Real-time 3D Gesture Recognition using Dynamic Time Warping and Simplification Methods

Alan dos Santos Soares
Federal University of Bahia, Brazil
Av. Adhemar de Barros, Ondina
40.170-110, Salvador, Bahia
alan.soares@ufba.br

Antonio L. Apolinário Jr
Federal University of Bahia, Brazil
Av. Adhemar de Barros, Ondina
40.170-110, Salvador, Bahia
antonio.apolinario@ufba.br

ABSTRACT

The recognition of dynamic gestures of hands using pure geometric 3D data in real-time is a challenge. RGB-D sensors simplified this task, giving an easy way to acquire 3D points and track them using the depth maps information. But use this collection of raw 3D points as a gesture representation in a classification process is prone to mismatches, since gestures of different people can vary in scale, location and velocity. In this paper we analyze how different techniques of simplification and regularization can provide more accurate representations of the gestures. Using Dynamic Time Warping (DTW) as the classification method, we show that the simplification and regularization steps can improve the recognition rate and also reduce the time of gesture recognition.

Keywords

Hands Gesture; Geometric Modeling; 3D Gesture Classification; Real-Time; Gesture Acquisition;

1 INTRODUCTION

The use of hand gestures in the construction of computer systems has many challenges [Pal+13]. For example, a single gesture can have different meanings depending on the culture of each country or region [HK12]. Furthermore gestures of different people can vary in scale, location and velocity.

The complexity of a gesture depends on the amount of body parts used in the movement [Pal+13], so it is necessary to define a descriptor or method to simplify the gesture in such a way that only key points are stored to improve the performance of the recognition. Our work provides an approach that recognizes gestures in real-time, regardless of position, lighting and physical aspects of the user. We define gesture as sequence of hand positions performed in the 3D space like "let's go", "bye bye", etc. We evaluated the recognition rate and performance using different combinations of regularization and simplification methods.

The main contributions of this paper are:

- A purely geometrical approach to gesture recognition.
- A method to recognize gestures in sequence without human intervention, requiring only an time interval between the executions.
- A comparison of different methods for curve simplification, showing that a step of pre-processing can reduce about a half the time consumption needed to recognize gestures.
- Support to recognize gestures with one or both hands, trained or not.

As an secondary contribution, we create a new dataset composed of depth, image and tracking information for 7 gestures performed by 7 people with different physical aspects, totalizing a set of 1099 executions.

This article is structured as follows: in Section 2 we discuss the advantages and drawbacks of related works. In Section 3 we show our proposed solution in detail. In Section 4 we present the results of the evaluation of different simplification and regularization methods through the performance and recognition aspects. In Section 5 we state our final remarks and suggest some future works.

2 RELATED WORK

Many works for gesture recognition have been published in the last few years [RA15] [Che+13] [HK12] [MA07].

The gesture recognition approaches can be divided in three main: glove-based, vision-based and depth-based. The glove-based approach uses a device to capture the 3D information (position and orientation) about hands or fingers directly, having the advantage of less input data and high speed [Bar+15]. However, the device has to be used all the time, besides it has a lot of cables and is considered more invasive [HK12]. Vision-based approaches are less invasive that the glove-based ones because the user does not need to use wearable devices [RA15]. However, vision-based approaches have

