Uniqueness in Air-Drawing of Symbols

Georgios Bouzias
Graduation Thesis, June 2018
Media Technology MSc program, Leiden University
Supervisors: Edwin van der Heide & Peter van der Putten
g.bouzias@umail.leidenuniv.nl

***Abstract*— For this research we did build and evaluate classification models trained with air-drawings of a simple symbol with the purpose of identity verification. For the task, we first collected air-drawing data from 16 participants with a software application that utilizes the depth camera of a Kinect v2. With the data, we trained two one-class classifiers for each of our users: an one-class SVM and an Isolation Forest. We then evaluated the models with AUC score across different groups of features. For the feature construction process, we made multiple features with a process tailored to the symbol of our focus. As far as the results are concerned, for their best performing features group, the one-class SVM and IF had AUC scores of 0.988 and 0.977 respectively. These AUC scores have been computed by the mean ROC curve of the 16 participants. The corresponding EERs were 4.05% for one-class SVM and and 7.2% for IF. Conclusively, we would say that given the performance, a verification system as discussed in this study, could possibly be a reality in cases that do not have very high security requirements. More research and advancements are needed to be considered as reliable in more sensitive cases such as a scenario of locking or unlocking the house door.**

I. Introduction

The purpose of this study is to research whether the air-drawing of a simple symbol can be an adequate behavioral biometric that could be used for user verification. To address that, in this project we focus on a specific symbol (that of Figure 1) and a motivation scenario, on top of which we collect a dataset of air-drawings, that we use in order to evaluate various classification models.

Our goal then is to test the performance of the system among different users as well as the effectiveness of the different classification algorithms that we use. In addition to that, we also aim to experiment with different features and find out the ones that perform best. The way we conduct the research to answer those things, is by following a typical pipeline such systems usually do [1], [2]. After collecting the data, the process is first to do some pre-processing with them, then to construct higher level features and finally to train several classification models.

As far as existing research goes which we mainly discuss in the following section, there does not seem to be any study that is completely in line with our case here. Instead, it seems that researchers have focused in online signature verification, though one can say that it is essentially the same thing but with a more complicated symbol. Also, in this signature verification branch, there are some cases where the signature is drawn in the air, similarly to our case, but it is safe to say that 2D signatures are much more well researched.

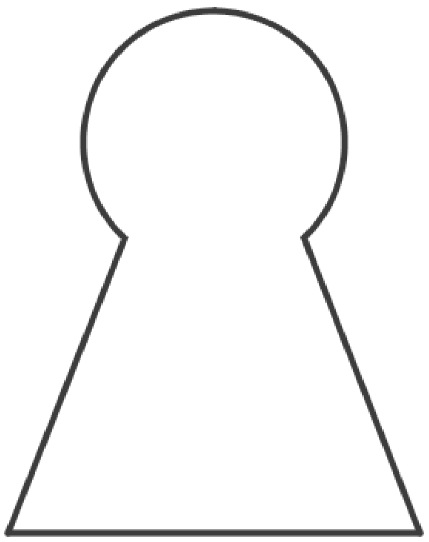


Fig. 1: A lock symbol that could potentially be air-drawn to open a door. This is the same symbol that was shown to the users during data collection as the "ideal" lock.

Next in the introduction section we discuss a motivation scenario, our reasoning behind methodological choices in data collection and classification process, some concerns regarding behavioral biometrics, and we also outline our research questions according to the purpose of our research. In section II we discuss most of the previous work done in air-drawing verification that we were able to find, as well as some leading studies in signature verification and also verification with Behavioral Biometrics in the broader sense.

Following that, in section III we outline how we conducted data collection that has to do with the lab environment and the application we developed for that purpose. In section IV we discuss the methods that we used to build the classification system, and in section V we present these classification results. Furthermore, in section VI we discuss the outcomes of this study, avenues for feature work and limitations of this project. While discussing the outcomes, we also compare our work with leading behavioural biometric approaches that we discuss in section II. Finally, in section VII we summarize our conclusions.

A. Motivation Scenario

Theoretically, verification by someone's air-drawings should be possible given the uniqueness of one's behavior. However it is important to discuss why this could be useful and motivate for a context in which it could be studied. The latter is needed mainly because the air-drawings could potentially be of many symbols and produced with different types of movement.

In the case of this research, motivation comes from the potential of communicating messages to items, with symbols drawn in the air. The core concept is that the items can probably know a list of affordances and that if they receive a drawing of a symbol, then they can perform a certain action. These affordances can be simple actions like 'open' if it is for a door or a lamp, or 'start' for a washing machine etc.

In that context, a potential application could have items (e.g. lamps or doors) sensing the environment with a depth camera in order to capture drawings in space. That is not very applicable though, given that there is not real incentive for simple items like a lamp, to become that much more complicated and also cameras everywhere might not omit a very nice feeling. Another scenario could be that a user makes air-drawings with his finger while having an accelerometer mounted to his nail. In this case a system in the user's hand should compile the drawing and send it to the item. A similar thing could be achieved if the user is making the drawing with his phone.

A system like the ones described could potentially be owned by a user. In the first case, cameras could be set up in somewhere like doors. In the second case an accelerometer could be mounted on the nail and stream acceleration data to a device with computing power like the user's phone or watch. Our research refers to a system like that, that is owned by a user and tries to explore the possibility of verifying the user's identity by air-drawings of a given symbol.

