A Machine Learning Approach for Gesture Recognition with a Lensless Smart Sensor System

Niccolò Normani[†,*] Andrea Urru[*] Lizy Abraham[*] Michael Walsh[*] Salvatore Tedesco[*] Angelo Cenedese[†]
Gian Antonio Susto[†] Brendan O'Flynn[*]
[*]WSN Group, Micro & Nano Systems - Tyndall National Institute, University College Cork, Cork, Ireland
[†]University of Padova, Department of Information Engineering (DEI), Padova, Italy

***Abstract*— Hand motion tracking traditionally requires highly complex and expensive systems in terms of energy and computational demands. A low-power, low-cost system could lead to a revolution in this field as it would not require complex hardware while representing an infrastructure-less ultra-miniature ($\sim 100\mu m$ - [1]) solution. The present paper exploits the Multiple Point Tracking algorithm developed at the Tyndall National Institute as the basic algorithm to perform a series of gesture recognition tasks. The hardware relies upon the combination of a stereoscopic vision of two novel Lensless Smart Sensors (LSS) combined with IR filters and five hand-held LEDs to track. Tracking common gestures generates a six-gestures dataset, which is then employed to train three Machine Learning models: k-Nearest Neighbors, Support Vector Machine and Random Forest. An offline analysis highlights how different LEDs' positions on the hand affect the classification accuracy. The comparison shows how the Random Forest outperforms the other two models with a classification accuracy of 90-91 %.**

***Keywords* - Lensless Smart Sensor, Machine Learning, Random Forest, Gesture Recognition**

I. Introduction

Gesture recognition represents an important topic of research used in a wide range of applications: Human Computer Interface, Sign languages, Entertainment, Augmented/Virtual Reality and many others. In the field of sign-language contributions to gesture recognition could be found in [2], while in speech recognition in [3]; a detailed survey of some recent works on hand gesture recognition using 3D depth sensors is given in [4]. Different approaches have been employed to provide a solution starting from the well-known camera-based system. In [5] it is demonstrated how the computational complexity is quite high for conventional vision-based hand detection and tracking. Many of those works share the same Machine Learning (ML) approach to perform a classification task and inspire the present paper. The novelties here illustrated rely upon the combination of two infrared (IR) filters and 5 LEDs being tracked. Two separate light tracking systems based on Lensless Smart Sensors (LSSs) [1] exploit the principle of stereo-vision to range and track multiple LEDs located at different key points on the hand, simultaneously moving within the Field-of-View (FoV) [6]. A gesture can thus be described by a sequence of tridimensional positions of the tracked LEDs. A gesture recognition algorithm relying on different ML techniques was designed and an analysis is conducted to highlight how different positions of the LEDs affect the classification accuracy. Different State-of-the-Art techniques are evaluated in terms of their performance through Misclassification Rate (MCR): k-Nearest Neighbors (kNN), Support Vector Machine (SVM), Random Forest (RF) [7], [8]. The manuscript is organized as follows: Section II gives a brief presentation of the Hardware Setup, while the Multiple Point Tracking algorithm to derive the 3D positions of 5 LEDs is explained in Section III. The main contribution of the paper is provided in Section IV and V: first, the Frame-based Approach is introduced [7] and applied to the framework; the protocol applied to generate the dataset needed in the training phase is briefly summarised; finally the trained algorithms are employed to evaluate the classification accuracy associated to each gesture. Conclusions and future works are described in Section VI.

II. Hardware Setup

The hardware is based on the novel LSS. As discussed in [1], [9], currently smallest traditional focusing cameras are roughly 1 mm in diameter, with size limited by need for lenses. By shifting some of the burden of focusing to computation, new classes of lensless imagers much smaller than traditional focusing systems could be designed. Those ultra-miniature computational image sensors employ phase gratings with optimal optical properties and are integrated with CMOS photodetector matrices (Size $\sim 100\mu m$; Cost $\sim$ a few Euro cents; depth of field $\sim 1mm \rightarrow \infty$). In Fig.1 each sensor is embedded in a 3D-printed casing and covered with an IR filter (on the front) to properly match the wavelengths of the 5 LEDs -$890nm$- which are equipped on the hand through a 3D-printed hand mounted fixture. The position of each LED in 3D is obtained through the Single Point

E-mails: brendan.oflynn@tyndall.ie, michael.walsh@tyndall.ie, salvatore.tedesco@tyndall.ie. This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) and is co-funded under the European Regional Development Fund under Grant Number 13/RC/2077. The authors would like also to thank Evan Erickson, Patrick Gill and Mark Kellam from Rambus Inc.