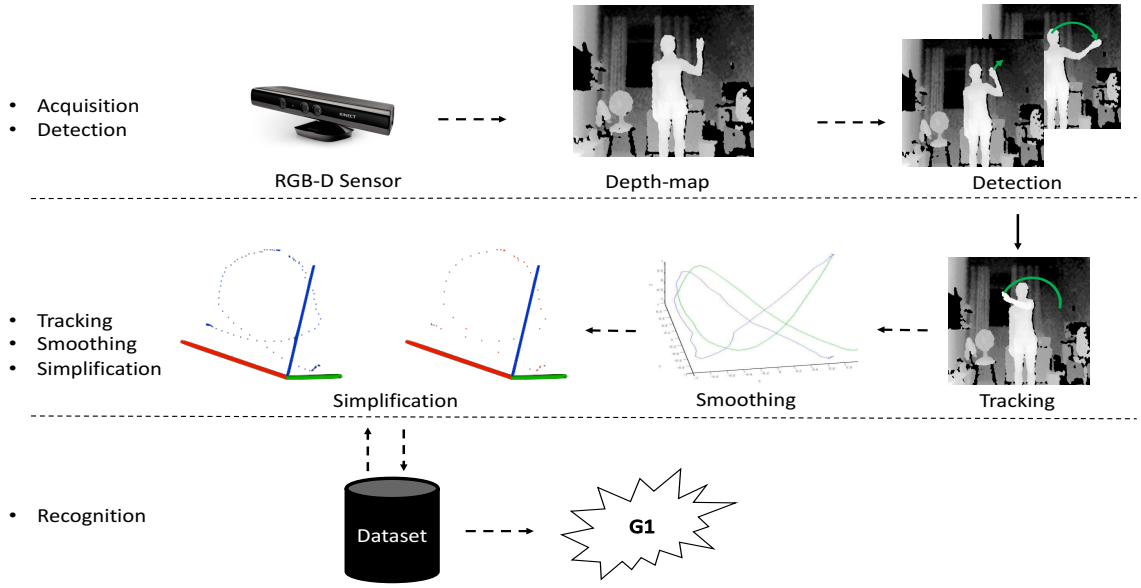

Figure 1: Overview of the proposed approach.

some disavantages like sensitiveness to lighting, color and shadow, limited acquisition by the distance, and the 3D information cannot be obtained directly [Han+13] [HK12].

Recent works uses the Dynamic Time Warping (DTW) [RK05] algorithm to recognize gestures. The DTW finds the cost of similarity through the alignment of two time series. [Iba+14] proposed a framework called EasyGR (Easy Gesture Recognition) that assists developers in the implementation of NUI applications, reducing the complexity through the encapsulation of the algorithms and management of the gesture data. [Bau+13] improved the recognition rates compared to the usual DTW [SC07] and Hidden Markov Model (HMM) [Rab89] using a probability-based DTW that updates the cost of the DTW according to a Gaussian model [Mat01]. However, both approaches were not evaluated taking into account the gestures obtained by people with different physical aspects.

[Wu+14] proposed a new method for view-invariant gesture recognition using a shape representation that is build from a set of euclidean distances between all trajectory points. The shape is smoothed using a ten-order B-Spline interpolation and the classification was performed using a Support Vector Machine (SVM) classifier [Mul+01]. Other approach [Bar+15] uses a wearable camera coupled to the user's head to recognize gestures performed by hands. This type of approach is can generate movement restriction, as it relies on wearable devices, batteries, cables, besides it can only recognize gestures that are in the field of view of the camera.

Our work provides an approach that use purely geometric information to recognize gestures. Different of some related works [Wu+14], we do not need train a model to recognize gestures. Different of [Bar+15], as we use a RGB-D sensor users do not need to use wearable devices to perform gestures. We introduce a step of simplification that can reduce the time consumption to recognize a gesture. This step allow the recognition in real-time.

3 PROPOSED SOLUTION

Our solution aims to use simplification and regularization techniques to speedup the recognition in real-time. We proposed an approach that is focused only in the use of geometric information.

The figure 1 shows an overview of our approach. First we use a RGB-D sensor (Microsoft Kinect) to capture the geometric information through the depth-map. Then, we use an algorithm [Sen13] to detect and track the 3D hands movement. The next step smooths and simplifies the gesture to remove noise and capture the key points. Finally, we use the DTW to classify the gesture.

3.1 Detection and Tracking

The first step aims to detect the hands using an RGB-D sensor. The RGB-D sensor used to acquire the depth information was the Kinect [CLV12]. We use a sample algorithm of the OpenNI [1] library to detect and track the movement of the hands. Detection begins from the execution of some of the basic gestures implemented by OpenNI, such as "bye bye". After the hand detection, the algorithm is able to track the movement of the hands, providing the central position $P_i(x,y,z)$ of the hand in each frame.

[1] http://www.openni.org/

We have developed an algorithm to automatically detect when a gesture starts and ends. This allow the continuous gesture recognition without human intervention using only a time interval between gestures. The gesture start consists of verifying if the sum S of the distances D_i between the previous n positions is greater than a threshold t.

$$S = \sum_{i=1}^{n-s} D_i \tag{1}$$

If the sum S is greater than the threshold t, then the hand is in motion, otherwise it is stopped. We choose $n = 25$ 25 empirically to detect the hand's movement.

