

Article

Learning and Mining Player Motion Profiles in Physically Interactive Robogames

Ewerton L. S. Oliveira ^{1,*} , Davide Orrù ¹, Luca Morreale ¹, Tiago P. Nascimento ²  and Andrea Bonarini ¹ 

¹ Department of Electronics, Information, and Bioengineering, Politecnico di Milano, Piazza Leonardo da Vinci, 32, 20133 Milano, Italy; davide.orru@mail.polimi.it (D.O.); luca.morreale@mail.polimi.it (L.M.); andrea.bonarini@polimi.it (A.B.)

² Department of Computer Systems, Federal University of Paraíba (UFPB), João Pessoa, PB 58051-085, Brazil; tiagopn@ci.ufpb.br

* Correspondence: ewerton.lopes@polimi.it; Tel.: +39-02-2399-3400

Received: 30 October 2017; Accepted: 14 December 2017; Published: 26 February 2018

Abstract: Physically-Interactive RoboGames (PIRG) are an emerging application whose aim is to develop robotic agents able to interact and engage humans in a game situation. In this framework, learning a model of players' activity is relevant both to understand their engagement, as well as to understand specific strategies they adopted, which in turn can foster game adaptation. Following such directions and given the lack of quantitative methods for player modeling in PIRG, we propose a methodology for representing players as a mixture of existing player's types uncovered from data. This is done by dealing both with the intrinsic uncertainty associated with the setting and with the agent necessity to act in real time to support the game interaction. Our methodology first focuses on encoding time series data generated from player-robot interaction into images, in particular Gramian angular field images, to represent continuous data. To these, we apply latent Dirichlet allocation to summarize the player's motion style as a probabilistic mixture of different styles discovered from data. This approach has been tested in a dataset collected from a real, physical robot game, where activity patterns are extracted by using a custom three-axis accelerometer sensor module. The obtained results suggest that the proposed system is able to provide a robust description for the player interaction.

Keywords: robot games; human-robot interaction; player modeling; game design; latent Dirichlet allocation

1. Introduction

In the last decade, video game companies have introduced a new paradigm that involves players actively moving in front of the screen, picking up simulated objects around them and interacting with the game in a three-dimensional space. This has been mostly enhanced by immersive, Virtual Reality (VR) games, where players are plunged into an artificial world reproduced by ad-hoc intelligent systems and devices [1]. These scenarios, although enabling a deep game experience, may in some cases introduce potential health and safety issues, such as cybersickness and deleterious physiological aftereffects [2].

Physically Interactive RoboGames (PIRG), on the other hand, comprise an emerging application field that consists of a number of autonomous agents (including software and hardware) of which at least one is an autonomous, physical robot and one is a person. These agents interact with each other in the real world by following some game rules so that the human players can have fun [3].

PIRG may be seen as well as a specific setting to study human-robot interaction in situations framed by rules, where attention and physical challenges are involved. Furthermore, the increased maturity of fields like social robotics, artificial intelligence and machine learning offers the possibility

to design a new generation of such games, where adaptive behavior to support human entertainment, for instance, plays a central role. In general, such applications can foster the wide spread of robots in the society via ludic applications.

In a PIRG, the physical, autonomous agents are actively engaged, creating some sort of interaction, either competitive or cooperative. From one's perspective, such games are similar to robotic Exergames [4–8], but with a mobile companion and with less concern for promoting specific activity conditions driven by therapeutic motivations. Like commercial video games, the main aspect in a PIRG is to produce a sense of entertainment and pleasure that can be “consumed” by a large number of users.

It has been noticed [3,9] that an important aspect of such playing entities and systems during the game should be an exhibition of rational behavior, and in this sense, they must be able to play the role of effective opponents or teammates, since people tend to lose their will to play with, or against, a dull entity. Therefore, a robot adapted to the skill and attitudes of the human player may achieve an acceptable performance, possibly providing a highly entertaining experience. Such requirements motivate the design of player's categorization models in order to understand the player's behavior and preferences, as well as to guide the expected robot adaptation.

Possible issues may arise, mainly due to the challenges in modeling the real world and the problem of overcoming common limitations present in a robotic platform. For example, a PIRG designer building a mobile playing agent has to deal with the problem of noisy sensory data, partial information, perception, locomotion and planing in a real environment. Safety issues regarding human-robot interaction are, as well, an important factor that has to be taken into account. All of those aspects, together with budget constraints, make the design of intelligent playing robots a difficult task.

Companies like [Anki®](#) started to make use of artificial intelligence techniques to bring consumer robotics into physical gaming experience. One of its product, Anki Drive®, features toy-sized robotic cars controlled by a mobile app. In the game, the cars sense their positions on the track and continually exchange data via Bluetooth® with a smartphone. The app computes possible actions and decides what each car should do based on its objective. Additionally, the cars can also show different “personalities”. Despite the app's ability to orchestrate the behaviors of the cars, players are still limited to the use of the smartphone to drive their car and cannot participate in first person in the game.

Following such an example, one may envision PIRG as one of the next robotic products for the mass technological market, which demands a large exploration of new methodologies, especially as concerns the methods for enabling high autonomy, intelligence and adaptive behavior.

Over the years, there have been some important attempts at presenting robots as games. However, in most cases, the robots act more or less as a mobile pet (e.g., [10,11]), where the interaction is often seen as limited to almost static positions [12], not exploiting rich movement, nor a high level of autonomy, which makes for a low credibility of these toys to really engage lively people [3,9].

Some PIRG prototypes have been reported over the last few years. Jedi Trainer 3.0 allowed for a more physical interaction between robot and players [3]. It was developed with the goal of replicating a scene of the first Star Wars saga movie “Episode IV—A New Hope”. An autonomous drone flew around the human player (Jedi trainee) and could produce, when needed, the sound of a laser blast. Jedi Trainer was successfully tested in different unstructured environments [9], being capable of showing some apparent adaptation, achieved by always making the drone tend to keep a fixed distance from the player so that the speed of movement of the latter could be matched by the drone. The randomness arising from noisy sensors added realism to the drone's behavior, which could be perceived as some kind of rational behavior by the human player [3].

Another example of PIRG with mobile robots was RoboTower [3,9]. In the game, a 30 cm high robot had to push down a set of towers in three minutes. Human players could interact with the robot by delaying its movement by placing specific cards on the robot's path; the robot was capable of reading the cards when passing over them. Each card represented either an action the robot had to execute (go back, turn around, stop) or a deficit of its sensors (go blind). There, the authors reported

that engagement was related to the challenge of putting the cards at the right place and by the pressure given by time.

A different and more complex version of RoboTower has been recently implemented [13,14], where humans interact with a three-wheeled omni-directional robot one meter high. In this version, only physical interaction is exploited. The present work uses this version of the game as the case study.

Closely related to PIRG, some proposals of user models in games have been made using Playware playground [15]. Some of them consider physiological data, such as heart rate [12], for capturing and modeling individual entertainment preferences in children. Others focus on more general attributes like force sensing data and entropy of game events [16].

Despite the examples of successful PIRG applications using mobile robots, only a few works [12,14] concern a direct quantification of the human behavior during game play. In this paper, we present an approach to categorize the players' behaviors by analyzing data about their movement. The primary source of information is a single three-axis accelerometer placed on the player's chest. The analysis is performed by first mapping the stream of data into an image representation, namely a Gramian Angular Field (GAF) [17], which is subsequently processed by a simple autoencoder [18] architecture for unsupervised feature extraction. After the definition of a dictionary of primitives on the feature space, we apply latent Dirichlet allocation [19], a generative model mostly used for document retrieval and classification, to uncover coherent groups of motion patterns that can be used to categorize players.

Our decision to use Latent Dirichlet Allocation (LDA) takes inspiration from recent approaches where each game session is represented as a "document", and chunks of generated stream data are encoded into discrete "words" [20]. This arrangement is based on the assumption that different players generate different streams of input words, thus different documents, depending on their own motivation or play style.

The possibility to apply LDA to this problem is further reinforced by noticing the apparent difficulty in objectively separating types of motion patterns, since a player in his/her playing activity may show one or more different motion styles. More explicitly, a player may, for instance, become bored or frustrated with the game and consequently show a low motion profile w.r.t. the one that the same player might have shown at other moments during the same game session; for instance, when he/she was excited about it. Here, we consider that a high motion profile, i.e., high turbulence in acceleration signal, is typical for a highly motivated player, since by common sense, a non-motivated player tends to stay relatively still or show a very low acceleration signal.