Tracking algorithm: the distance along Z is calculated using trigonometry together with the Snell's Law while the lateral and vertical displacements (along X and Y) w.r.t. the reference frame can be derived according to the striking' position of the LED w.r.t. the LSSs' focal points [6].

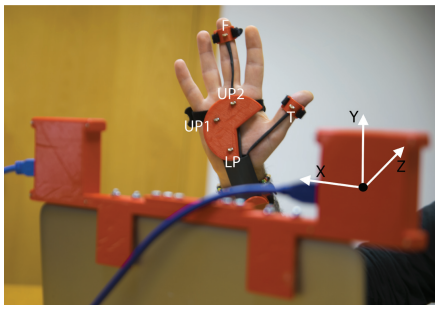


Fig. 1: Hardware Setup - Labels: - F (Middle Finger) - LP (Lower Palm) - T (Thumb) - UP1 (Upper Palm 1) - UP2 (Upper Palm 2)

III. Multi-Point Tracking

The Multi-Point Tracking algorithm can be structured into two main phases: calibration and tracking.

A. Calibration Phase

The calibration phase is assumed that the hand is kept still in front of the sensors with the palm and fingers open and the middle finger pointing upwards. The calibration is performed once at system start to capture the LEDs relative distances from each other and use them as a reference in the Tracking Phase. The detection to identify the points within the field-of-view is performed through hard-thresholding by looking at the LEDs' intensities. Exploiting the positions of the LEDs on the stationary hand the labels (see Fig. 1) are assumed to be known. The 2D coordinates in the image domain together with the assigned labels are singularly processed according to the Single Point Tracking algorithm [6]. As shown in the Hand Setup, it is noticeable how the points lying on the palm $LP, UP1$ and $UP2$ are characterized by fixed relative distances thus providing a useful information to be used in the Tracking Phase. The matrix of relative distances of the palm coordinates is computed and stored.

B. Tracking Phase

The tracking algorithm explained in this subsection refers to using a iteration. The tracking phase can be performed continuously by iterating the algorithm every time a new frame is captured. The detection is performed using the same reference threshold used during the calibration phase. If the number of detected points is asymmetric among the two LSSs or is less than 3, the useful information to provide an estimate of the current coordinates is given by the previous frames by applying a polynomial function ($deg = 2$) that fits the last ten progressive time stamps and the corresponding stored coordinates. In case of successful detection the 2D coordinates in the image domain are individually processed according to the Single Point Tracking algorithm [6]. Combinations of three tracked LEDs are generated without repetitions: the sought combination is represented by the one that shows the closest matrix of relative distances to the matrix stored during the calibration phase. By permuting the three LEDs composing the identified combination, the labels' assignment is performed by establishing the permutation which provides the minimum residuals w.r.t. calibration matrix. The fingers are labeled according to their reciprocal positions w.r.t. the palm's plane. A detailed explanation of the multi-point algorithm can be found in [10].

IV. Formulation

A. Dataset

We here consider a multi-classification problem where each class is identified with a gesture. The vocabulary of gestures is reported in Table I.

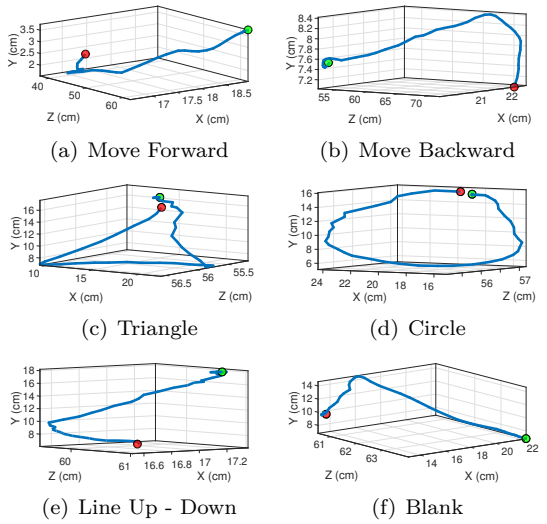
Gesture Label	Description
1 - Forward	Forward movement along Z
2 - Backward	Backward movement along Z
3 - Triangle	X-Y: Basis parallel to X
4 - Circle	X-Y plane
5 - Line Up - Down	X-Y plane
6 - Blank	None of the previous gestures

