Sign Language Support – Adding a Gesture Library to the Leap Motion SDK

Tiago Lopes, Tiago Cardoso(✉), and José Barata

Faculdade de Ciências e Tecnologia,
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
tm.lopes@campus.fct.unl.pt, {tomfc,jab}@uninova.pt

Abstract. There are several research initiatives that tackle gesture recognition. Nevertheless the interaction between the input devices and an application level is still a hard task that has to be accomplished each time a new system is being developed. The objective of this research work is to facilitate that endeavor by introducing a new generic software layer between the gesture capture device and the application level. This layer will provide the introduction of a gesture library and a set of functionalities both to feed this library and pursue gesture recognition afterwards. The objective is to hinder lower-level software/hardware details from a developer towards letting him or her to focus directly at the Application Level. This article presents the created architecture for this new layer. The validation was made using the Leap Motion, at the Sensor Level, and creating a Serious Game devoted to Sign Language exercising, at the Application Level.

Keywords: Natural user interface · Gesture recognition · Serious games · Leap motion · Middleware

1 Introduction/Motivation

Serious Games might be seen as a knowledge distribution method that keep the players focused on a problem and allows them to overcome it. This gives a reward satisfaction and a background knowledge that continues with the player long after the game session ends. The interaction between the user and an application is a factor to have in account, especially in the particular case of Serious Games that expect the player to feedback through gestures. Several examples of this need may be identified, especially when one consider games focused on teaching Sign Languages (SL). These do require the player to make gestures that a typical input devices, like a game controller, a keyboard, or a mouse are not able to capture.

There are already some sensors in the market that have the ability to capture hand position information, facilitating the interaction between users and software applications. Nevertheless two major drawbacks may still be identified: 1 - this information is provided in a proprietary schema dependent on the sensor Software Development Kit (SDK) and 2 – too few gestures are provided at the firmware level and a mechanism to store and recognize new gestures is not provided. In other words, this data, comes with a low abstraction level, introducing the need to implement a gesture recognition algorithm to translate raw data in known gestures. The developer, by choosing a sensor,

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2017
C. Vaz de Carvalho et al. (Eds.): SGAMES 2016, LNICST 176, pp. 109–116, 2017.
DOI: 10.1007/978-3-319-51055-2_14

locks himself in a dilemma that obligates him into modifying the low level data alongside with the implementation of an interpretation algorithm and the creation/maintenance of a database storing known gestures.

This research work proposes a solution for such dilemma by creating a SDK extension that hides the low level information by transforming it into a known and common format with a higher level of abstraction that can then be used in the Application Level. This solution is not only responsible for the representation of the sensor data but also for managing a gesture library and the usage of a classification algorithm for the data it is working with. This means that an Application Level developer only needs to choose a sensor, add this SDK extension and use its common gesture information for classification/recognition to develop his or her application. This is explained in more detail on Sect. 3.

The research work presented in this article is three-folded: 1st the sensors that form the basis for the gesture SDK extension proposed, in Sect. 2.1; 2nd the role of the games in serious purposes like education, Sect. 2.2 and 3rd the Portuguese Sign Language, on Sect. 2.3, that was used at the Application Level.

To validate the Proposed Architecture a Serious Game to exercise Portuguese Sign Language was developed. This game was tested with six Portuguese SL static signs achieving satisfactory results, for further detail refer to Sect. 4.

2 Related Work

In compliance with the proposed objective, this section provides an overview on some sensors that can be used to capture gestures, the role of Serious Games as a teaching aid and how the Portuguese Sign Language is structured.

2.1 Natural User Interfaces

As stated in [1] "The mouse and the keyboard are being replaced by touch and motion based interfaces, increasingly known as Natural User Interfaces (NUI)". Some NUI sensors that have the required capabilities to solve some of the objectives proposed for this research work include:

The Microsoft Kinect – This sensor is capable to detect and track hand and body movements. This features were explored in detail in various projects such as the ones developed by [2–4]. Currently this sensor has an SDK that allow development in applications for Microsoft's Windows OS and Xbox game consoles.

Leap Motion – Leap Motion was particularity design to capture hands and small tools in its field of view. It also has greater precision than the Microsoft's Kinect as studied in [5], with some issues with finger occlusion in certain hand configurations. Leap Motion comes with a very small form factor (80 mm × 30 mm × 11,25 mm). The current SDK provides support for Windows, Mac OS X, Linux and a beta version for Android. One project that demonstrates its capabilities is the communication tool (gesture - sound) provided by the company MotionSavvy [6].