The claim that motion style conveys engagement or even emotion [21–23] types has been supported by several papers. As an example, systematic approaches for describing movement, such as Laban Movement Analysis (LMA) [24], are often used in deriving low-dimensional representation of movement, facilitating affective motion modeling [25]. This same type of analysis has been applied when investigating motor elements that characterize movements whose execution enhances basic emotions, such as: anger, fear, happiness and sadness [22]. In game situations, despite the limited capacity for entertainment detection and modeling in Exergames, LMA had been reported as useful in fostering emotion recognition in states like: concentration, meditation, excitement and frustration [26].

Here, we hypothesize that, on average, a player will display his/her game-intrinsic motion style and convey some information about his/her interest in/motivation to play. Discovering which coherent groups of motion style exist in a dataset allows us to estimate to what extent an unknown player relates in terms of his/her own motion profile to known groups. This may ultimately help in designing PIRGs that are able to adapt accordingly in order to offer a better user experience for the players.

Following this, in this manuscript, a player representation is addressed as how much of each discovered motion types to which we are confident to say a given player is mostly related. We call this representation a mixture proportion of motion types (similar to topics in the LDA perspective).

In summary, we expect to see that similar player-generated data are grouped together in a coherent collection, allowing the distinction of the existing motion types. The results obtained are validated with human supervision, suggesting the applicability of our method to future mechanisms for designing auto-adapting PIRGs agents.

2. Mathematical Preliminaries

This section presents the basic concepts and principles of the Gramian angular field, autoencoders and latent Dirichlet allocation, our targeted probabilistic graphical model.

2.1. Gramian Angular Field Images

The results obtained in this work rely on the transformation of the time series data generated by the player motion into images that can then be processed and understood in terms of emergent patterns that can, in turn, be discovered by state-of-the-art machine learning algorithms. The type of image we have chosen is Gramian Angular Summation/Difference Field (GASF/GADF). Such images had been proposed in the field of time series classification [17], where the authors evaluated the efficacy of representing time series in a polar coordinate system instead of the typical Cartesian coordinates. For a deep understanding of the method, we suggest reading the original paper, where the authors did not only present the idea of GASF/GADF, but also reported experimental results comparing them with other time series imagery techniques. Here, we present an overview of the technique in order to familiarize the reader with such a method and to point out its role in our proposal.

In general, in order to obtain a GASF/GADF representation for a time series $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$ of size $n \in \mathbb{N}_{>0}$, rescaling is first performed in order to put the time series in the interval $[-1, 1]$ or $[0, 1]$. The second step is to represent the rescaled time series $\tilde{\mathcal{X}}$ in polar coordinates, where each sample value is now mapped into the angular cosine and the time stamp information is represented as the radius. The formula for this step is as follows:

$$\begin{cases} \phi = \arccos(\tilde{x}_i) & -1 \leq \tilde{x}_i \leq 1, \tilde{x}_i \in \tilde{\mathcal{X}} \\ r = \frac{t_i}{N} & t_i \in \mathbb{N}_{>0}, \end{cases} \quad (1)$$

where t_i is the time stamp and N plays the role of a normalizer for the span of the polar coordinate system. Some properties of this transformation are known:

- Property #1: The encoding map of Equation (1) is bijective as $\cos(\phi)$ is monotonic when $\phi \in [0, \pi]$. Given a time series, the proposed map produces a unique result in the polar coordinate system together with a unique inverse map.
- Property #2: As opposed to Cartesian coordinates, polar coordinates preserve absolute temporal relations.
- Property #3: Rescaled data in different intervals have different angular bounds. That is, $[0, 1]$ corresponds to the cosine function in $[0, \frac{\pi}{2}]$, while cosine values in the interval $[-1, 1]$ fall into the angular bounds $[0, \pi]$. These intervals are known to produce different information granularity in the Gramian angular field, especially for classification tasks [17].

The third step in completing the transformation is to exploit the angular perspective by considering the trigonometric sum/difference between each point. This allows for the identification of the temporal correlation within different time intervals. The two types of GAFs, namely GASF and GADF, are obtained as follows:

$$\begin{aligned} GASF &= [\cos(\phi_i + \phi_j)] \\ &= \tilde{\mathcal{X}}' \cdot \tilde{\mathcal{X}} - \sqrt{I - \tilde{\mathcal{X}}'^2} \cdot \sqrt{I - \tilde{\mathcal{X}}^2} \end{aligned} \quad (2)$$

$$\begin{aligned} GADF &= [\sin(\phi_i + \phi_j)] \\ &= \sqrt{I - \tilde{\mathcal{X}}^2} \cdot \tilde{\mathcal{X}} - \tilde{\mathcal{X}}' \cdot \sqrt{I - \tilde{\mathcal{X}}^2} \end{aligned} \quad (3)$$

I is the unit row vector. After translating the time series from the Cartesian to polar coordinate system, each time step in it is taken as a 1D metric space. By defining the inner product as $\langle x, y \rangle = x \cdot y - \sqrt{1 - x^2} \cdot \sqrt{1 - y^2}$ and $\langle x, y \rangle = \sqrt{1 - x^2} \cdot y - x \cdot \sqrt{1 - y^2}$, respectively, the GAFs become quasi-Gramian matrices, since the defined $\langle x, y \rangle$ do not satisfy the property of linearity in inner-product space.

Still, according to [17], the GAFs have several advantages:

- Advantage #1: They provide a way to preserve temporal information. The temporal correlation is present because $G_{(i,j||i-j|=k)}$ represents the relative correlation by the superposition/difference of directions with respect to time interval k . The main diagonal $G_{i,i}$ is the special case when $k = 0$, which contains the original value/angular information.
- Advantage #2: From the main diagonal, it is possible to reconstruct the time series back into the original Cartesian space. This is useful especially in a classification task, where we can convert the time series back to numerical Cartesian space from the high level features learned by, for instance, a deep neural network.

However, the size of the Gramian matrix G is $n \times n$, for n being the length of the raw time series. The authors propose a solution for this issue by attempting to reduce the size of the produced GAFs using, for instance, Piecewise Aggregation Approximation (PAA) [27] to smooth the time series while preserving the trends. An example of the conversion from time series to GASF image is shown in Figure 1.

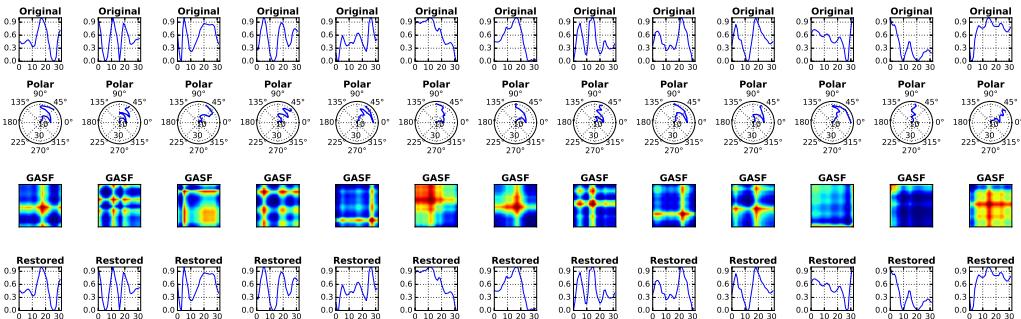


Figure 1. Conversion of time series into a Gramian Angular Summation Field (GASF) image and then back into its original space.

2.2. Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is one of the most simple and influential topic models. A topic model is a formal statistical relationship between a group of observed and latent (unknown) random variables, which specifies a probabilistic procedure to generate topics. The latter, in turn, is defined as a probability distribution over a collection of words [28].

The graphical model shown in Figure 2 presents the structure of the LDA model [19]. Here, as in [20], the K -th probability distribution over input words β (called topics) is representative of the K groups (types) of player motion styles in which we are interested. In turn, each document d is a game play session for the game. We had to deal with the deterioration of model accuracy when considering the whole game section as a single document. Our solution for the problem was to segment each one of those into subsets based on duration in seconds. This approach raises several issues that are going to be properly addressed in a future work, but still allow one to obtain interesting results.