3.2 Normalization

The next step normalizes the gesture since it can vary in scale and location. We use the same normalization proposed by [Iba+14] first calculating the centroid c_i of the gesture by dividing the sum of the points by the total number of points n of the gesture.

$$c_i = (\bar{x}, \bar{y}, \bar{z}) = \frac{\sum_{i=1}^{n}(x_i, y_i, z_i)}{n} \tag{2}$$

Then, the centroid is used to move all points to the origin with (3), that subtracts from each point of the gesture the respective coordinate of the centroid.

$$(x_i, y_i, z_i)' = (x_i - \bar{x}, y_i - \bar{y}, z_i - \bar{z}) \tag{3}$$

In the end, we scale the gesture in the interval -1 and 1. These processes ensures that the gesture is recognized regardless of the location and physical aspects of the user.

3.3 Smoothing

Once normalized the gesture, the next step smooths the raw gesture data to reduce noise by the depth sensor. The method used was the Laplacian [Tau95], which consist of recalculating all points using the mean of each point and its neighbors, according to (4). In our approach we used the 1-ring neighbourhood to smooth one time the gesture. Figure 2 shows the smoothing.

$$\bar{\mathbf{x}_\mathbf{i}} = \sum_{i=1}^{n-1}\left(\frac{x_i}{x_{i-1} + x_i + x_{i+1}}\right) \tag{4}$$

3.4 Simplification

After the normalization and smoothing of the gesture, the next step consists on its simplification. We use this step to provide a compact representation of the gesture, further improving the performance of the gesture recognition.

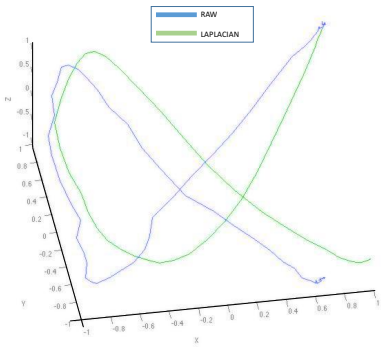

Figure 2: A raw gesture before and after smoothing using the Laplacian operator.

We use two algorithms to perform this task. The first approach simplifies the gesture using a curvature-based method and the second uses the Douglas-Peucker (DP) algorithm [DP73]. As we will show in the section 4, both of them keep the high recognition rate, while improving the algorithm's performance.

3.4.1 *Curvature*

The first simplification method of the gesture consists in checking whether the curvature of each segment is below a pre-defined threshold $t = 0.01$. In section 4.2 we explain how we choose this value.

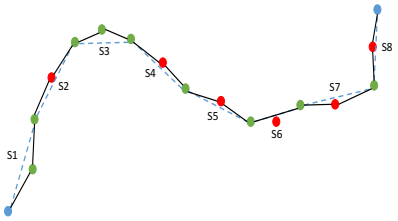

Figure 3: The endpoints in blue cannot be removed. The green points are points evaluated that cannot be removed. The red points can be removed because its curvature are below the threshold t.

As shown in Figure 3, the curvature-based method evaluates the curvature iteratively using segments defined by a point and its neighborhoods, e.g. $S_i = (p_{i-1}, p_i, p_{i+1})$. Then, for each segment S_i, if the curvature of S_i is below the threshold t, then the middle point p_i is removed. The point p_{i-1} of the next segment is the point p_{i+1} of the previous segment.

3.4.2 *Douglas-Peucker*

The Douglas-Peucker (DP) algorithm introduced by [DP73] is a polyline simplification. As shown in the

figure 4, the DP algorithm first use the endpoints $[A,B]$ to find and calculate the distance to the furthest point C. Then it uses the points $[A,C]$ and $[B,C]$ to calculate again the furthest points of $[A,C]$ and $[B,C]$, that is D and E. Finally it add the points $[C,D,E]$ if distance exceeded the tolerance t. This condition of similarity is based on the maximum distance measured between the original and the simplified curve. The original endpoints always are inserted in the simplified curve.

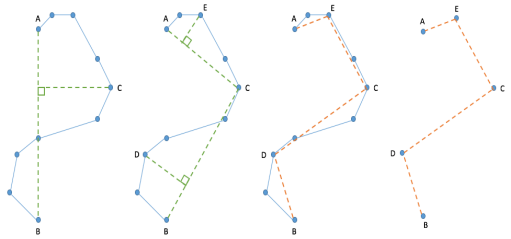

Figure 4: Steps of the DP algorithm to simplify a curve with 10 points.