This study is elaborated based on this concept. However, in order to approach it in a meaningful manner we had to focus on specifications in three areas:

- The application to capture the drawings,
- a communication scenario and,
- the interaction with the application.

The application we focused on was the one that involves depth cameras, the communication scenario was the opening of a door with a lock symbol like the one of Figure 1, and the interaction was simply set as drawing the lock with the finger while standing.

Application: Both of the applications that are being mentioned earlier in the section could potentially be suitable to use in this research. In this research we will go with the scenario that uses depth cameras to capture the air-drawings in space. We recognize, however, that probably a system with accelerometers could have bigger potential, given that it does not include a camera to be integrated in an item for it to be available for communication.

There are several reasons that we are taking this direction that we will discuss here. First of all, a major aspect is the freedom of movement that the depth camera allows. This enables us to study the behavior in a natural and intuitive interaction which makes sense. If we were to use an Inertial Measurement Unit (IMU) and a microprocessor, it would be much harder to achieve the same level of freedom of movement with the same resources.

In addition to that, with positional data that the depth camera provides, there is potential to create features that have to do with our perception of the lock. That is not the case though with the acceleration space that is much more unintuitive. Also because of that, with the acceleration data it is harder to visualize the drawing for making sure that it is indeed a lock and for training the user with our system. As we discuss later on, the first is needed to make sure that we train our models only with locks, and the second mainly because the users need to understand what they are doing and familiarize themselves with the application.

Furthermore, for depth sensing there are several technologies and complementary tools that can produce very good accuracies that are relatively easy to use in the context of data collection sessions where a participant needs to interact with a software application. In that regard, another thing that encourages the use of such tools is that other studies in signature verification by air-drawing have produced good results with them [3], [4].

Communication Scenario and Selected Symbol: The communication scenario is the scenario with which we will study the air-drawing as behavioral biometric and how we will introduce the people we will collect drawings from, to the concept. Simply because the whole idea of verification is mostly needed for security, we think that the scenario of locking/unlocking a door with a lock symbol would be a very suitable one to go with. Other scenarios could be to start the washing machine, to open a lamp, perhaps by drawing a whirlwind or a lamp respectively. The exact lock symbol that we used and showed to the participants can be seen in Figure 1.

Regarding the lock symbol, besides the intuitive connection as a lock/unlock icon, there is another reason that also makes it appealing. That is because, given the fact that symbols differ in complexity, a simple enough symbol such as the lock will presumably be a tougher task for a model than a more complex one. A simpler symbol would also be preferred by the user for any task since it is quick and easy to draw and remember. In addition to that, given our focus to a particular symbol, we can construct some symbol specific features that we can try for classification.

Interaction: The interaction with the application is a key area a for the research since it is about the way the drawing is being produced. That is because we are looking for behavioral uniqueness in the drawing process as expressed in the drawing trace. However, if the process is different in each participant (or in each drawing), then the variability might be because of that. Difference in the drawing process we can see because of a choice the user made for drawing, or because of other more complex things like the environment or the person's mental state.

In this research we try to control some choices the participants make, in order to have traces that are produced by behaviors that are different only in the domain of unconscious natural movement. For the same reason we also try to control the environment to the extend we can, by keeping it in the same condition for all participants. However, there is little we can do for more intangible things that can also vary, like mental states. Also, despite any efforts to control things, since the main concern is to have lock drawings in the most natural way possible, we wanted the participants to draw completely free from the shoulder and onwards without any constraints.

In regards to the choices one makes when it comes to making a drawing, there might be many possibilities, with some choices having bigger impact potential than others. For example one choice could be whether to make the drawing with a finger or with a pen, and another choice could be a decision for where to lift the arm to start drawing, or with what speed. In this research we will try to control all the big choices that usually come after a cognitive process and we will treat all the others as part of a behavioral continuum that we are trying to study.

In that sense, while collecting the data we ask the users to do the drawing with the index finger of their right hand, to start drawing from the bottom left corner of the shape, and to stand straight a hundred and thirty centimeters in front of the Kinect camera. Also, only people with a dominant right hand were considered. Despite of that, it is important to keep in mind that users would conduct themselves differently in the real world where they could make all kinds of choices. This would probably add variability between different people but also between the drawings of the same person as we also discuss later on.

As far as making the drawing with the finger or some other tool (something like a stick) goes, despite the fact that we instructed the users to go with the finger, a wand or a stick could also be an appealing choice. The main advantage of using another tool instead of simply the finger, is that it probably adds more variability to the movement as there are some extra variables involved. That is due to the fact that in a sense the tool adds something like an extra joint. However, using the finger can probably be more convenient to the user in a real world scenario. Also, this is something that could not be demanded by the application and we had to make sure that this was the case on site during data collection.

Controlling other undesirable factors that are causes of the unwanted part of all possible behavioral variability like the environment or the mental states is also a target. In the case of environment, to the extend we can, we keep things the same for all participants as the data collection happens in the same lab room with the same setting for everyone. However, as we said earlier what happens in the psychological side of the participant we cannot control.

B. *Choice of Methods for Data Collection*

The problem of verifying the identity of a user by his air-drawings as has been described in the previous section can be approached in multiple ways. Our task for this research has two main parts. First we aim to collect some data of air-drawings, and then to create and evaluate a classification system that adequately does the verification for a claimed identity. For both of those things we have chosen a methodology that is explained more thoroughly in sections III and IV. In this section we will discuss our motivation behind our choices for some of our methods in regards to the first big part of the research, the data collection.