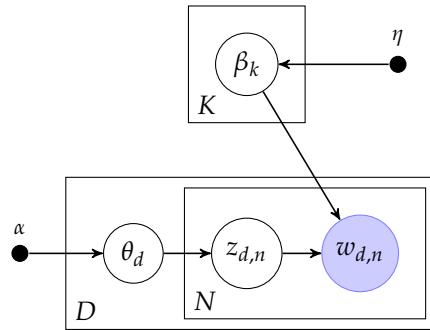


Figure 2. Latent Dirichlet Allocation (LDA) model diagram [19]. Each word $w_{d,n}$ corresponds to a GASF encoding obtained from an accelerometer attached to the player during game play. In turn, a game play is interpreted as a document d (set of player-generated words, i.e., action patterns). $z_{d,n}$ represents the type (topic) assignment for a given word $w_{d,n}$, and θ_d is the mixture proportions representing the game session. In this work, we are interested in representing the player game session by θ_d .

In these terms, a document may present multiple types β_k with mixture proportions θ_d . This mixture of proportions defines a player at each instant of time. The choice of the N_D words comes from $\bar{\beta}_d$, which acts as the weighted average of the game play types for a given document. This average is defined as follows [19,20]:

$$\bar{\beta}_d = \sum_{n=1}^K \theta_d(k) \cdot \beta_k \quad (4)$$

The n -th word $w_{d,n}$ of document d is, in our case, the GASF encoding obtained, as we will explain later. $z_{d,n} \in \{1 : K\}$ refers to the random type from which a word is drawn. Following the standard LDA model, α and η are hyperparameters that control the sparsity in the mixture proportions and word distributions [20].

2.3. Autoencoder

In many problems, we may have to deal with highly dimensional data that represent the situation of the world as captured by the sensors. Given that the raw data dimensionality could be practically intractable, the common procedure in these cases is to reduce the dimensionality possibly keeping only the important features of the input data.

In image processing, for instance, understanding which pieces of an image represent a significant aspect for the task ahead is not a trivial issue. Significance may be related to specific parts of the image and may depend on many aspects. A common technique to face this issue is autoencoder. Autoencoder is an unsupervised learning technique that learns a representation of the input data, usually derived by means of non-linear transformations of the input [18]. A typical implementation of autoencoder consists of a feedforward neural network that first encodes the input into a code, the representation and then reconstructs the input from it by means of a decoder (see Figure 3).

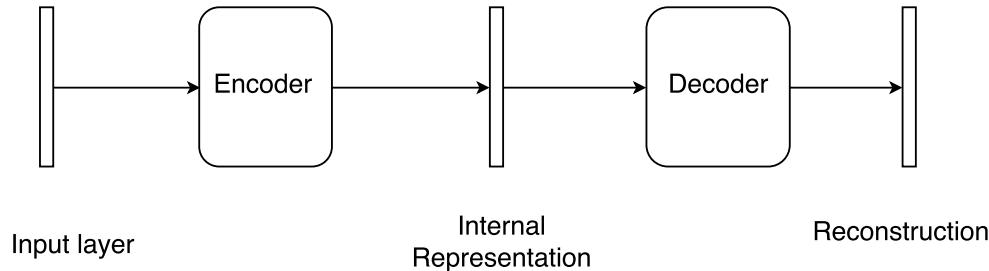


Figure 3. General architecture of an autoencoder; the input layer is processed by an encoder that transforms it. The derived representation is then decoded by a decoder that yields a reconstruction.

In general, autoencoders are good at capturing the structure of the input and tend to represent it well [18]. We can represent an autoencoder as a neural network that tries to copy its input to its output. We can represent its general architecture with the following terminology:

- the input x ,
- the encoder function, $f(x)$, composed of some hidden layers,
- the code, or internal representation, of the input h ,
- the decoder function, $g(x)$, composed of other hidden layers,
- the reconstruction obtained from the decoder, r .

A loss function, $L(x, r)$, should also be defined to evaluate how close the reconstruction is with respect to the input. A simple loss function is shown as Equation (5).

$$L(x, r) = \sum_i (x_i - r_i)^2 \quad (5)$$

However, if we do not impose any restriction, the transformation it learns would be an identity, since this would perfectly reconstruct the input. To prevent this transformation, two approaches have been devised [18]. The first, called undercomplete autoencoder, adopts a representation smaller than the input (see Figure 4a), while the latter, overcomplete autoencoder, uses a representation with a dimension larger than the input (see Figure 4b). Although increasing the representation size is counterintuitive in order to sparsely represent the input, this technique has proven its effectiveness.

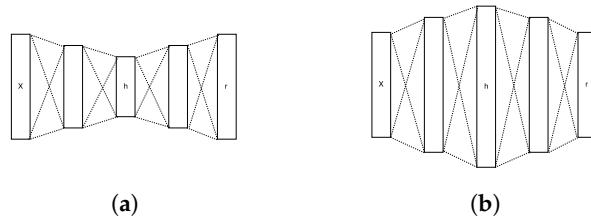


Figure 4. (a) Representation of an undercomplete autoencoder, where the representation has a lower dimension with respect to the input. (b) Representation of an overcomplete autoencoder, which uses a larger space to represent the input.

The undercomplete autoencoder is the oldest and best known architecture for the autoencoder and also the one we use in our work. The code the autoencoder derives is smaller than the input: it learns how to combine the input so that the relevant information is retained in a lower dimensional space. The combination is often non-linear due to a non-linear activation function of the neurons.

On the other hand, the decoder function learns how to unroll such compressed code in order to obtain an extended representation. This architecture could be trained using the classic back-propagation approach, although different methods are available [18].

The overcomplete autoencoder, instead, increases the dimension of the representation; however, as already mentioned, the model could easily learn the identity transformation. In order to tackle this issue, the common approach is to apply a regularization term over the weights of the network. In general, a l_1 penalization is added to the loss function. In this way, the representation will be sparse since some of the dimensions are set to zero. However, this new architecture would be more complex to train since the model would be larger than the previous one, although still providing a sparse representation. Despite its sparsity, this architecture is able to extract more information from the input, and it is often used to capture also the input distribution and not just the structure of x . It is not possible to estimate such a distribution with an undercomplete autoencoder since such a task would require a higher representation capacity.

In this work, we adopt an undercomplete autoencoder, since our goal is to represent an image in a lower dimensional space. Our architecture has multiple layers: the first layer takes only the input, while the following ones decrease in size following the power of two. We represent the input with a code having 64 dimensions; thus, from an input of 1024 features, we derive a representation of 64 components using an autoencoder.

The reconstruction of the input by the decoder function has a very low loss, below 0.05%. The layers obtained by the encoder are often difficult to understand by people; for instance, in Figure 5a–c, we can observe the first layer, and we cannot derive any structure or meaning from it. However, looking at deeper levels, the layers show a clear structure; in Figure 5g–i, we can observe that the filters are extracting horizontal and vertical patterns. Indeed, most of the original images show a cross pattern, or multiple crosses; therefore, this representation seems to be able to represent the input data with a lower dimension.

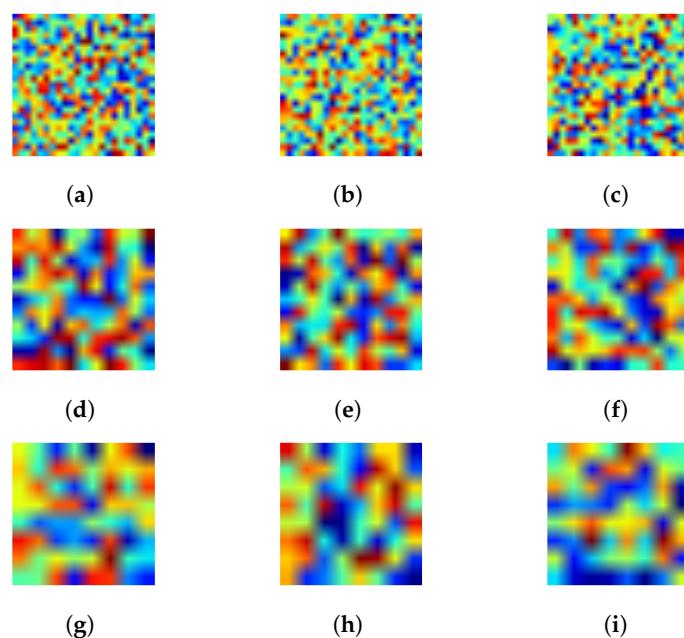


Figure 5. Example layers from the trained autoencoder. (a–c) represent the information extracted by the first layer of the autoencoder. It is hard to grasp the meaning of the representation derived. Going deeper, in (d–f), the representation seems to become more clear and with a structure. Indeed going directly to the obtained code, in (g–i), we can see a more accentuate structure composed mainly of horizontal or vertical lines.