We choosed this algorithm because it can reduce the number of points of the gesture while retaining its shape. Furthermore, the DP algorithm is faster then others algorithms like Bend Simplify [VH99].

3.5 Recognition

The last step use Dynamic Time Warping (DTW) [RK05] as classification method to recognize gestures. We use the nearest neighborhood algorithm with DTW to find the closest gesture according with the cost distance provided by it.

3.5.1 Dynamic Time Warping - DTW

The DTW algorithm finds the cost of similarity through the alignment of two time series, which in our case are the gestures. The basic idea is to construct an array of distances between the two trajectories and find the minimum distance between each pair of points. The result of the comparison is the sum s of the smallest distances found. The lower the value of s, the higher the degree of similarity between the two trajectories.

One of the main advantages of using the DTW is that it allows to compare two trajectories, even if they have different lengths [RK05]. This property of the DTW is important, since the gestures can be done with different speeds, so that the sampling rate is not always the same.

4 EXPERIMENTAL RESULTS

This section describes in detail the experimental setup and results. First we did a cross-validation to evaluate different parameters and obtain the optimal values to use in the Curvature, DP and DTW algorithms. Then we use the parameters found to evaluate the recognition rate and performance for each class of gesture applying these algorithms. Also, we evaluate the recognition and performance using as template the median gesture from each class.

The experimental evaluation was performed in a Macbook Pro (13-inch, Late 2011), Processor 2.4GHz Intel Core i5, Memory 4GB 1333 MHz DDR3, Intel HD Graphics 3000 384 MB, macOS Sierra version 10.12.3.

4.1 Dataset

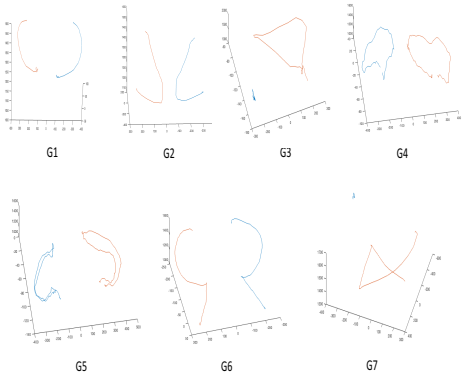

Figure 5: Set of 7 gestures of our dataset.

We tried use the MSRC-12 [Fot+12] dataset, but it has gestures written continuously in a single file and we did not have the executions separator to do it. ChaLearn [Guy+14] has only RGB and depth videos. Therefore, we create our own dataset.

Our dataset contains 1099 gestures, collected from 7 individuals performing 7 gestures, with different physical aspects and positions. In our dataset, we have 5 gestures performed using two hands and 2 gestures performed by one hand. The gestures were recorded using the Kinect XBox 360 sensor at a sample rate of 30Hz. We recorded both RGB, depth and 3D motion of hands, but we only used the motion of hands. Each motion contains a set of 3D positions of both hands. Figure 5 shows our 7 gestures, 6 of them are the same defined in dataset [Fot+12].

After create the dataset, we splits our dataset with 1099 gestures to perform the evaluation using 70% for test and 30% as template matching for each class of gesture. The next step was to generate 7 median gestures from our dataset to use as template and evaluate the recognition rate using all 1099 as tests.

To generate median gestures, we first apply for each gesture class a method to normalize the distance between points according with the desired Euclidean dis-

tance $k = 0.1$. The method calculate the Euclidean distance d_i of each segment s_i and remove the point p_{i+1} if $d_i < k$, otherwise we apply a linear interpolation in the segment s_i until $d_i < k$. Then, we equalize the number of points removing or adding points according with the average point of each gesture class. Finally, we calculate a simple mean for each gesture class g_i with (5), where we divide the sum of x_i, y_i and z_i by the total number of gestures n of each class.

$$g_i = \frac{\sum_{i=1}^{n}(x_i, y_i, z_i)}{n} \quad (5)$$

4.2 Cross-validation

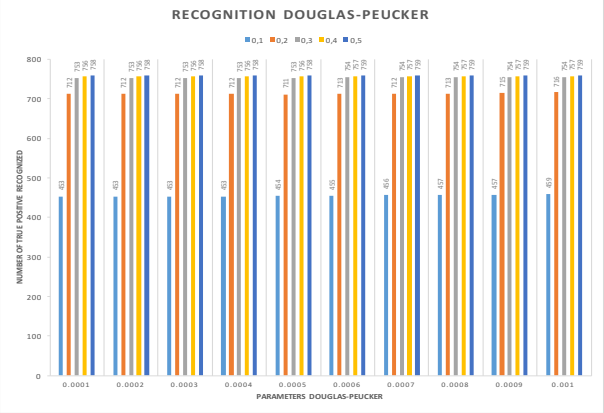

Figure 6: Cross-validation applied for DP with DTW using the parameters in Table 1.