More specifically, the choices we had to make about building the dataset were:

- How to get the data (with what technology) and,
- how to make sure that the data is of the kind we want (lock drawings and not any other symbol).

As we discuss in this section, we extract positional data with an application that utilizes the depth camera of a Kinect v2, and we manually annotated which of the drawings were locks post-processing.

Getting the Data: Given that we have already discussed the context in which the data should be acquired, in this section we mostly aim to motivate on what technology we will be using to collect positional data and why. Microsoft Kinect v2 and Leap Motion Controller (LMC) devices are used in two separate and quite similar studies that examine verification by air-drawing [3], [4]. The LMC however allows for a very small interaction space compared to our needs, as it has a pyramidic field of view of 150 degrees with a range up to 1 meter [4].

The Kinect v2 on the other hand offers quite a large interaction space, up to 4.5 meters deep, a field of view of 70×60 degrees [5]. It makes use of a Time-of-Flight technique that gives it a tracking accuracy of 2mm in its prime tracking area [6]. Two more parameters that enhance the suitability of Kinect are its accessibility as an of-the-shelf device and the open source libraries that we could use with Processing 3.0 [7] to access its depth image [8].

The fact that we can use these open source libraries makes it even easier to use it in the context of a data collection session and by making an interactive application where some user can draw locks with his finger and have the trace captured. Apart from the ease of integrating the Kinect to our application, with the Processing language we can add visuals and cues that can dictate the flow of a session where a user can understand when it is time to make a drawing, or when to rest his hand etc.

Furthermore, it should be mentioned that besides the technologies mentioned here, there are also others that incorporate depth cameras. Different approaches with LIDAR, RADAR, sonar techniques or others with structured light, are all being used for tracking purposes [9]. However, to a different degree each one, seems to be lacking compared to the Kinect in the aforementioned areas.

In a final note, it would maybe be interesting to discuss the use of IMUs as a way to acquire positional data, given that it was one of the application scenarios that we have discussed earlier and much of the reason that made them less appealing was that the data we could collect with them would be in the rather unintuitive space of acceleration, with which we cannot make plots, create instinctual features, or train our participants.

However, we can also get positional data from acceleration if we integrate data from a 9DOF IMU twice to go from acceleration, to velocity and to position. The issue with this though, is that the small integration errors accumulate over time, which results to a well known problem of drift [10]. It also appears that techniques that filter out the drift are still a matter of scientific research [11], and that it is quite hard to get very good accuracies with of-the-shelf sensors.

Symbol Recognition: Another thing we need to discuss about the data collection process, is how can we make sure that the drawings we receive from the participants are indeed locks. In our broader description of the problem, the verification system receives a lock and, prior to finding out who did it, it needs to find what the symbol is about (it needs to be a lock in our case). That is because the verification process might require that knowledge (which it does in our case), apart from the item's need for knowing what the intended action should be.

However, for this research this concern is simply out of scope as it greatly increases the complexity. Indeed, adding a robust symbol recognition feature would be a hard task, but it is also necessary especially for a real world system that would probably support many symbols. Nevertheless, since we only want to study verification, and we only have one symbol and not too much data, it seems that this issue poses no real obstacle if we can manually check the symbols before the feature construction or the training process. Checking the symbols is very important not only for the algorithms to not be trained with bad data, but also for the way we are constructing features, that are based on the shape of the drawing.

Moreover, despite the fact that we are dealing with user verification it can be said that image recognition is not a very different kind of problem. The key distinction for a classification algorithm for that purpose, is that the labels instead of being different persons (drawing the same shape) would be different shapes (drawn by multiple individuals). For the latter, a great deal of different algorithms and techniques have been applied on the MNIST dataset [12], which is a database of images of handwritten digits. Another technique is applied by Vikram *et al.* in a relevant context, where they attempt to recognize - on the fly - handwriting made in the air [13].

C. Choice of Methods for the Classification Process

In continuation from the previous section, here we will discuss our motivation behind our choices for some of the methods we are using that are about the classification process. In that sense, some of the decisions we had to take were for:

- The learning strategy,
- the classification techniques and,
- how to evaluate our system.

In short, we used a Writer Dependent strategy, with models that we trained for each user separately. These models were one-class SVMs and Isolation Forests, and we evaluated them based on their Area Under the Curve (AUC) score and Equal Error Rate (EER). Also, the classification process is discussed more thoroughly in section IV.

Classifier Learning Strategy: The first important distinction that we have to make as far as classification methods go, is about the learning strategy for building a classifier. According to signature verification literature, there are two main ways to go about it, a Writer Dependent (WD) System and a Writer Independent (WI) one [14], [15]. Writer Dependent (WD), means that there is a specific classifier trained separately for each user we want enrolled and Writer Independent (WI) indicates that the system is connected to a database of templates, and when a user makes a query and claims an identity, both the query sample and one or more templates are used to make the final judgment [14], [15].

For our case we think that the most suitable way to go is with a Writer Dependent system, mainly because it fits better with our description of the problem. That is because in the cases that this system might be used, there will probably be the drawing of only one user available. In addition to that, WD systems have been employed successfully in other areas of behavioral biometrics [1], [16] but not so much in air-drawing, which makes it worth exploring.