3. Problem Statement

This section formulates the problem of modeling the player as a probabilistic mixture of uncovered player styles from data. There are at least three basic dimensions (or problems) of interest that must be considered when building a model of this kind for PIRGs. Such dimensions are related to the specific capabilities of a mobile robot to achieve a reasonable in-game performance. We delineate such dimensions as follows:

- Dimension #1: Fundamental robot behavior, which concerns the fundamental aspects of the robot locomotion, timing of actions, localization and other abilities to keep it alive and well-functioning during the game play.
- Dimension #2: In-game competitive capabilities, which are more related to the intelligent selection of actions aimed at achieving in-game goals. In another words, this dimension captures the intricacies of planning and rational behavior, using solutions for Dimension #1.
- Dimension #3: Adaptive power, which is a higher level dimension concerned with how and when during the game to adjust the robot performance to match that of the current player.

Dimension #1 concerns the setup of an appropriate robotic platform, including aspects like: electrical disturbances, power consumption, kinematic constraints, processing capabilities, sensing and safety. These are recurrent issues that must be appropriately dealt with. In turn, Dimension #2 encompasses the problem of finding a plan or strategy to achieve the robot goals. This is the problem most related to the competitive behavior the robot can impose on the human player, since action selection is driven by the tendency to maximize the robot's own pay-off. After all, the robot has to give the idea that it is at best trying to beat/help its opponent(s)/companion(s).

Dimension #3, acting as the superset for #1 and #2, defines constraints on the robot behavior such that the action selection is not longer just governed by the tendency to be competitive/helpful, but is also driven by the compelling necessity to support the player interaction, in real time, as robustly as possible.

One can see that the present work as a step towards advancements in Dimension #3. Here, we exploit data coming from a single accelerometer placed on the human player's body; thus, player movement is our primary source of information for describing the playing style and engagement.

The basic assumption is that a highly active player would naturally express his/her status by a very turbulent motion signal that is compatible with strong in-game activity. Consider, for instance, the difference in signal shapes reported in Figure 6.

Following this line of reasoning, we also assume players do not fake their engagement by unnecessarily performing strong activities, like running, when they actually are not engaged in the game or feel bored by the robot behavior. Therefore, we assume that a highly motivated player would act more intensively and frequently than a non-motivated one, thus showing a more turbulent acceleration signal profile.

In situations like PIRG, movement information is useful and should not be disregarded. Furthermore, a model that is able to extract a description of the player's movement profile for the purpose of robot behavior adaptation would generate interesting dynamics able to hide existing platform limitations (kinematics constraints, tracking problems, etc.) and boost game experience.

One important aspect not considered yet by our assumptions is the role of skill level. For example, a highly skilled player could try to minimize his/her energy expenditure by performing precise movements towards winning the game and in theory could perfectly show a low-movement profile. Moreover, these assumptions cannot directly measure player engagement in the game, given that it is also possible for a player to move not too much, but, at the same time, be highly engaged.

In summary, we are interested in investigating time series data, especially acceleration patterns as proxy and/or engagement cues (given the mentioned assumptions) towards the creation of auto-adjusting robotic companions or opponents.

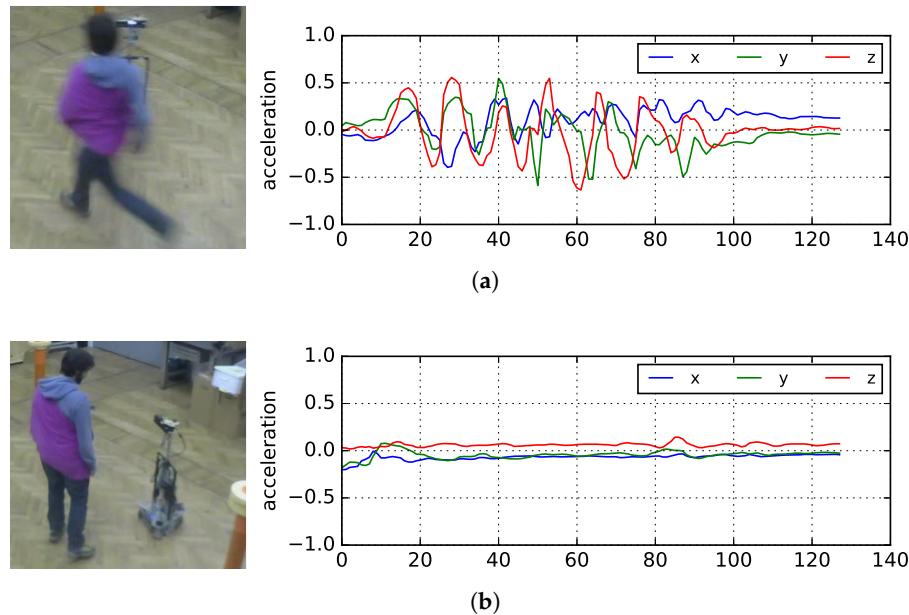


Figure 6. Three-axis accelerometer signal during game play. (a) A running player. (b) A player standing still. Notice that signal shape and amplitude are very characteristic for the respective activity.

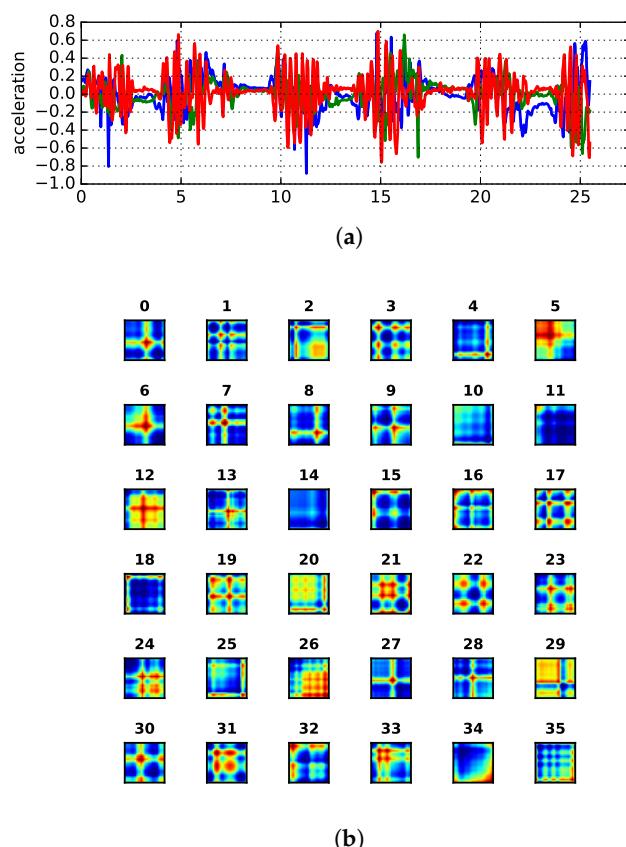


Figure 7. Example of the conversion of a time series into GASF images. (a) Three-axis accelerometer data (48.8 Hz) of about 30 seconds of real game play. (b) Thirty five GASF segments of a linear combination of the multidimensional time series. Each window comprises 0.65 seconds with 32 samples.

4. Technical Approach

This section describes our approach while emphasizing the differences and similarities with respect to existing methods. We take inspiration from approaches like [17,20,29] and aim at modeling game sessions as “documents”, where player’s acceleration data are considered as “words”. We use Latent Dirichlet Allocation (LDA) [19] to categorize the player’s motion profile as a composition or mixture of game play motion types, the latter being analogous to the “topics” in the general application of LDA for document modeling.

Differently from [20], our input word atoms are not discrete. Instead, they are continuous (acceleration data) and also are not attached to any explicit meaning. Take, for instance, the natural interpretability of joystick input signals like “left” and “right”, which convey their respective meaning when controlling a game character, and compare it with an acceleration signal that carries no obvious absolute meaning.

The adopted method for selecting the input words is based on sliding windows (just as in [20]), without overlap. Considering each game session as a document, the windowed segments play the role of words. Since the space is continuous, a procedure of discretization and dictionary learning for LDA had to be performed. For this, we clustered the segments and used the so-obtained cluster centroids as dictionary words.