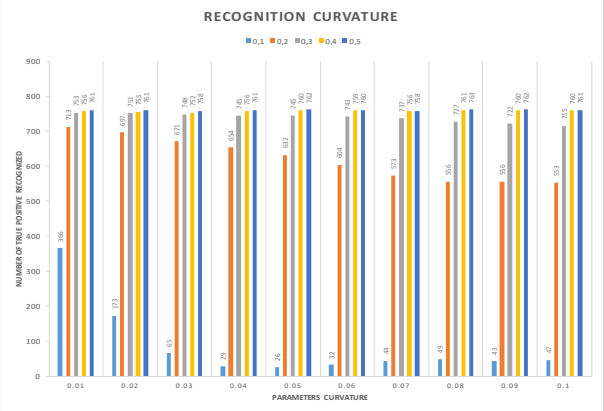

Figure 7: Cross-validation applied for Curvature with DTW using the parameter shown in Table 1.

Algorithm	Parameter domain	Final
Curvature	$0.01 <= t <= 0.1$	0.01
DP	$0.0001 <= t <= 0.001$	0.001
DTW	$0.1 <= d <= 0.5$	0.5

Table 1: Table with the cross-validation parameter domains for each algorithm and the chosen ones.

To perform cross-validation, we use all the gestures in our dataset, totaling 1099 gestures of different classes. We selected 30% of each class for tests, being the same applied for the different combinations of parameters.

After the data preparation phase, we selected a parameter domain for each method described in Table 1. The domains of the parameters were defined according to the normalization of the gesture in the interval of -1 to 1 and in some values tested manually to find the minimum and maximum thresholds of each algorithm. The best threshold criteria was the recognition rate resulting from each combination. The column Final in the Table 1 shows the selected final parameters for each method.

Figures 6 and 7 show the cross-validation result applied in the DP and Curvature algorithms for each DTW parameter shown in Table 1.

4.3 Recognition rate

We have created an algorithm to automate the execution of the tests. Initially the algorithm loads and divides the data into test and template according with section 4.1. Then, the pre-processing is applied for each test iteration before the classification using the DTW algorithm. We save in a file all the parameters used, including the time needed to process and classify the gesture.

As we can see in the Table 2, the results show a recognition rate above 90% with 83.75% on average, even applying a simplification step. The Laplacian with DP provides an improvement of 1.73% compared to the recognition with raw data simplification. Compared to the DTW results of [Iba+14], our approach with simplification showed an improvement of 2% in the recognition rate. We noticed that some classifications of the $G1$ gesture always made matching with the $G6$ gesture, however the opposite did not occur. The same occur to the one hand gestures $G3$ and $G7$, where the recognition rate for $G3$ was 100% while the $G7$ had an average of 94.5%.

Table 3 show the results using median gestures as template. We get a reduction in the recognition rate for the gesture $G2$, where we identified matching with $G1$ that are similar with it. Using the DP algorithm we get an improvement of $47,62\%$ over the raw data. In general the recognition rate was above 90%.

We also identified during the cross-validation process that the variation of the parameter for simplification using curvature did not affect the recognition rate for an evaluation of ten-order threshold. On the other hand the parameter for the DP reduced the recognition rate for larger values, being not robust to variation.

4.4 Performance

Figures 8 and 9 shows the performance results using simplification and regularization methods. In figure 8, compared to the classification with the raw data, the time needed to recognize the gesture was reduced more