Classification Methods: As far as which models we will be using in this research, it will be one-class SVMs [17] and Isolation Forests [18], [19]. These are one class models, that can learn to model a given training set of drawings (represented by more high level features). They do this by either learning a decision boundary (one-class SVM) or by creating anomaly scores (Isolation Forest). Such model-based approaches, heavily rely on the features that describe the drawings. Also, in our understanding according to [2], model-based approaches are mostly the case in signature verification literature when it comes to Writer Dependent systems.

This connection with the learning strategy is the main reason that we selected a model-based approach instead of a distance based one. Also, the models are one-class mainly because of the absence of the negative class in a practical case. In addition to that, the choice of one-class SVM and Isolation Forest is well connected with our goal of evaluating different kinds of features, since we can train the models with various sets of features and see with which set the models perform best. It would also be interesting to see the performance difference of the two models with the various sets of features or if there are differences between participants.

In addition to that, it needs to be stressed that even in the case where there was a dataset with the negative class, probably the one-class models would still be the better choice. That is because the negative class is theoretically huge if we would account in it all the people that can successfully make an air-drawing, which would make it really hard for the negative class to be represented. Finally, among one class models, one-class SVM and Isolation Forest seem to be among the most powerful - even with high dimensional data - and among the easiest to use since there are some clear and robust implementations in open source libraries.

Evaluation Methods: According to our research in signature verification and other behavioral biometrics, the typical evaluation of such models is usually being done with False Rejection Rate (FRR) and False Acceptance Rate (FAR). The former refers to type I error which is also called False Negative Rate (FNR) and the latter refers to type II error or False Positive Rate (FPR) [2]. Because the two types are highly related since by trying to drop one the other increases, usually the Equal Error Rate (EER) is considered as the overall error of the system, that is the error of the system when the FRR is equal to FAR [2].

According to reviews like [20] and [2], more recently in online signature verification the Receiver Operating Characteristic (ROC) curve seems to be widely endorsed for evaluating performance, which has True Positive Rate (TPR) plotted against False Positive Rate (FPR) for different threshold values [21]. That is mainly due to the probabilistic interpretation of the Area Under the ROC Curve (AUC), which is also the metric we will be mostly using in this study [2], [21], [20].

The AUC score is interpreted as the probability that a model will score a random positive sample higher than a random negative one and not as the probability for an instance to be classified correctly [21]. That thing makes AUC an adequate metric to assess the performance of models but in order for the models to be used in practice, a suitable threshold needs to be selected as well. In addition to that, because the ROC curve is expressed in ratios of only positives (True Positive Rate) and only negatives (False Positive Rate) in the two axis, its corresponding AUC remains unaffected from highly skewed test sets [21], which is also the case in this research.

In addition to that, there are three other interesting metrics that are sometimes being reported: accuracy, precision and recall. Accuracy is simply the percentage of the correct predictions to all the predictions made, precision is the ratio of True Positives to all predicted as Positives (meaning how many of the predicted positives are indeed positives) and recall the ratio between True Positives to all Positives (including those that were predicted as Negatives) [21]. The main reason that we will not be using any of these three metrics is because our dataset for every participant is heavily skewed with locks of the negative class (which is all the other participants).

D. Major Concerns

Due to the nature of the study of behavioral biometrics which is the area this research belongs, as well as our scope and given resources, we had some concerns under which our results need to be looked at. The key issue is that behavior can change over time for various reasons or simply be inconsistent for whatever reason. For example a user might learn the system better and change his behavior, or he might just be inconsistent for any number of random reasons that have to do with the environment or his state of mind. In this section we discuss inconsistency under these two perspectives.

The Learning Factor: One can easily see how a learning factor is involved in such a system by thinking how people naturally can improve at any task over time. It is also the case that a system like that makes sense only when its users have some familiarity with it (first and foremost, locks need to look like locks). For that reason, before the data collection, we had a training session with all participants in order to make sure that they understand how the system works, how they can do the drawings, and also to improve the drawing skills up to a point that their lock is acceptably close to the lock of Figure 1. However, it is important to ask what is the impact of this factor to our problem more broadly and over time?

In terms of performance, it is difficult to judge how the learning factor would impact the performance of the models. One could argue that a more experienced user will be more consistent and therefore his model will perform best. However the user might learn the shape very well and he might have very few characteristics and nuances to it, which would make it very similar with all the other experienced users. A dystopian - and probably the most challenging - scenario in that sense, is one where the system gets adopted by a group of people, matures within their community and that leads to all users start making very good locks, closely resembling the one in Figure 1. Then it would potentially become much harder for any model to decide whether a given drawing is by the user it is trained on, or someone else.

From that, it seems that the classification results should be interpreted in light of the experience of the user group where the classification is performed. Partially because of that, in this study we tried to train every participant to a level that he can start making acceptable locks, by holding training sessions prior to the data collection session for about 10 to 15 minutes, and we avoided having participants with more or less experience than that. The training was with visual feedback of the user's drawing trace, in order for people to quickly pick up their mistakes and correct themselves - we discuss training more thoroughly later on. Another reason for going with lightly trained participants is time constraints as the aforementioned challenging scenario requires working closely over long periods of time with the users to improve their drawing skills.