We have used GASF images instead of GADF because the mapping functions of the rescaled time series in $[0, 1]$ are bijections, which allows for precise reconstruction of the original time series [17]. The reconstruction takes advantage of the main diagonal of GASF, i.e., $\{G_{ii}\} = \{\cos 2\phi_i\}$, and is performed as reported in Equation (6) [17]:

$$\cos(\phi) = \sqrt{\frac{G_{ii} + 1}{2}} \quad \phi \in [0, \frac{\pi}{2}] \quad (6)$$

An example of signal decomposition into GASF images is shown in Figure 7. Given recent advances in deep learning and pattern recognition for images, we take the route of [29] and exploit the applicability of autoencoders for extracting unsupervised features from GASF images (see Section 2.3). This is relevant since the process of feature engineering is time-consuming and laborious. Moreover, unsupervised feature extraction from such methods has been proven to work well [17,30]. Sparse encoding via linear-algebra methods for dictionary learning is also a possibility. However, in our experiments, it turned out to be not as efficient as autoencoders. We present details of our experimental activity in the next section. Our methodology is summarized in Figure 8.

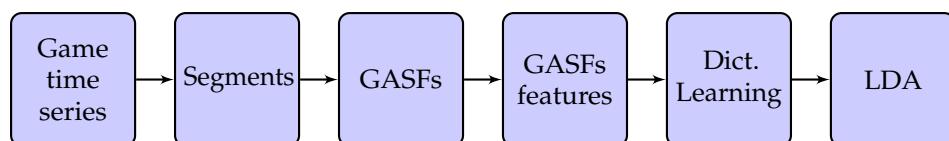


Figure 8. The general diagram for our methodology. At first, game time series are extracted, which are then segmented into windows (sliding windows method). Such segments are then turned into GASF images, whose representations are obtained using the autoencoder. Since the basic LDA method works with a multinomial distribution at the topics, we define a dictionary of visual words before feeding each GASF feature document representation to it.

5. Materials and Methods

5.1. The Robotic Platform

We have adopted a holonomic robot (Triskar), free to move in any direction at a maximum speed comparable to that of people in indoor environments (1.4 m/s). Triskar presents simultaneously

and independently controlled rotational and translational motion capabilities thanks to three omni-directional wheels, each actuated by a motor and pointing in different directions.

The kinematics of the holonomic robot is described by the following matrix equation:

$$\begin{bmatrix} V_x \\ V_y \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{\sqrt{3}}r & \frac{1}{\sqrt{3}}r \\ -\frac{2}{3}r & \frac{1}{3}r & \frac{1}{3}r \\ \frac{r}{3L} & \frac{r}{3L} & \frac{r}{3L} \end{bmatrix} \cdot \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \quad (7)$$

where r is the wheel's radius, V_x and V_y are the linear velocity components of the robot along x and y axis, $\dot{\theta}$ is the angular velocity component around the z -axis passing through the robot center of mass and, finally, ω_i [$i=1,2,3$] are the angular velocities of each wheel. We consider $[X_f, Y_f]$ the fixed (world) reference frame, while $[X_m, Y_m]$ is the moving (robot) reference frame; the robot's orientation is denoted by angle θ , which is the angle of the axis X_m with respect to X_f (positive in the counterclockwise direction), and finally, the wheel orientation in the robot coordinate system is equal to 30 degrees. Figure 9 depicts how these variables play their roles in our robot kinematics.

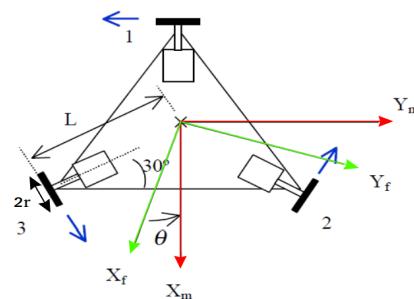


Figure 9. Kinematic representation of a holonomic base.

The robot body consists of a triangular, omni-directional base 8 cm high (diameter 40 cm), where motors, batteries and electronics are embedded. On the base, an auxiliary aluminum structure is mounted, hosting the on-board computer and a Microsoft Kinect® sensor, used for player tracking. In total, the robot is 85 cm high (see Figure 10). In our experiments, we have used the Adaptive Monte Carlo Localization (AMCL) node from the Robotic Operating System (ROS) environment, which made localization and navigation easier [31].



Figure 10. An example of the game match configuration. The human player is wearing a purple robe, which allows him to be identified by the robot. Towers are seen in orange (height 110 cm). The particular match in this figure used three towers; however, in normal conditions, the game comprises four different towers positioned on the corners of the playground; the latter being a rectangle marked on the floor.

5.2. Game Scenario

For the present work, the playground is a rectangular area of $4\text{ m} \times 2\text{ m}$, where on each corner, tubes representing towers are placed (see Figure 10). Each tower is equipped with a button and four Light Emitting Diodes (LEDs) that can be progressively turned on. Each tower LED requires the tower button to be pushed for about 2.5 s to turn on, meaning that a tower takes about 10 s to light up all four of its LEDs. Such LEDs are used as a representation of the progress of the human player in capturing a specific tower.

After turning on all LEDs of a particular tower during game play, it is said that the player has captured it. Button pressing time is cumulative and can be distributed for different moments; this means that the player will not lose his/her progress if he/she stops pushing the button before a tower is captured. Each tower has a circuit, which transmits its status to the robot.

In order to win, the human player must be able to secure all the existing towers without letting a single one be knocked down by the robot. If, at anytime, a tower falls (because of the robot or player), the game ends, and the human player loses.

Since the robot is a holonomic base, it is able to move across the entire playground just as the human player, and it is only constrained by the fact that an already captured tower, or one whose button is currently being pushed by the player, cannot be torn down. The player can also block the robot path by staying on it. Notice that, while the player is trying to capture a given tower, the robot can try to tear down another one. By relying on the laser scans, the robot can also perceive the human player during the game, so as to avoid hitting him/her while moving. Figure 11 summarizes the overall idea of the game.

When playing the game, we ask the human player to wear a colored robe in order to allow for blob detection and user tracking. These tasks are carried out by the use of a Microsoft Kinect® sensor. In the future, we plan to study the extraction of behavior features using such device, which may potentially help to understand gameplay in a more fine-grained way.

We have used a custom accelerometer board (see Figure 12) attached to the player's chest in order to capture detailed player motion information. The choice of the accelerometer position was conditioned by the need to avoid the introduction of noise from irrelevant player motion. With the sensor placed on the chest, we aimed at getting a more stable reference for motion in comparison to, for instance, data gathered from the sensor placed on the arm or thigh [13]. The custom accelerometer module is based on the [InvenSense MPU-6050](#) three-axis accelerometer board and an [Arduino Uno](#) micro-controller (see Figure 12). The circuit also contains a [Nrf24l01](#) radio-frequency module that allows accelerometer and tower data to be sent to the on-board computer.

Using the dataset from previous work [13,14], we considered 29 matches involving 15 male participants of different ages. The age distribution consisted of children (7–10) and adults (26–40). Recorded matches had a minimum duration of about 40 s and a maximum of about 1 min and 10 s. The collected data correspond to acceleration values along the x -, y - and z -axis with a sampling frequency of 50 Hz, which is five-times larger than the frequency considered to be sufficient for detecting daily activities from accelerometer data (10 Hz) [32–34].

In our experiments, we have also decided to use univariate accelerometer data instead of the multivariate three-axis information. For that, we took a linear combination of the axis favoring the z -axis, the one pointing forward from the player. The justification for this is that from our observations, very rarely, the player would run sideways or jump. Therefore, with the linear combination, we intend to collapse the multivariate axis information into a univariate signal that encompasses all the motion information more compactly; this makes the analysis easier to perform. Specifically, we considered the linear combination $\text{comb}(d) = 0.5 \times d.z + 0.25 \times d.y + 0.25 \times d.x$, where $d.x, d.y, d.z$ are acceleration samples from the x -, y -, and z -axis, respectively.