TABLE I: Gestures Vocabulary

The last class is added to model the "non - gesture" class that embeds all of the movements not included in the first 5 classes. The gestures are performed at different distances and positions w.r.t. the sensors while keeping the movement within the space between the two sensors (Fig. 1). Established through experimental validation, a viable capture time needed to describe a gesture consists of 50 captures at $20fps$, which corresponds to a window of 2.5 seconds gesture. Faster movements are contained in such window, while slow ones could be either classified correctly or uncompleted, thus the blank class. The latter includes the hand rotation as a non-gesture as well. The LSSs together with the 3D-printed casings were placed on a selection of laptops during the data collection to take into account also the variability among different displays' heights and to make the recognition more robust for general usage. The user is also required to perform each of the combinations (clockwise/counterclockwise/different vertex as starting point) making the switch to a left hand-based system immediate. In Fig. 2 an example for each class is provided. In the data collection 10 individuals were involved to derive a total of 600 gestures. The sample size is detoned as $\mathcal{D} = \{(\mathbf{X_1}, Y_1), ..., (\mathbf{X_i}, Y_i), ..., (\mathbf{X_n}, Y_n)\}$, where $\mathbf{X_i}$ embeds the features extracted from the i^{th} gesture and Y_i is the associated class label.

B. Feature Extraction

The feature extraction here presented is based upon the concept of Frame-based Descriptor [7]. The latter represents a successful approach for extracting features



(a) Move Forward (b) Move Backward (c) Triangle (d) Circle (e) Line Up - Down (f) Blank

Fig. 2: Green: Starting Point - Red: Ending Point

from inertial measurements, i.e. 3D accelerometers. It is of great interest applying it to a dataset of different nature which consists of trajectories of the absolute positions indexed by time. Each trajectory of the positions $([t_x, t_y, t_z])$ is divided into $M+1$ contiguous portions of equal length ($M = 10$); every contiguous portion constitutes a frame [7]. For each frame, i.e. $j \in [1, ..., M]$ and each trajectory i.e. $t \in [t_x, t_y, t_z]$, the following quantities are computed:

- $\mu_{j,t}$: the continuous component of the Discrete Fourier Transform (DFT), representing the mean;
- $\varepsilon_{j,t}$: the energy computed without the contribution of the continuous component of the DFT;
- $\delta_{j,t}$: the entropy computed without the contribution of the continuous component of the DFT;
- $\sigma_{j,t}$: standard deviation;
- γ_j : axis correlation.

They are vectorized to represent the i^{th} gesture as follows: $\mathbf{X}_i = (\mu_{1,i}, ..., \mu_{m,i}, \varepsilon_{1,i}, ..., \varepsilon_{m,i}, \delta_{1,i}, ..., \delta_{1,m}, \sigma_{1,i}, ..., \sigma_{m,i}, \gamma_{1,i}, ..., \gamma_{m,i}) \in \mathbb{R}^{1\times p}$, where each variable embeds the features related to the three axis x, y, z. In [7] a hard thresholding is used to identify to segment the gesture by establishing how the energy and the variance of the inertial signals change. The reason why no gesture identification is performed here is related to the application. The defined scenario is a simple drawing application where the user is able to paint on the screen and control the tools through gestures by waving the hand in front of the LSSs sensors. Here, there is always energy associated to the trajectories making the previous approach unreliable. The idea is not to train the models with segmented gestures but with fixed windows containing the entire gestures.