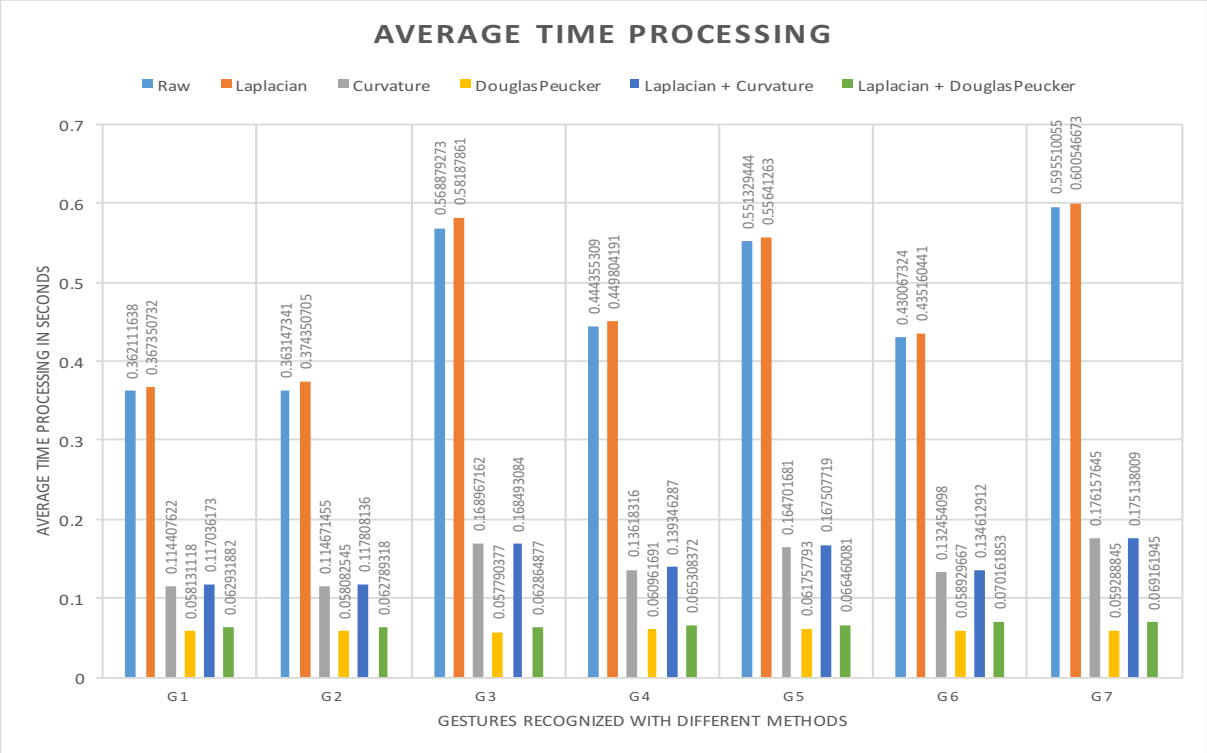

Figure 8: Average time processing for recognize each class of gesture. The process includes both pre-processing and classification time.

	Raw	Laplacian	Curvature	DP	Laplacian + Curvature	Laplacian + DP
G1	90.55	90.55	90.55	90.55	90.55	90.55
G2	100	100	100	100	100	100
G3	100	100	100	100	100	100
G4	98.93	98.93	98.93	98.93	98.93	98.93
G5	100	100	100	100	100	100
G6	100	100	100	100	100	100
G7	94.55	91.81	94.54	95.54	93.63	97.27

Table 2: Recognition rate using different combinations of algorithms for simplification and smoothing for each gesture class.

than a half using a simplification step. Also, figure 9 shows that the average time processing to recognize with median gestures as templates was lower then 2 milliseconds. The Curvature was better then DP for median gestures.

The gestures *G*3, *G*5 and *G*7 had a longer processing time because they are more complex, as can be seen in figure 5. The difference between the raw and Laplacian gestures was very subtle, with a slight increase in the time of recognition with the Laplacian, since it only smooths without simplifying the gesture.

4.5 Discussions

As shown in this section, the recognition rate does not changed significantly when we apply a step to simplify the gestures. However, the performance was reduced more then a half when we apply the simplification. Furthermore, the median gestures reduced the average time processing to 2 milliseconds. With median gestures we allow recognize without compare all samples of the dataset.

The DP algorithm shown better results in performance, but lost for curvature in the recognition rate. We note that the curvature-based method is more robust in simplification in the sense of maintaining the key points that describe the shape of the gesture. This explains why the curvature algorithm obtained better recognition rate results than DP and because DP processing time was better. The DP tends to remove more points in the simplification.

As we conclude, the filter is interesting because can improve the performance without affect the recognition using specifically the DTW.