Myo - Myo is an armband that captures gestures by measuring electrical activity from the user's muscles, through the use of electromyography sensors, and data provided by accelerometers, gyroscopes and magnetometers [7]. Being an armband it has a circular shape with a circumference that can go from 19 to 34 cm and it connects to a computer (Windows or Mac OS X) or a smartphone/tablet (iOS or Android) wirelessly over Bluetooth.

2.2 Role of Games as a Learning Tool

As early as 1992 there are research works that show the importance and impact of Serious Games as a learning tool. As stated in [8] "Because games require the active participation of students, the material has a greater chance of being integrated into the cognitive structures of the individuals and thus being retained". A game is not just a tool for entertaining but one with value for teaching its users about different types of subjects. There have been multiple studies, [9–11], that demonstrate the success of games as a teaching tool.

Nowadays games are not restricted to boards, fixed consoles or computers but available everywhere, facilitated by the wide spread of smartphones and tablets increasing the ease of access to games even more. Because of this wide potential the development of Serious Games for teaching purposes gained increasing attention, namely to teach sign language. [2, 3, 12], show two examples exercising the gestures of the Portuguese Sign Language (PSL) through a game.

2.3 Portuguese Sign Language (PSL)

The PSL is the aggregation of different types of gestures [13] which involve:

- The hand configuration, the shape that the hand assumes during a gesture;
- Hand movement, the motion involving the hand and/or fingers;
- Non-hand expressions, marking made with the cheek, mouth, tongue or teeth;
- Localization, the local were the gesture is being made;
- Orientation, the direction the gesture hand is facing during its movement.

For a gesture to be classified in gender, number or verb form, a classifier type gesture is used [13]. This gesture is preceded by the gesture to classify. In PSL it is also possible to state an interrogation, a declaration, or an exclamation by using non-hand expressions [13], normally this is achieved by using face or shoulder expression.

The PSL also has signs that represent the individual letters and numerals. This is commonly used to spell names [14].

3 Proposed Architecture

The proposed architecture for solving the mentioned problem makes use of three modules and two support classes that are integrated in a single layer – SDK Extension. The placement of this architecture in an application can be observed in Fig. 1. A class diagram with some of the most important routines can be found on Fig. 2.

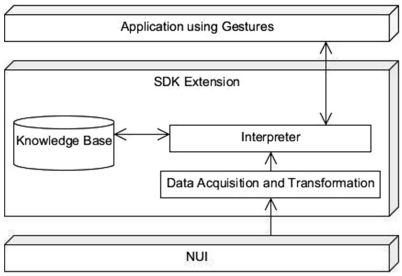

Fig. 1. Use case of the proposed architecture.

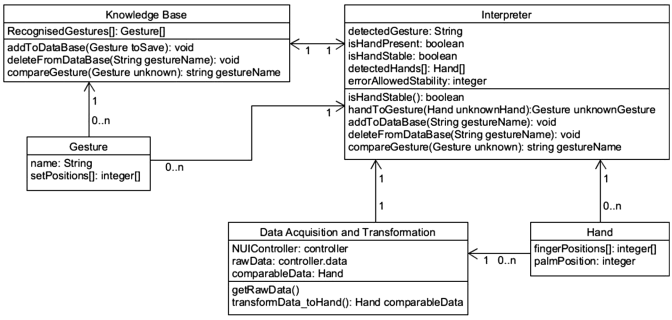

Fig. 2. Class Diagram for the proposed solution.

3.1 Knowledge Base Module

This module is responsible for managing the known gesture patterns in a pre-set format (Gesture Class) in a database. It is also responsible for storing the rules that allow a gesture to be classified, meaning that the gesture recognition algorithm (GRA) should be set in this module. The chosen GRA (the `compareGesture` function) is a Nearest Neighbor algorithm that receive has inputs the unknown Gesture and compares it to all the stored Gestures, returning the name of the most similar gesture.

3.2 Data Acquisition and Transformation Module

The DAT module is the one responsible for direct interaction with the chosen NUI. It requests raw data using `getRawData()`, in the case of the Leap Motion, by polling

the most recent Frame captured by the sensor. This Frame is then processed by `transformData_toHand()` which creates a Hand information holder with the 3 dimensional positions of the fingertips and palm of the captured hand. This data is then made available to the Interpreter module.