Besides of the impact of the learning factor on the performance of the system given the environment (and how other users become better at drawing the same shape), it is important to see that the performance of the individual system is also subject to change, according to how the system updates its models or its knowledge of the user while he is learning. Given that with learned behavior we refer to stable behavior that is acquired by a user based on his understanding of the shape and the process, it would be possible for the models to adapt to a user's new pattern by re-training themselves in light of more data. In this research however, we do not deal with this issue at all, given that we collect data from one session only. It simply seems logical though, that as long as there are such stable behavioral changes they could be easily assimilated.

Drawing Consistency: In addition to the learning factor, a very close related issue - albeit quite distinct - is the drawing consistency users might have over long periods of time. This might have to do with the user just being inconsistent or if the user forgot the drawing after some time not practicing it etc. This is different from the stable behavioral change acquired by learning that we discuss above, although the learning factor might also have an impact on this issue in case the more someone learns the less other things effect his behavior.

It is possible that participants make very different drawings each time that they are using the system. This can happen as we said merely from the different choices they make from time to time, or because of environmental and psychological factors. A user can for example choose to draw with a prolonged arm one day, and with the arm close to the chest the other day. Or someone's typographic styling might be different each time according to his mood or other things. Nonetheless, this issue is out of our scope in this research as it would require long term monitoring of the participants behavior for which would need to largely increase the scale of this project.

Our study does not take these factors into account and of course the results need to be interpreted under this perspective. In fact our results will rather display the capability of such a system to verify a user, given that the intra-person variability in the drawings within a session is not much different than the drawings between two days or even longer periods of time. In order to leak as much intra-person variability into the drawings as possible though, and to avoid users mechanically standardize their drawings in every iteration, we made sure that during the data collection sessions the participants would reset their positions multiple times - as we will also discuss in a later section.

Research in permanence of behavior in similar cases shows weak signs that such permanence exists. In [22], a very close related study was conducted for online signature verification where the participants were verified by the signature that they were drawing mid-air with their smart-phone device. In that study there were eight users that participated in 20 sessions over a period of two months and signature samples were collected from them. The authors then conducted a permanence analysis with some dynamic techniques, in order to see the extent to which the participants retain their behavior. In the end, it seemed that there were two prominent groups of users: one where participants were consistent in repeating their signature, and another one where there were significant differences between drawings of the same participant in different sessions.

The same problem is apparent to other cases regarding behavioral biometrics. For example in [23], identification is attempted by measuring the arm-swing acceleration and the motion of the same person fluctuates even among trials. Also, in a 2004 review [24], where different biometrics are ranked in various characteristics, two prominent behavioral biometrics like Keystrokes and signature making are ranked as 'Low' when it comes to permanence.

While the analysis in [22] and [23], or the review in [24] shed some light into what happens, they do not go into why it happens which is probably the harder and more complex question. Given that we knew the circumstances under which permanence could be achieved, we would study air-drawing under such circumstances. However, since this is not a very well studied area and as we said this issue is out of scope for this study, we will proceed with the assumption that permanence is possible.

E. Research Questions

In order to be clear and methodical in approaching this problem, we need to be explicit in terms of extending the purpose of this study that we outlined earlier, into a research question we can quantifiably answer. Such a question would be: "Can someone be verified accurately by the way that person air-draws a not too complex symbol like a lock? And if so what are the right features, the right model and how much data do the models need to converge?". That is decomposed into the following four elementary questions which we will try to answer:

Question 1 - ***To what extent a user can be verified to have made a lock (Figure 1) drawing in the air ?:*** In order to find out how accurately we can do verification, we will use the locks we get out of 16 participants and train one-class classification models with them. We will then measure their performances with Area Under the ROC Curve (AUC) metric [21] as well as Equal Error Rate (EER) and we will use them in order to draw conclusions.

Question 2 - ***How do different kinds of features help the system perform ?:*** For this question we need to make various different sets of features. Then the aim is to test them in isolation or combine them in groups and find how they impact the performance of the models. Also, it is maybe important to stress here, that our goal is to evaluate the different features and not the method by which we acquired them which is in many cases custom made for the lock shape.

Question 3 - ***How do different classification methods compare to each other ?:*** In this research we will try out two different types of models, an one-class Support Vector Machine (SVM) [17] and an Isolation Forest [18], [19]. For each model we will see its performance in all the aforementioned cases (the different users and sets of features as well as its convergence rate), and we will try to identify the areas one method might have the edge over another.

Question 4 - ***How fast does a classifier learns the model of a user ?:*** For this question we will be making a convergence rate analysis. For that we will train models with different numbers of samples and see the performance in terms of AUC while the samples increase. Also, besides the participants that we already have referred to, we have data of 590 drawings from one extra participant that we will also use for the convergence rate analysis.

II. Previous Work

The two main challenges of this research were to first build the dataset of air-drawings, and then train and evaluate classification models with that data. As far as building the dataset goes, this involves capturing the trace of a drawing as a time-series of positions with a Kinect v2. For training and evaluation, we took into consideration previous research on the matter to find out the ideal models to build and compare. Given our research goals, as mentioned earlier, we went with one-class classification models, and used the one-class SVM [17] and Isolation Forest [18], [19].

We will more explicitly discuss the methods we are using in the following sections. In this section we will discuss other research on the topic of air-drawing, online signature verification, and a little more general in terms of behavioral biometrics. We will mainly try to highlight similarities and differences with this study and others. In addition to that, when it comes to air-drawing we will also discuss potential uses of the Kinect in research. Before looking more into the specific studies though, it is worth making an overview over the classification methods that are being most used in the bibliography.