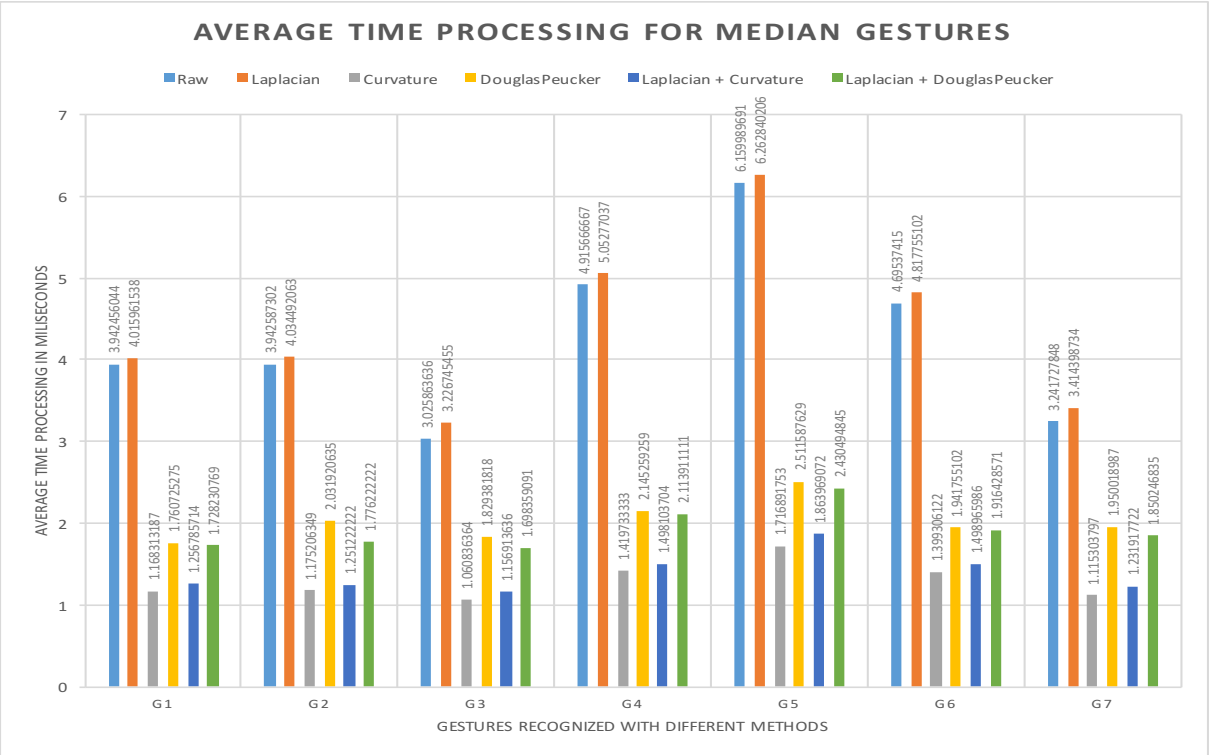

Figure 9: Average time processing for recognize each class using as template the median gesture. The process includes both pre-processing and classification time.

	Raw	Laplacian	Curvature	DP	Laplacian + Curvature	Laplacian + DP
G1	93.40	93.40	93.40	93.95	93.40	93.40
G2	44.44	44.44	47.61	92.06	52.38	79.36
G3	100	100	100	100	100	100
G4	99.25	99.25	99.25	99.25	99.25	99.25
G5	91.23	92.78	93.29	93.29	93.29	93.29
G6	100	100	100	98.63	100	97.27
G7	100	100	100	100	100	100

Table 3: Recognition rate using as template median gestures and different combinations of algorithms for simplification and smoothing.

5 CONCLUSION

In this paper, we propose an approach to gesture recognition based on geometric data and simplification of its representation. We analyzed two simplification methods based on curvature and DP algorithm. In the first, we obtained a recognition rate of 97.7% on average, while for the DP algorithm, we have obtained 98.1%. Using median gestures, we obtained a recognition rate above 90% with exception of the gesture $G2$. Both simplification methods evaluated reduced the recognition time in more than 2 times, being the DP more efficient for the first case, while for median gestures the Curvature was better then DP.

Simplification plays an important role in gesture recognition systems that have large robust datasets. The classification in such systems can not be robust in real-time without a pre-processing step because we noted in our results that performance depends of the number of gestures and points. This makes sense, because we must compare all gestures template to ensure the best match. One of the more important advantages of the simplification is the recognition time reduction.