The transformation made by this module hides the NUI raw information from the Interpreter and by extension to all the higher abstraction level applications. With the proposed architecture this is the only module that needs to be changed if it is to be used with a different NUI.

3.3 Interpreter Module

This is the nuclear module that aggregates and processes all the available data into a meaningful information to be later used. It is the liaison between the processed data and the Application level.

The Interpreter receives the processed data from the DAT module and uses it for multiple objectives:

- Transforming the received Hand data into a Gesture format. This is done by the method: `handToGesture(Hand unknownHand)` which takes the fingertip positions and calculates the distances between all the fingers returning a Gesture with no classification but ready to be compared.
- Classifying the captured gesture through the Knowledge Base `compareGesture` method and making this classification available to the Application level through the `detectedGesture` object. Alongside this gesture classification it also provides information about the hands stability, by using the `isHandStable()` method, judging the stability by calculating the differences between the captured Hand and the previously stored Hands. If, and only if, the difference is smaller than a pre-defined threshold the hand is stable.
- Adding, deleting or modifying the gesture data in the Knowledge Base. This allows the Application level to add new specific gestures. Making it more versatile by not locking the developer to a set of pre-established gestures. Adding a new gesture requires the user to choose a name and make that gesture multiple times capturing multiple Hand data sets. These are then averaged and transformed into a single Gesture object to be added to the KB.

3.4 Support Classes

For support and information exchange, in this solution, two support classes were defined:

- Gesture class – responsible for storing the gesture information. This support class stores an array of values, representing distances between fingers, which are the key values to a gesture. It also has a text field to hold the gesture name.
- Hand class – this class contains all the data required for a loyal representation of the hand to be used by the Application level. It provides the information of the fingertips and palm positions in a 3 dimensional environment. This class makes the representation of captured hands independent from the NUI.

4 Validation

To validate the proposed solution a prototype application has been developed for Android with the objective of exercising PSL. This prototype uses the Leap Motion NUI (LM NUI) because of its precision, the availability of an Android SDK, and the portability given by its reduced dimensions.

4.1 Proposed Architecture Validation

The validation of the proposed architecture was done by implementing the architecture as a middleware. This was later tested by executing multiple gesture to test the interaction and recognition capabilities.

The DAT module transforms the data polled from the Leap Motion and recovers from it the fingertips and palm x, y and z coordinates of the hand captured in the Frame, which are then made available to the Interpreter as a Hand Object as explained in Sect. 3.2. In the KB module a Nearest-Neighbor algorithm has been implemented as the GRA and a set of five gestures (the PSL letters: 'b', 'i', 'm', 't' and 'u') were saved for comparisons. These values that are to be compared against and stored in the KB are the ten distances between every two fingers. The stored values were obtained by averaging multiple Hand objects of the same gesture done at various distances from the NUI. The Interpreter modules polls, at each frame of the game, the hand information (Hand object) made available by the DAT module and queries the KB for the gesture classification. This information, is then made available to the Application level.

The tests which this prototype was submitted to, consisted on a user performing one of six gestures, chosen randomly, ten times at different distances from the sensor. The six gesture set consisted on the five gestures that can be classified plus one not loaded in the KB, the 'v' sign, due to its similarities with 'u' and 'm'.

The results can be seen in Table 1. As one can observe, the results are satisfactory, with the lowest success rate being 70%. It is noteworthy that the results were obtained after the hand is stable which, in a small subset of gestures, due to the workings of the Leap Motion SDK, took a reset of the gesture (a reset consists on the user taking his hand from the LM capture field and putting it back in it). The gesture 'i' is an example of this, which due to the hand being somehow closed, made the tracking software lose the hand.

Table 1. Confusion Matrix with the test results.

		Gesture performed					
		b	i	m	t	u	v
Gesture recognised	b	8	0	0	0	0	0
	i	0	7	0	0	0	0
	m	0	0	8	0	0	0
	t	0	0	0	9	0	0
	u	0	0	0	0	10	2
Not recognised		2	3	2	1	0	8
Success rate (%)		80	70	80	90	100	80

4.2 Game Validation (Application Level)

The developed game consisted on a simple gesture detection, with which the user interacts with the game by placing his/hers hand in the air near the Leap Motion, and performing gestures. These gesture are compared in real time with the ones loaded on the KB and the player is informed of what PSL letter it represents. This evaluation type game is not very interesting or challenging for the player, a situation to be solved in future work with more challenging and educational game modes.