C. Training

kNN classifies inputs according to the k closest training vectors (Metric distance: Euclidean norm) in the feature space by majority of vote. The SVMs are modelled through One Versus All approach. The kernel of choice is the RBF (Radial Basis Function): $K(x_i, x_j) = e^{-\gamma||x_i - x_j||^2}$. The RF is an ensemble of decision trees where each tree is trained by bootstrapping with replacement a dataset $\mathcal{D}_{boot} \in \mathbb{R}^{n\times p}$. Each node splitting is performed by choosing the i^{th} feature (among $\sqrt{p}$ features randomly selected at each split) that minimizes the Gini criterion [8]. The tuning of the neighborhood size k, the SVM hyperparameters C, γ and the number of trees n_{tree} (Table II) are all determined through 10-fold Stratified Cross-Validation as the preferred method to leave-one-out CV. It avoids skewness in the training-test datasets separation and performs better in the model selection [11]. The analysis are performed using MATLAB on a Desktop Machine (Intel i5, 3.5 Ghz, 16 GB RAM).

Dataset	k-NN - k	SVM - $[\gamma, C]$	RF - n_{tree}
C. of M.	15	[0.0146, 5936.6]	830
F	14	[0.0091, 138.9]	690
LP	7	[0.057, 3727.6]	280
T	12	[0.0146, 2330.0]	520
UP1	9	[0.0146, 33.9]	520
UP2	13	[0.0146, 15264]	270

TABLE II: Optimal Parameters - Labels: C. of M. (Center of Mass) - F (Middle Finger) - LP (Lower Palm) - T (Thumb) - UP1 (Upper Palm 1) - UP2 (Upper Palm 2)

V. Offline Analysis

The offline analysis is performed by repeating the 10-fold Cross Validation process employing the optimal parameters and averaging the 10 test sets' (created by the 10-fold CV) generalization performances. The comparison of classification accuracies pictured in Fig. 3 highlights how the RF is able to describe better the complexity of the data. Both kNN and SVM perform slightly better with

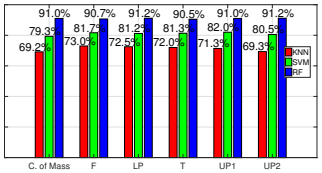


Fig. 3: Accuracies Comparison - Full Dataset

the F dataset than the other datasets (SVM - F classifies very closely to UP1). Such a result is coherent with the chosen hardware setup (Section 1): the sensors placed on top of the display follow its tilting. The F dataset might perform better since the middle finger is closer to the focal points of the LSSs. The RF performs better on the

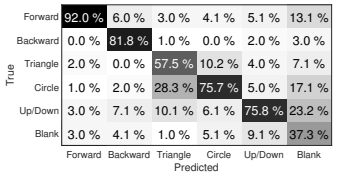


Fig. 4: Confusion Matrix kNN - Full Dataset UP2

LP dataset than the F one, for which could be related

to the typologies of different boundaries built by the algorithms. We first analyze the worst case scenario: kNN with the UP2 dataset. Fig. 4 highlights how it struggles to model the Blank gesture. The result is coherent as the Blank gesture needs to capture all the non-gestures' diversities. The improvement of the accuracies related to the SVM and RF reported in Fig. 3 are also well explained by the associated confusion matrices (Fig.5). Even by increasing the complexity of the models the difficulty of modeling the last class is noticeable. The precision brought by the SVM is particularly increased with the Blank Gesture and the Line Up/Down gesture while the misclassification among the Circle and the Triangle gestures remains almost untouched. The latter is lowered by introducing the RF model where the classification accuracy related to the Triangle and the Circle has increased greatly. The imprecision of

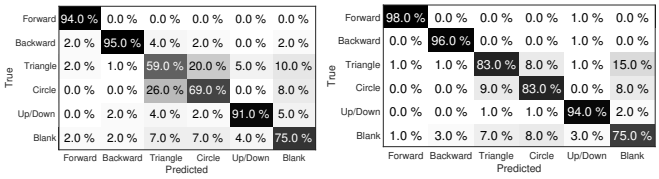


Fig. 5: SVM and RF confusion matrices - Full Dataset UP2

the Blank gesture which was more spread out among the classes in Fig. 4 is here concentrated around the geometrical figures: uncompleted triangles or diagonal lines and swipes could often be interpreted as triangles. This suggests that the Frame-based Approach may not represent a good choice when dealing with geometrical gestures described by absolute positions. In Fig. 6 it is