As future work, we want to create more sophisticated gesture descriptor and use it with a tree decision to avoid full comparison of gestures in the dataset. We also will evaluate the simplification in supervised approaches with HMM to check if the recognition keeps robust after the simplification. As future work, we will also use standard datasets to evaluate our approach. Finally, we will try to recognize gestures continuously, without human intervention and without the need for time intervals between the beginning and end between the gestures.

6 REFERENCES

[DP73] David H Douglas and Thomas K Peucker. "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature". In: *Cartographica: The International Journal for Geographic Information and Geovisualization* 10.2 (1973), pp. 112–122.

[Rab89] Lawrence R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[Tau95] Gabriel Taubin. "A signal processing approach to fair surface design". In: *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. 1995, pp. 351–358.

[VH99] Mahes Visvalingam and Simon Herbert. "A computer science perspective on the bendsimplification algorithm". In: *Cartography and Geographic Information Science* 26.4 (1999), pp. 253–270.

[Mat01] Mark W Matsen. "The standard Gaussian model for block copolymer melts". In: *Journal of Physics: Condensed Matter* 14.2 (2001), R21.

[Mul+01] K-R Muller et al. "An introduction to kernel-based learning algorithms". In: *IEEE transactions on neural networks* 12.2 (2001), pp. 181–201.

[RK05] Chotirat Ann Ratanamahatana and Eamonn Keogh. "Three myths about dynamic time warping data mining". In: *Proceedings of the 2005 SIAM International Conference on Data Mining*. SIAM. 2005, pp. 506–510.

[MA07] Sushmita Mitra and Tinku Acharya. "Gesture recognition: A survey". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37.3 (2007), pp. 311–324.

[SC07] Stan Salvador and Philip Chan. "Toward accurate dynamic time warping in linear time and space". In: *Intelligent Data Analysis* 11.5 (2007), pp. 561–580.

[CLV12] Leandro Cruz, Djalma Lucio, and Luiz Velho. "Kinect and rgbd images: Challenges and applications". In: *Graphics, Patterns and Images Tutorials (SIBGRAPI-T), 2012 25th SIBGRAPI Conference on*. IEEE. 2012, pp. 36–49.

[Fot+12] Simon Fothergill et al. "Instructing people for training gestural interactive systems". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2012, pp. 1737–1746.

[HK12] Haitham Sabah Hasan and S. Abdul Kareem. "Human Computer Interaction for Vision Based Hand Gesture Recognition: A Survey". In: *2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT)* (Nov. 2012), pp. 55–60.

[Bau+13] Miguel Angel Bautista et al. "Probability-based dynamic time warping for gesture recognition on RGB-D data". In: *Advances in Depth Image Analysis and Applications*. Springer, 2013, pp. 126–135.

[Che+13] Lingchen Chen et al. "A survey on hand gesture recognition". In: *Computer Sciences and Applications (CSA), 2013 International Conference on*. IEEE. 2013, pp. 313–316.

[Han+13] Jungong Han et al. "Enhanced computer vision with microsoft kinect sensor: A review". In: *IEEE transactions on cybernetics* 43.5 (2013), pp. 1318–1334.

[Pal+13] Jose Manuel Palacios et al. "Human-computer interaction based on hand gestures using RGB-D sensors." In: *Sensors (Basel, Switzerland)* 13.9 (Jan. 2013), pp. 11842–60.

[Sen13] Prime Sense. "NITE Algorithms". In: *PrimeSense NITE Algorithms 1.5*. Feb. 2013, pp. 1–3.

[Guy+14] Isabelle Guyon et al. "The ChaLearn gesture dataset (CGD 2011)". In: *Machine Vision and Applications* 25.8 (2014), pp. 1929–1951.

[Iba+14] Rodrigo Ibanez et al. "Easy gesture recognition for Kinect". In: *Advances in Engineering Software* 76 (2014), pp. 171–180.

[Wu+14] Xingyu Wu et al. "View-invariant gesture recognition using nonparametric shape descriptor". In: *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE. 2014, pp. 544–549.

[Bar+15] Lorenzo Baraldi et al. "Gesture Recognition using Wearable Vision Sensors to Enhance Visitor's Museum Experiences". In: *IEEE Sensors Journal* 15.5 (2015), pp. 2705–2714.

[RA15] Siddharth S. Rautaray and Anupam Agrawal. "Vision based hand gesture recognition for human computer interaction: a survey". In: *Artificial Intelligence Review* 43.1 (2015), pp. 1–54.