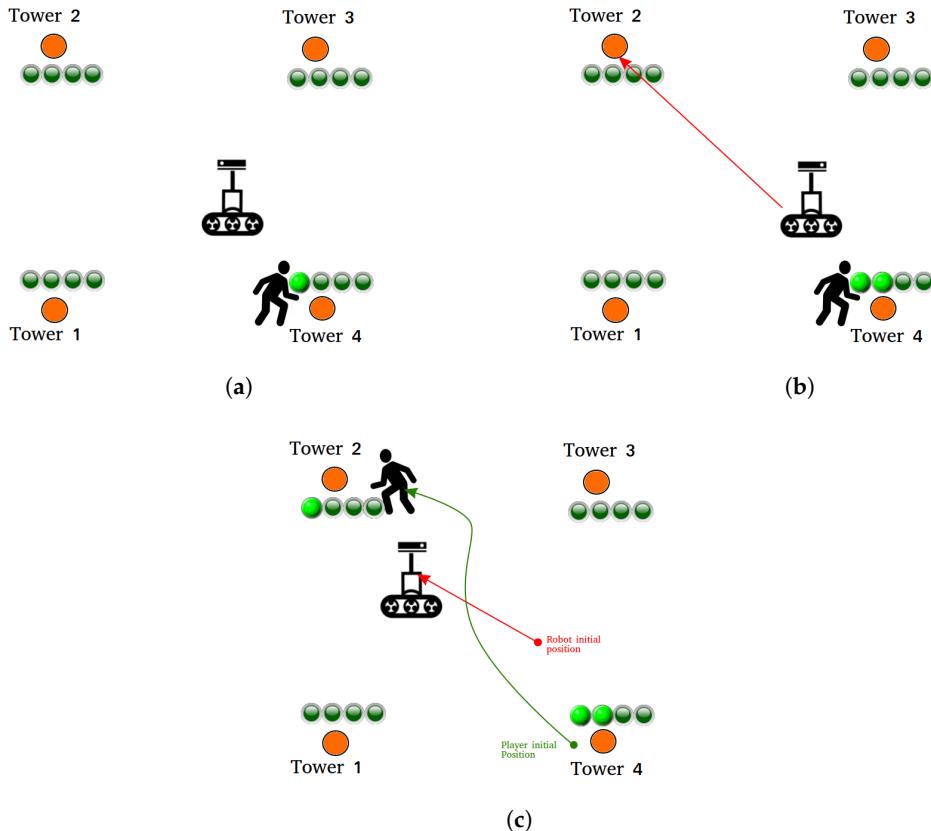


Figure 11. Schematic representation of the game. (a) The player is trying to capture Tower 4. When all four LEDs on the tower are lit, the tower is captured. (b) The robot starts moving in order to attack and tear down Tower 2. The robot cannot try to tear down the tower that is currently being captured by the player or an already captured one. (c) The human player defends Tower 2. The player stops the action of the robot by blocking its trajectory. Notice how the progress on Tower 4 is not lost even if the player stops pressing its button.

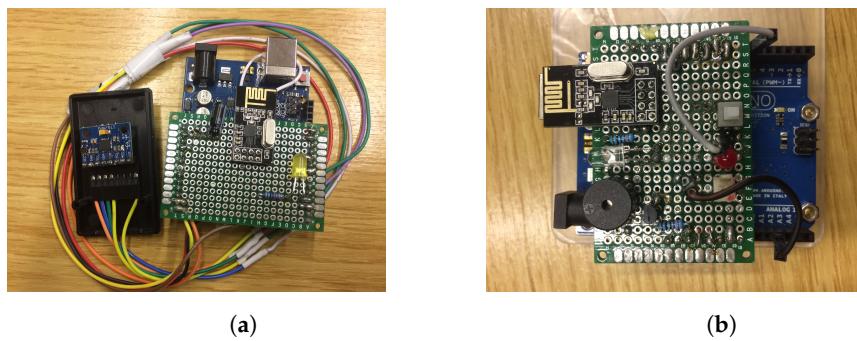


Figure 12. (a) The custom accelerometer board. (b) The on-board receiver circuit.

6. Experimental Results

In this section, the exploration of an undercomplete representation of the GASF images using a simple autoencoder architecture is firstly reported, then we discuss how we transform the extracted features into input words for the LDA model. Finally, we show the results when confronting the model

output with human judgment. All experimental code used is available at the link in supplementary materials.

6.1. Feature Extraction

The first question we need to answer concerns the size of the sliding window. This parameter regards the definition of the GASF images used as building blocks for the dictionary learning. The window size should be large enough to capture motion patterns relevant for the player characterization, but not so large to confound their interpretation. We tried different window sizes, and in the end, we followed [13] and decided to consider half a second of data per window without overlap, which allows for a fine-grained data representation.

We applied no preprocessing to the signal, since we were interested in observing how an autoencoder would perform using the unfiltered input. The expectation was that it would naturally compensate for the differences in the images, decisively choosing to capture important features (the main signal characteristic) and disregarding non-representative ones (the small noise associated). We defined an architecture composed by dropout and dense layers where only hyperbolic tangent activation functions were used, except in the last layer, where a linear activation function was used instead. The best performing architecture was composed by the following layers: dropout (0.1 drop fraction), dense (size = 256), dropout (0.2) and dense (size = 64). The same layers (in reversed order) were used for the decoder part.

A window of half a second corresponded to 32 samples given our accelerometer average frequency of 48.8 Hz. As a consequence, we obtain GASF images (without using PAA) with a total of 1024 pixels. Using the autoencoder, we manage to reduce the representation down to a 64-dimensional input, resulting in the reconstruction presented in Figure 13. The choice of a 64-dimensional representation is empirical, meaning that we were in principle searching for an undercomplete representation of the data and, upon testing with different configurations, we have chosen the one with the lowest reconstruction error. As it turned out, a 64-dimensional vector representation was the best result, given our dataset.

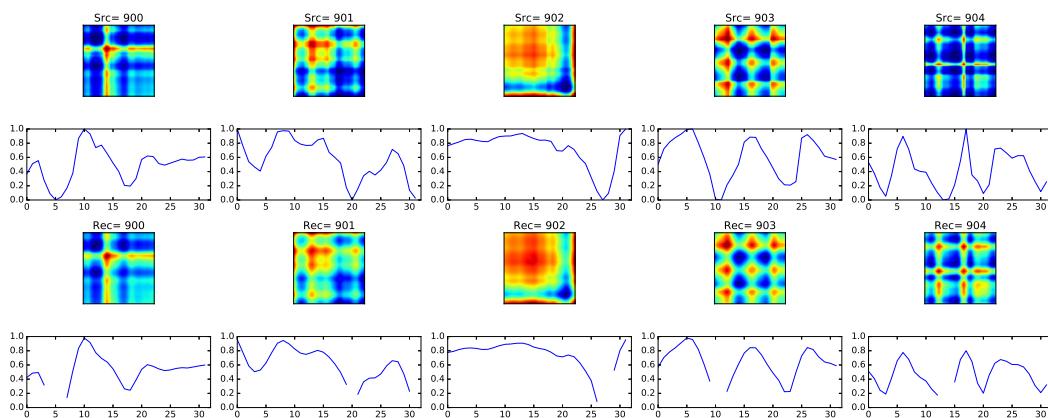


Figure 13. Autoencoder reconstruction for five input segments corresponding to half a second of data (linear combination of the acceleration axis). (Top line) Input GASF images. (Second line) Original time series for each GASF input. (Third line) The autoencoder reconstruction for the input images. (Forth line) Time series from the reconstructed GASF images.

It is possible to notice from the reconstruction result that, despite such a large compression (from 1024 to only 64 values), the input image structures are captured. This can also be seen in the reconstructed time series. A closer look reveals that the autoencoder practically smoothed the time series while preserving the overall shape, this being a nice property that can help to prevent overfitting.

From Figure 13, we can also see that the obtained time series reconstruction presents some holes. This seems to be caused by the information loss in the reconstruction and demands further investigation. However, in our experimentation, it did not seem to affect performance.

6.2. Dictionary Learning and Input Word Definition

Latent Dirichlet allocation works by considering the relationship among word tokens with respect to a collection of documents. Therefore, in order to use the method, we needed to come up with a way to encode the taxonomy of GASF images into a discrete set of fixed tokens describing the motion primitives appearing in each game session. We opted to follow [35] and encode the features extracted from the GASF using the autoencoders as visual words, composing the game session document.

After getting the 64-dimensional vector representation for each GASF image (see Section 2.3), we cluster them into V groups using the K-means algorithm. In this way, we define a dictionary of size V , with each word corresponding to a cluster centroid. Each game session is then translated into a collection of V possible clusters, which can then be fed into the LDA algorithm.

From text mining and information retrieval, we know that having documents with different sizes can be a potential issue. This is because given the bag of words representation used in LDA, a small document may be included as a subset in a larger one. Additionally, it might be a problem to evaluate longer documents in our scenario, since, at least from a human perspective, it is not simple to define the overall behavior.

In text mining, this issue is handled by imposing a weight based on the length of the document; in our case, instead, we decided to split each session into smaller segments of 15 s each. In this setting, the evaluation is more consistent, and it is possible that the behavior of a player will be maintained within segments. We decided this length for each document both for convenience with respect to the window length and from the assumption that in each fraction of 15 s, the player can show a consistent behavior, which can be somehow easily identifiable. In the next section, we present the results achieved with the approach we just presented.