For signature verification by air-drawing, the most popular method seems to be Dynamic Time Warping (DTW). The studies that are referred to here are in fact all using DTW [3], [4], [22] one way or the other. In [4] the DTW approach is compared with another approach based on Hidden Markov Models (HMM) and in [22] again a DTW approach is compared with one using an HMM and another one using a statistical (Bayesian) classifier. In both studies the DTW approach seems superior. The area though is largely under-researched and not too many methods have been tested.

On the other hand, online signature verification is much more well researched. A late review in [2] mention plenty of methods. However, it is hard to pinpoint the best performing method or study, because of the different datasets, features and also the evaluation methods that are being used. In spite of that, it is fair to say that DTW and HMMs seem to be the most popular techniques. DTW refers to distance based methods, and HMM to model-based ones. Later in this section we will discuss more closely some interesting cases.

In addition to that, it should probably also mentioned here that in some cases in bibliography the term authentication appears instead of verification. This is sometimes the case in more recent studies. To our understanding the terms are being used interchangeably and we hereby stick to the term verification (unless we are quoting the original text). That is mainly because it seems to be consistently used for a longer period of time across signature verification and air-drawing research.

A. Research in Air-drawing and Drawing Trace Capturing

Research on the exact topic of verifying an individual by air-drawing is sparse but not non-existent. In [3] an air-drawn signature is captured by the depth camera of a Kinect v2. The signature is being drawn by the index finger as the tip of the finger is tracked, just like our case. Also, it is fair to say that signatures are in general more complicated than the simple lock shape of Figure 1 that we are using. After capturing the trace, verification is attempted by the means of DTW in a Writer Independent classifier style. The precision and recall rates of this study are 100% and 70% (worst-case scenario) respectively. This research, besides its methodology for classification, is very similar to our case as they also build a software application to capture the signatures with, that goes by the name KinWrite.

Another study that utilizes a depth camera to get the data for verification is [4]. Similarly to the previous research, the verification is also between signatures but the technology used to capture the trace is a Leap Motion Controller (LMC). Yet again, the signatures are drawn in the air by the index finger. A concern one might have in regards to the use of LMC is that due to its narrower field of view than for example the Kinect's, it poses some constraints for the drawing size a user is allowed to make. This, does not cause any problem though, as the users familiarize themselves with the system and the drawing box with a visualizer, similar to our case. In terms of the verification methods, two approaches are followed, a DTW one and a HMM one, with accuracies of 95.5% and 90.0% respectively. Both approaches use some higher level features constructed by the lower XYZ points from LMC.

The two previous studies capture the air-drawn trace by depth cameras. However, this is not always the case since there is some research for signature verification drawn in the air that captures the trace with Inertial Measurement Units (IMUs). In [22], users cast their signatures mid-air with a smart phone, and the acceleration trace by the phone's IMU is tracked. For the classification part, three methods are used, an HMM, a statistical - Bayesian - classifier with mathematical features, and DTW. The Equal Error Rates (EER) of each method is reported which corresponds to 5.96%, 14.09% and 4.58% in the stress situation of impostors signatures; notably the models or templates were trained with only up to five instances.

In terms of the technology we use in this study to capture the drawn trace, we have already stated that it is Kinect v2 for Windows. This technology besides that it has already been used successfully in an air-drawing authentication scenario in [3], it is also being used in other fields such as robotics [25], Sign Language recognition [9] and others. In [25], Kinect allows a visual servoing robot do visual tracking. In [9], Kinect is used for sign language recognition and is compared with another technology for the same purpose that involves IMUs combined with color tracking.

B. Research in Online Signature Verification

Research in the field of Online Signature Verification is well documented in a series of reviews done in 1989, 1993, 2000, 2008, 2012 and 2014 [26], [27], [28], [29], [20], [2]. From the systems that are discussed in the reviews, the ones that build models for users separately, are the ones closest to our case (sometimes called Writer Dependent systems). Systems that take a distance based approach are very popular though, similar with the air-drawing literature of the previous section. In addition to that, given that research in online signature verification refers to different types of forgeries, and in our case the lock symbol is always the same for all users, the approaches that refer to skilled forgeries are the most relevant to us.

As far as some interesting studies of online signature verification goes, in 2004 the SVC2004 competition was held, with teams competing into the same dataset at two different tasks [30]. The first task included only positional data and the second also had pen orientation and pressure data. The best approach when tested with skilled forgeries achieved an EER of 2.84% and 2.89% for the first and second task respectively. When tested with random forgeries, the EER score for the first task was 2.79% and for the second 2.51%.

The methods that have been used from the winning team are described in [31], where they are also evaluated with another dataset. In short, the features were constructed with the help of DTW and then the problem is treated as a two-class classification one. In that sense, a linear classifier combined with Principal Component Analysis (PCA) is compared with a Bayesian classifier and a two-class SVM. The aforementioned linear classifier in conjunction with the PCA method seems to be performing best with the dataset of [31], with a FRR and FAR of 1.64% and 1.28% respectively, and is also the method used in the competition [30].