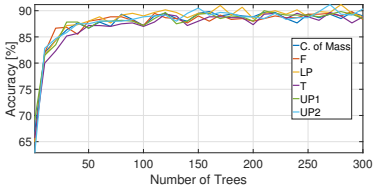


Fig. 6: Random Forest - Accuracy

illustrated that increasing the number of trees doesn't affect the accuracies, thus guaranteeing a faster prediction without sacrificing the precision. Such a result could become very useful when dealing with the computational burden in an online gesture recognition. To evaluate the importance of the extracted features, the RF is trained with UP2 dataset and the optimal n_{tree}: the Out-of-Bag samples corresponding to each built tree are considered.

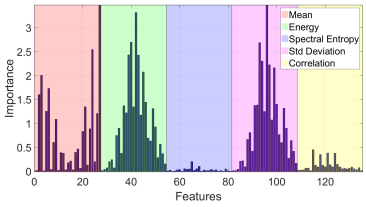


Fig. 7: Features Importance

In Fig. 7 the percentage increase in MCR is pictured by considering the classification accuracy if the OOB samples are run down the related trees with the m^{th} variable randomly permuted ($m \in [1, ..., p]$) [8]. It is noticeable how the mean, the energy and the standard deviation are the most important factors which affect influence the prediction: the first shows the highest decrease in accuracy on the z axis which is significant mostly in the classification of the forward and the backward movements. The importance is primarily focused on the nucleus of the gesture in the energy and the standard deviation features where the hand motion is characterized by most of the variability.

VI. Conclusions

The paper presented the novelty associated with the RAMBUS LSSs and their potentialities in hand-tracking applications. In particular, it was shown that the Multi-Point algorithm allows the identification and tracking of 5-LEDs located at key points on the hand and the application of the Frame-based Descriptor to the current framework. The offline analysis pointed out the ability of the RF to generalize better than the SVMs and kNN algorithms. The confusion matrices show the behavior of the classification as a function of the gestures. Even if straight movement are modeled well, the misclassification among the Circle, the Triangle and the Blank is still significant. The discussion pointed out the difficulty to model the non-gesture class as the category which contains the most variability in terms of possible hand motions. The RF represents a promising choice as a starting point to furtherly improve the accuracy and the robustness of the gesture recognition. A different feature extraction approach and the real-time implementation will be the main objectives of future works.

References

[1] P. Gill *et al.*, "Optical, Mathematical and, Computational Foundations of Lensless, Ultra-Miniature Diffractive Imagers and Sensors," in *International Journal on Advances in Systems and Meaurements*, Rambus Labs, Sunnyvale, 2014.

[2] S. Escalera *et al.*, "Challenges in Multimodal Gesture Recognition," *Journal of Machine Learning Research*, vol. 17, no. 72, pp. 1–54, 2016.

[3] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," in *Proceedings of the IEEE*, 1989, pp. 257–286.

[4] S. Mitra and T. Acharya, "Gesture Recognition: A Survey," *IEEE Transactions on Systems, Man and Cybernetics - Part C*, vol. 37, no. 3, pp. 311–324, 2007.

[5] X. Suau *et al.*, "Real-Time Head and Hand Tracking based on 2.5D Data," in *2011 IEEE International Conference on Multimedia and Expo*, July 2011, pp. 1–6.

[6] L. Abraham *et al.*, "3D Ranging and Tracking Using Lensless Smart Sensors," in *Smart Integration Systems*, Cork, 2017.

[7] G. Belgioioso *et al.*, "A Machine Learning based Approach For Gesture Recognition from Inertial Measurements," in *53rd IEEE Conf on Decision and Control*, Dec 2014, pp. 4899–4904.

[8] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001.

[9] P. Gill *et al.*, "Lensless Ultra-Miniature Computational Sensors and Imagers," in *SensorComm*, Barcelona, Spain, 2013.

[10] L. Abraham *et al.*, "Hand Tracking and Gesture Recognition using Lensless Smart Sensors," in *IEEE Sensors*, pp. 1–8, submitted.

[11] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," vol. 14, 03 2001.