However it can be evaluated that the user only needs to load the game in the smartphone, plug the Leap Motion to it and make a gesture near its sensor. This demonstrates the ease of use of this solution, refer to Fig. 3 and you can observe the application being used.

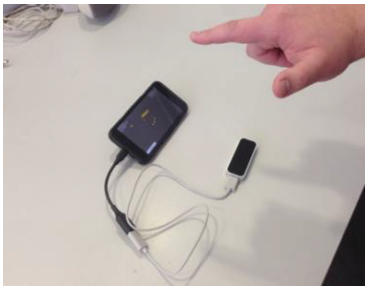

Fig. 3. User interacting with the developed prototype (by making the gesture 't').

The game, with this solution implemented, had a satisfactory response time between the user making a gesture and its classification. There are some cases where the hand was not detected by the NUI. This latter issue is due to the beta stage of the LM Android SDK as it performs better in a laptop version with a stable LM Windows SDK.

5 Conclusion and Future Work

The work presented here intends to demonstrate the increase in value that is brought by the proposed architecture. It has been demonstrated that the proposal can be implemented and that a prototype application can be built upon it. The objective of gesture recognition, as demonstrated in the validation section, is within acceptable values. Managing a knowledge base with gestures has been made transparent to the application developer completing another objective of this proposal, but there is still work to be done as this is an ongoing project.

5.1 Future Work

To improve on the gesture recognition the rest of the static gestures of the PSL will be added to the KB and there are improvements to be made to allow the recognition of gestures that involve motion. The game will also be improved to be more appealing and interactive for its players.

After these improvements, this prototype will be deployed for tests in a real use case scenario. This will be made possible with the help of the *Instituto Jacob Rodrigues Pereira*, a school dedicated in teaching Portuguese Sign Language to children and young adults based in Lisbon.

References

1. Câmara, A.: Natural User Interfaces. In: Campos, P., Graham, N., Jorge, J., Nunes, N., Palanque, P., Winckler, M. (eds.) INTERACT 2011. LNCS, vol. 6946, p. 1. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23774-4_1
2. Gameiro, J.M.F.: About using Serious Games to teach (Portuguese) Sign Language (2014)
3. Galveia, B.M.G.D.: Extensão do SDK do Kinect : Criação de uma Biblioteca de Gestos (2014)
4. Zhang, Z.: Microsoft kinect sensor and its effect. IEEE Multimed. **19**(2), 4–10 (2012)
5. Weichert, F., Bachmann, D., Rudak, B., Fisseler, D.: Analysis of the accuracy and robustness of the leap motion controller. Sens. (Basel) **13**(5), 6380–6393 (2013)
6. MotionSavvy, Uni. http://www.motionsavvy.com/. Accessed 06 May 2016
7. Thalmic Labs Inc., Tech Specs | Myo Gesture Control Armband. https://www.myo.com/techspecs. Accessed 02 Feb 2016
8. Randel, J.M., Morris, B.A., Wetzel, C.D., Whitehill, B.V.: The effectiveness of games for educational purposes: A review of recent research (1992)
9. Papastergiou, M.: Digital game-based learning in high school computer science education: impact on educational effectiveness and student motivation. Comput. Educ. **52**(1), 1–12 (2009)
10. Camerer, C.F.: Sophisticated experience-weighted attraction learning and strategic teaching in repeated games. J. Econ. Theory **104**(1), 137–188 (2002)
11. Erhel, S., Jamet, E.: Digital game-based learning: impact of instructions and feedback on motivation and learning effectiveness. Comput. Educ. **67**, 156–167 (2013)
12. Gameiro, J., Cardoso, T., Rybarczyk, Y.: Kinect-sign, teaching sign language to 'listeners' through a game. Proc. Technol. **17**, 384–391 (2014)
13. Escola Virtual de Língua Gestual Portuguesa, Introdução à Língua Gestual Portuguesa. http://www.lgpescolavirtual.pt/index.php?module=modulo&m=1. Accessed 06 May 2016
14. Associação Portuguesa de Surdos, Informação - Língua Gestual. http://www.apsurdos.org.pt/index.php?option=com_content&view=article&id=41&Itemid=56. Accessed 24 Oct 2015