In [32], a Discrete Wavelet Transform (DWT) method is used to decompose an online signature into sub-bands from which individual features are extracted. According to those, for enrollment a template is created and for verification an Adaptive Signal Processing technique is utilized to capture the similarity of a processed signature with the template. According to the similarity, a decision is taken with a non-linear function for whether the new signature is genuine or forgery. The reported EER of this approach is 4% on a dataset collected for this research.

In [33], a three stage verification technique is used, where distances from three templates are found for three different kinds of features (global, local and point-to-point matching). A decision for whether a signature is genuine or forgery is taken independently in each stage with a different decision function. The stages are applied in a sequence and a signature is accepted as genuine when it passes all three stages. A FRR of 5.8 and a FAR of 0% is reported on a dataset collected for the purpose of the study.

All the aforementioned studies are characterized as distance-based by [2]. That is because there is an enrollment phase where a template is created, and according to the method used each time, for any new signature the distance to the claimed user's template is calculated. Then, a classification technique is applied to make a decision on whether the distance is too high to characterize the new signature forgery, or too low to call it genuine. In the case of [31] where a two-class SVM model is used to take the final decision, one could say that the model is trained to model the similarity for when it is too high or too low according to all users in database and the specific feature used. That is different with our case where we are using one-class SVM models, to model user behavior according to parameters.

Despite the seemingly good performance of the distance-based approaches, according to [2] it is systems that build models for each participant separately, and do not construct templates that have the edge in performance. In this study our approach is also model-based, as we train and evaluate one-class SVMs and Isolation Forests across our participants. However, our models are based on parameters and not on functions as it is usually the case in the model-based studies of online signature verification. Two of the model-based approaches that seem to have the better results in [2] are [34] that models functions and [35] that models parameters.

More specifically, in [34] an HMM model is build for every user based on 5 initial signals of the training samples, that are captured from the data collection device. These signals refer to position (x,y), pressure, inclination and altitude as a function of time. The average EER across all participants on skilled forgeries is 0.35% when a user-depended threshold is used for the decision function.

In [35] instead of signals, global and local features are combined for user-dependent models. There are 23 global features, that have to do with statistical properties of the signature data, like total time, length-to-width ratio and others, that are used to build a statistical model for each participant. We also use global features in this research but they are quite different than these ones. As far as the local features go, they are created after segmenting the signature into parts that characterize each state of an HMM. After combining the information by the two models that rely on the global and local features, the EER reported by the research is 2.5%, as evaluated on a proprietary database.

C. Research in the broader topic of Behavioral Biometrics

In terms of the larger area of Behavioral Biometrics, there are some lengthy taxonomies such as [36], and it is not very feasible to present a comprehensive view in a few paragraphs. However, research in keyboard and mouse dynamics, seems to be the most relevant and interesting for our case. It is also true that very similar methods to the ones of signature verification are used in general for the various verification systems.

By cherry picking studies that display good performance and are relevant for their methodology to ours, we would say that [1] is a mouse dynamics study that effectively leverages Principal Component Analysis (PCA) and an one-class SVM among other things, to achieve an FAR of 8.74%

and an FRR of 7.96% in 11.8 seconds authentication time and 32 samples by users. However, this result can be greatly reduced to 0.87% FAR and 0.69% FRR by increasing the authentication time and the samples produced by the user to 800. Additionally, [16] is a keyboard dynamics research using an one-class SVM among other algorithms and various features for which it achieves EERs as low as 1%.

III. Data Collection

The first step of this research was to collect data from different participants that we could make models for. In order to do that we made a software application that utilized MS Kinect v2, and can capture the drawing trace of a finger. Then, we used that system in the context of several data collection sessions in order to acquire the data.

The reason to make the application and conduct data collection sessions was to meet the specifications of our problem both in terms of the data but also the process that would produce them. Specifications that to our knowledge we could not meet easier (e.g. with a ready made dataset). More specifically we wanted:

- the positional data to a format that we could process in order to find features and plot in order to decide if the lock is indeed a lock or not,
- a frictionless interaction as we have described it previously,
- a way to be able to use the system in a training mode where users could correct themselves with visual feedback on the locks they are making.

By taking these things into account we made a data acquisition system (the aforementioned application) that we used in the context of some data collection session, in order to answer our research questions. In this section we will explain how we made that software, and how we conducted the sessions with it.

A. Application and Drawing Trace Capturing

As it has already been said, in order to capture the drawing trace we are using Microsoft Kinect v2. The purpose of the system is to produce a csv file with a timeseries of XYZ points that plotted together in sequence can reproduce a lock that a participant drew. We also aimed for ease of use since we expected the user to manipulate the system in a slightly theatrical manner to produce the locks. The manipulation is along the context of the data collection sessions that is described later on. Most importantly, the system should facilitate capturing the instance, give some visual feedback of the drawing trace, and work in two modes, one for training and one for capturing locks.

In order to build the system, we used the Processing language [7] and the Open Kinect for Processing library [8]. From that library we got the Kinect's depth image in every frame. We also used g4p controls [37], to get an additional display window in the Processing environment. In order for the Open Kinect library to interface with the device, we used the libfreenect2 drivers [38]. In terms of hardware, besides the Kinect v2 device, we used two monitors (one laptop monitor 13" and one standalone monitor 22"). From the Kinect we used its depth camera, that has a field of view of 70×60 degrees, a depth image resolution of 512×424 and has a reach of 0.5 to 4.5 meters [5].