6.3. Validation

Our ultimate objective is to represent the player motion style by a mixture of topic proportions following the LDA framework. Unfortunately, the method by itself demands the programmer to initialize the desired number of topics. Thus, we had to perform a search for the optimal value given our dataset.

By performing grid search, we selected a range of values for the two main hyperparameters, namely dictionary size, for the input word definition, and number of topics. We would like to keep the number of topics low, preferably less than 10, since by observing our dataset, it is not possible to distinguish many groups of different motion styles; this suggests that a coarse-grained definition would be better. Choosing the number of topics to be at most 10, as in [20], the evaluation of the uncovered topics remains manageable, while maintaining a certain degree of freedom for the existing motion diversity.

The distribution of cluster centroids (words) given topics (our player motion types) is shown in Figure 14. Each line represents a topic, while each column represents a word. As one can see, there are no topics represented by the same distribution over the same word. Figure 14 shows how much each word is important for the specific topic.

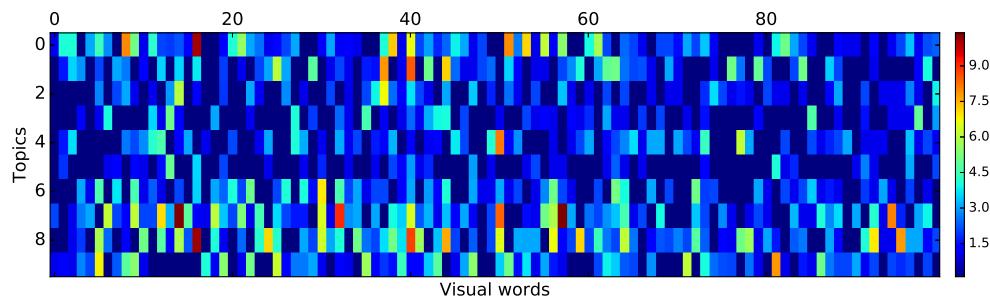


Figure 14. Schematic representation of the game: each line represents a topic, while each column represents a word that is representative of the specific topic.

In order to validate the discovered topics, we use human judgment. We invited volunteers to watch a set of pairs of recorded game segments (15 s each) and point out the similarities between players in their motivation as perceived by their motion. Subjects had to judge how similar, on a scale from 0–5, the movements of two players were. With this, we generated a similarity matrix to be used as the ground truth and main guide for deciding which hyper-parameter value to use and to check whether the proposed method could achieve reasonable results.

The similarity matrix was composed of 406 independent evaluations from six different human subjects. The generated evaluation matrix was then compared with a cosine similarity matrix generated after training the LDA in our dataset. The similarity, in this case, is based on the mixture proportions among the discovered topics.

We used this information to define the number of topics and the number of words. The best set of parameters is the one that presents a higher similarity compared to the one expressing human judgment, thus the smallest Mean Squared Error (MSE). The graph in Figure 15 shows the results obtained varying the size of the dictionary (number of cluster centroids) and the number of topics. From it, we have decided to pick 100 as the number of visual words and 10 for the topics present in our dataset.

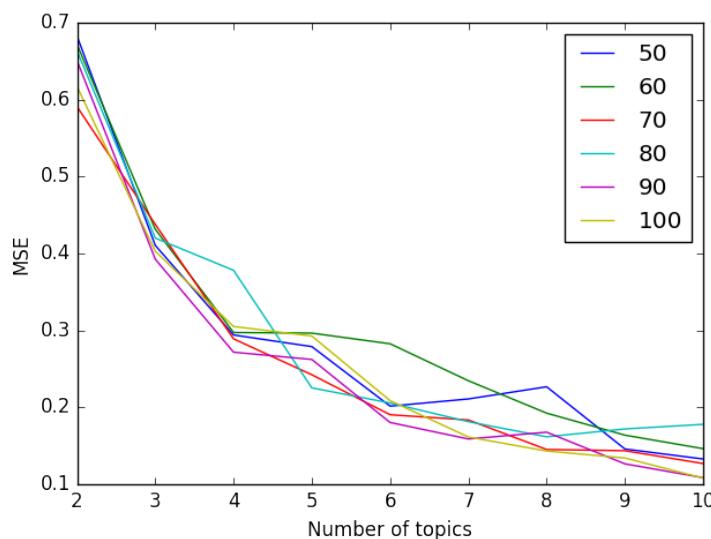


Figure 15. The figure shows the hyperparameter tuning results. The Mean Squared Error (MSE) is estimated with respect to the similarity expressed in the matrix containing the human judgment. The different lines represent the MSE obtained for different dictionary sizes and number of topics.

The observed similarity results were verified by visual inspection of video logs from estimated similar players; that is, by checking how reported mixture proportions actually look similar in video. However, we state that much work should still be performed towards easily and reliably assessing model correctness and selection. This also relates to the possibility of confidently giving names to discovered motion types, very much in the way as is done in classic LDA for text categorization.

Another aspect that demands further investigation is how to avoid the decrease in accuracy when considering variable-sized game sessions. As mentioned before, game sessions with different sizes may artificially increase similarity in case one is a subset of another. The solution to this problem provides only a baseline, and much more sophisticated approaches to the problem should be devised.

We believe that a potential way for improvements in interpretability and accuracy is that of removing the vector quantization in defining each topic. It would be interesting to remain in the continuous domain defined by our input data representation (GASF features), instead of relying on a fixed predefined vocabulary set used for training the LDA. Such vectorization is known to lead to information loss [36]. In avoiding such an issue, one possibility may be replacing the topic multinomial distribution by a mixture of Gaussians [36].

7. Discussion

In this work, we introduced an approach to identify the player motion styles from measurements obtained by the use of a single three-axial accelerometer sensor. Such a signal is then transformed using the Gramian angular field in an image that represents its content. The reason for such representation stems from recent publications aiming at using deep learning and state-of-the-art image processing techniques for achieving better classification results. Additionally, using the autoencoder for representing the obtained GASFs reduced the workload necessary for hand-crafting features, as often done when using machine learning classification/clustering methods with time series data.

By applying latent Dirichlet allocation to characterize the motion behavior of different players, we were interested in getting a description that would foster the development of self-adjusting PIRGs agents capable of improving/sustaining engagement. In this, we had shown that the model is able to capture the overall human judgment by matching reported similarities.

This is the first work, to our knowledge, that tries to model the motion behavior of the player using this type of signal in conjunction with LDA in a robogame scenario. We have demonstrated that the proposed approach is able to cluster and represent different behaviors learned from data. We have evidence that our methodology can achieve better performances than that of relying solely on the measurements of the accelerometer as input for the dictionary learning step. We are still working to provide a fair comparison between the two approaches.

As future work, we want to test our approach with a series of different input signals in order to assess its capability. One alternative could be the incorporation of proximity information in order to capture how proximity patterns relate to a player game play style. Moreover, we are interested in measuring the player's effort expenditure in terms of difficulty of motion given a difficult setting for the robot.

In our method, we also wish to incorporate time dependency. We believe such information would help to describe variations in playing style during play, given the opportunity to detect change points in styles and favor game (robot behavior) personalization. This also may help to avoid the rubber-band effect, a system instability concern often present when designing auto-adjusting game systems.

Another means for improvements, and one that we are currently working on, is that of removing the vector quantization step necessary for the definition of the multinomial distribution of words that defines each topic. We believe that it would be interesting to remain in the continuous domain defined by our input data, instead of relying on a fixed predefined set of tokens, i.e., the vocabulary set used for training the LDA. Our current experimentation goes along the lines of testing how a Gaussian version of LDA would work in our domain and whether that could give better results. Additionally, comparing our results with a continuous version of LDA would help to answer questions regarding

stop word filtering, such as: To what extent removing certain common segments would improve or worsen accuracy?

Finally, we are working to redesign the experiments in order to allow for collecting user survey data, which comprise a valuable resource regarding player style and engagement levels. This is all in order to increase our understanding in interpreting the obtained results. We plan to collect self-reported engagement levels when comparing with different robot behavior configuration while allowing for adaptation. Regarding such adaptation, we currently are investigating how to relate the robot game's difficult settings (such as velocity, tower LED control rules, among others) and the mixture proportion of motion types discovered by our reported method. One alternative is to design a utility function that would describe what is the pay-off in facing a certain player given a certain difficult setting, therefore turning the adaptation problem into a full on-line optimization procedure where we try to match player motion types with robot behavior implied by difficulty settings.

Supplementary Materials: The experimental code is available at <https://github.com/ewerlopes/lda-player-model>.

Acknowledgments: This work was supported by the Brazilian National Council for Scientific and Technical Development (CNPq) under the Science Without Borders scholarship program Number 203677/2014-5.

