto-tuning setting. Outcomes show that the proposed automated approach globally improves the system performance w.r.t. the original framework [11]. When auto-tuning is enabled, almost all false positives are eliminated and F-score values are higher in two out of five cases. However, a slight decrease in recall is observable, due to the fact that true positives decrease in favor of false positives. Concerning the Golfing gesture sequence, the reference posture is detected correctly only
TABLE I: Gestures tested from the G3D dataset.

<table>
<thead>
<tr>
<th>Gesture</th>
<th>Descriptive text</th>
<th>Key postures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driving</td>
<td>Swerve left and right</td>
<td></td>
</tr>
<tr>
<td>Fighting</td>
<td>Raise right hand, kick and then bring hands up on the face</td>
<td></td>
</tr>
<tr>
<td>First Person Shooter</td>
<td>Point with right hand, walk, run, jump, climb and crouch</td>
<td></td>
</tr>
<tr>
<td>Misc.</td>
<td>Raise right hand up, stand with arms along sides, raising arms on sides, standing with arms along sides again, and applauding</td>
<td></td>
</tr>
<tr>
<td>Golfing</td>
<td>Hit a ball with a golf club</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 5: Experimental results

when inferred joints are excluded from annotations, due to the profile posture always producing occlusion during the gesture execution. Consequently the precision sensitivity tends to stay close to saturation, as described in Section III-B. The observed behaviour justifies the exclusion of inferred joints in hard cases.

VI. CONCLUSION AND FUTURE WORK

The paper extended a general-purpose framework for semantic-based gesture annotation and recognition. It exploits 3D joint position data provided by a motion sensing device such as Microsoft Kinect. The work improved on the features in [11] with heuristic algorithms for motion and precision sensitivity auto-tuning. Furthermore, leveraging the Open World Assumption, the work suggests the opportunity to disregard inferred joints due to occlusion. The framework has been implemented in a prototypical tool: early results obtained with respect to a reference dataset validate the proposed approach. Future work aims to enhance the presented framework toward action recognition as a sequence of gestures. This goal will require an extension of both data model and domain ontology, including context information. Finally, an experimental comparison with state-of-the-art approaches is planned both in terms of accuracy and computational performance.

REFERENCES