Author Contributions: E.L.S.O. developed the hypothesis, experiments, data collection and analysis. He also wrote the paper. D.O. contributed to the data collection, robot control and paper writing. L.M. performed the experiments, data analysis and paper writing. A.B. supervised the research and contributed to the overall development of the reported results. He also reviewed and corrected the work. T.P.N. provided some support in writing the paper.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this work:

GADF	Gramian Angular Difference Field
GAF	Gramian Angular Field
GASF	Gramian Angular Summation Field
LDA	Latent Dirichlet Allocation
LMA	Laban Movement Analysis
MSE	Mean Squared Error
PAA	Piecewise Aggregation Approximation
PIRG	Physically Interactive Robogame
VR	Virtual Reality

References

1. Zyda, M. From Visual Simulation to Virtual Reality to Games. *Computer* **2005**, *38*, 25–32.
2. Stanney, K.M.; Mourant, R.R.; Kennedy, R.S. Human Factors Issues in Virtual Environments: A Review of the Literature. *Presence Teleoper. Virtual Environ.* **1998**, *7*, 327–351.
3. Martinoia, D.; Calandriello, D.; Bonarini, A. Physically Interactive Robogames: Definition and design guidelines. *Robot. Auton. Syst.* **2013**, *61*, 739–748.
4. Di Tore, A.; Raiola, G. Exergames e didattica delle attività motorie e sportive. *Eur. J. Sustain. Dev.* **2012**, *1*, 221.
5. Di Tore, P.A.; Raiola, G. Exergame-design and motor activities teaching: An overview of scientific paradigms on motor control. *Mediterr. J. Soc. Sci.* **2012**, *3*, 119–122.
6. Di Tore, P.A.; Raiola, G. Exergames in motor skill learning. *J. Phys. Educ. Sport* **2012**, *12*, 358.
7. Huang, H.C.; Wong, M.K.; Lu, J.; Huang, W.F.; Teng, C.I. Can using exergames improve physical fitness? A 12-week randomized controlled trial. *Comput. Hum. Behav.* **2017**, *70*, 310–316.
8. Cruickshank, V.; Cooley, D.; Pedersen, S.J. *Exergames: Good for Play Time, but Should Not Replace Physical Education*; The Conversation Media Trust: Melbourne, Australia, 22 September 2017.

9. Bonarini, A. Timing Issues in Physically Interacting Robogames. In Proceedings of the Timing in Human-Robot Interaction Workshop at 9th ACM/IEEE Conference on Human-Robot Interaction (HRI 2014). Available online: <http://home.deib.polimi.it/outgoing/Andrea.Bonarini/Papers/BonariniRoboGameTiming.pdf> (accessed on 19 December 2017).
10. Fujita, M.; Kageyama, K. An open architecture for robot entertainment. In Proceedings of the First International Conference on Autonomous Agents, Marina del Rey, CA, USA, 5–8 February 1997; ACM: New York, NY, USA, 1997; pp. 435–442.
11. Brooks, A.G.; Gray, J.; Hoffman, G.; Lockerd, A.; Lee, H.; Breazeal, C. Robot’s play: Interactive games with sociable machines. *Comput. Entertain. (CIE)* **2004**, *2*, 10–10.
12. Yannakakis, G.N.; Hallam, J.; Lund, H.H. Entertainment capture through heart rate activity in physical interactive playgrounds. *User Model. User Adapt. Interact.* **2008**, *18*, 207–243.
13. Oliveira, E.; Orrù, D.; Nascimento, T.; Bonarini, A. Activity Recognition in a Physical Interactive RoboGame. In Proceedings of the 2017 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics, ICDL-EpiRob 2017, Lisbon, Portugal, 18–21 September 2017; pp. 1–6.
14. Oliveira, E.; Orrù, D.; Nascimento, T.; Bonarini, A. Modeling Player Activity in a Physical Interactive Robot Game Scenario. In Proceedings of the 5th International Conference on Human Agent Interaction, Bielefeld, Germany, 17–20 October 2017; ACM: New York, NY, USA, 2017; pp. 411–414.
15. Lund, H.H.; Klitbo, T.; Jessen, C. Playware technology for physically activating play. *Artif. Life Robot.* **2005**, *9*, 165–174.
16. Yannakakis, G.N.; Lund, H.H.; Hallam, J. Modeling Children’s Entertainment in the Playware Playground. In Proceedings of the 2006 IEEE Symposium on Computational Intelligence and Games, Reno, NV, USA, 22–24 May 2006; IEEE: Reno, NV, USA, 2006; pp. 134–141.
17. Wang, Z.; Oates, T. Imaging Time-Series to Improve Classification and Imputation. *ArXiv* **2015**, *1506*, arXiv:1506.00327.
18. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
19. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
20. Smith, B.A.; Nayar, S.K. Mining Controller Inputs to Understand Gameplay. In Proceedings of the 29th Annual Symposium on User Interface Software and Technology, Tokyo, Japan, 16–19 October 2016; ACM: New York, NY, USA, 2016; pp. 157–168.
21. Aristidou, A.; Charalambous, P.; Chrysanthou, Y. Emotion analysis and classification: Understanding the performers’ emotions using the LMA entities. *Comput. Gr. Forum* **2015**, *34*, 262–276.
22. Shafir, T.; Tsachor, R.P.; Welch, K.B. Emotion regulation through movement: Unique sets of movement characteristics are associated with and enhance basic emotions. *Front. Psychol.* **2016**, *6*, 2030.
23. Tsachor, R.P.; Shafir, T. A Somatic Movement Approach to Fostering Emotional Resiliency through Laban Movement Analysis. *Front. Hum. Neurosci.* **2017**, *11*, 410.
24. Laban, R. *The Language of Movement: A Guidebook to Choreutics*; Plays Inc.: Boston, MA, USA, 1974.
25. Burton, S.J.; Samadani, A.A.; Gorbet, R.; Kulić, D. Laban movement analysis and affective movement generation for robots and other near-living creatures. In *Dance Notations and Robot Motion*; Springer: Cham, Switzerland, 2016; pp. 25–48.
26. Zacharatos, H.; Gatzoulis, C.; Chrysanthou, Y.; Aristidou, A. Emotion recognition for exergames using laban movement analysis. In Proceedings of the Motion on Games, Dublin, Ireland, 6–8 November 2013; ACM: New York, NY, USA, 2013, pp. 61–66.
27. Keogh, E.J.; Pazzani, M.J. Scaling up dynamic time warping for datamining applications. In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, USA, 20–23 August 2000; ACM: New York, NY, USA, 2000; pp. 285–289.
28. Reed, C. *Latent Dirichlet Allocation: Towards a Deeper Understanding*; Technical Report; The Pennsylvania State University: State College, PA, USA, 2012. Available online: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.399.2859> (accessed on 19 December 2017)
29. Wang, Z.; Oates, T. Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks. In Proceedings of the Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–26 January 2015.
30. Wang, Z.; Yan, W.; Oates, T. Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline. *arXiv* **2016**, arXiv:1611.06455.

31. Ballardini, A.L.; Fontana, S.; Furlan, A.; Sorrenti, D.G. *ira_laser_tools*: A ROS LaserScan manipulation toolbox. *arXiv* **2014**, arXiv:1411.1086.
32. Atallah, L.; Lo, B.; King, R.; Yang, G.Z. Sensor placement for activity detection using wearable accelerometers. In Proceedings of the 2010 International Conference on Body Sensor Networks (BSN), Singapore, 7–9 June 2010; pp. 24–29.
33. Ravi, N.; Dandekar, N.; Mysore, P.; Littman, M.L. Activity Recognition from Accelerometer Data. In Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence, Pittsburgh, PA, USA, 9–13 July 2005; AAAI Press: Pittsburgh, PA, USA, 2005; pp. 1541–1546.
34. Kikhia, B.; Gomez, M.; Jiménez, L.L.; Hallberg, J.; Karvonen, N.; Synnes, K. Analyzing body movements within the laban effort framework using a single accelerometer. *Sensors* **2014**, *14*, 5725–5741.
35. Prince, S.J.D. *Computer Vision: Models, Learning, and Inference*, 1st ed.; Cambridge University Press: New York, NY, USA, 2012.
36. Hu, P.; Liu, W.; Jiang, W.; Yang, Z. Latent topic model based on Gaussian-LDA for audio retrieval. In Proceedings of the Chinese Conference on Pattern Recognition, Beijing, China, 24–26 September 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 556–563.



© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).