The laptop monitor displayed a "console screen" (left side of Figure 2) which facilitated the training process since it showed the depth image by Kinect and most importantly the drawing trace a given participant was producing. That monitor was available to the user only during training. Also, the software application was run on that laptop and the other monitor along with the Kinect were attached to it.

The other monitor was the user's monitor that displayed only the lock of Figure 1. It was used to co-ordinate the flow of the session so the participant can know what do to when. The display screen can be seen in the right side of Figure 2 and signaled three things: when the status was idle and there was no recording (Figure 2a), when the trace was being recorded (Figure 2b), and when the recording was completed successfully (Figure 2c).

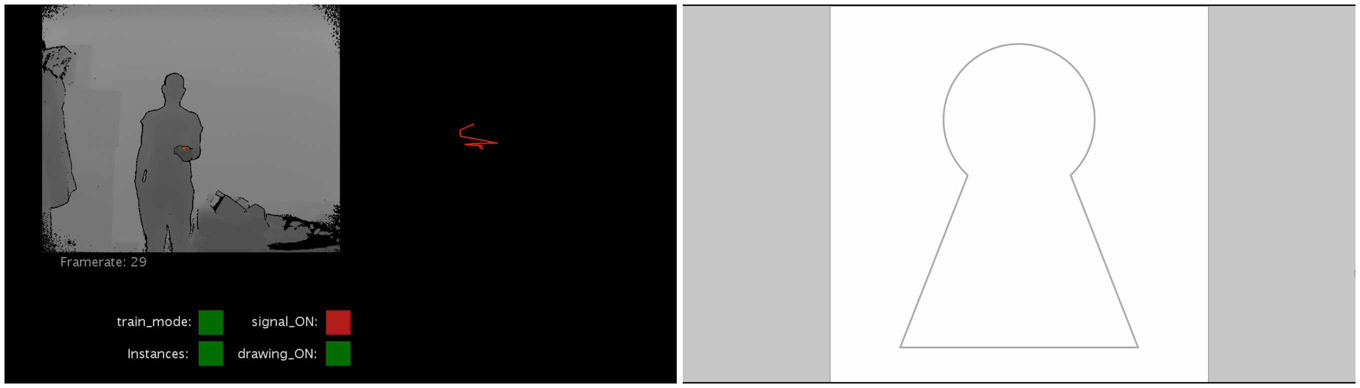
Besides giving feedback to the monitors, the main purpose of the software was to listen for a gestural signal by a user and to keep track of the closest point once the signal gesture was made. A signal could be made by holding someone's finger still in front of the Kinect, given that the finger is the closest thing to it, which it was because it was taken care of from the set up of the environment as well as the intention of the participant (for which he was instructed to).

When the software was to find out that the closest point did not move for a couple of seconds, it would immediately, remove the lock from the main monitor and start recording the position of the finger (given that it remains the closest point while drawing). When the user was to make the signal again, then the software would drop the trace into a csv and conclude the documentation of an instance. Then the user would have to reset his finger, and signal again for the next drawing.

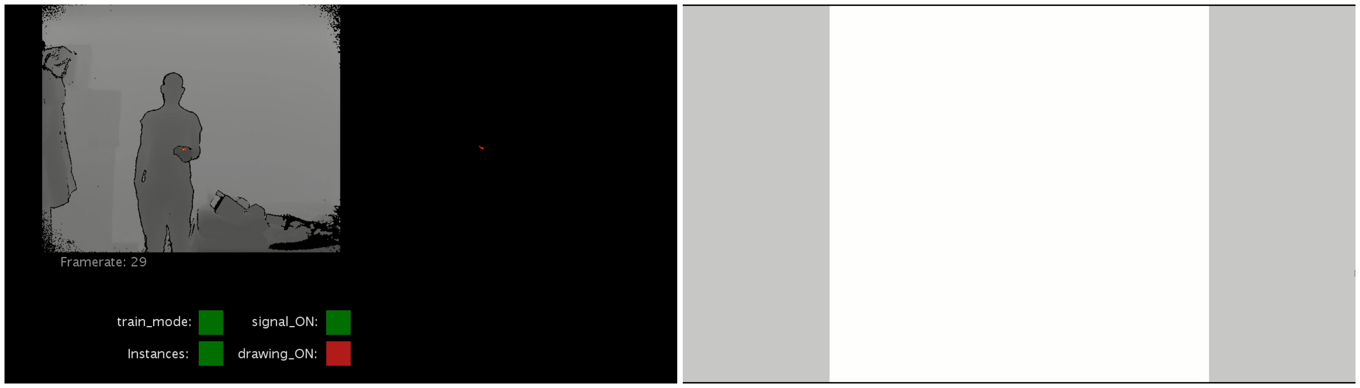
In that sense, the basic cycle of moves the participant would have to do to make a lock would be 1) signal to start, 2) draw, 3) signal to save, 4) reset the hand. Resetting the hand was happening to avoid mechanistic drawing and capture the natural intra-person variability as we have mentioned earlier. Screens from different stages of this cycle as captured by both the feedback monitors can be seen in Figure 2.

During the drawing part of the cycle, the application was designed to make the symbol disappear from the user's monitor. That was the case for three main reasons. First, because this serves as feedback to the user to start the movement, and also because it prevents cheating (by following the drawing with the hand). Another reason was because it is closer to a real world situation. For the same reason we did not allow the participants to see the console screen during the production of locks.

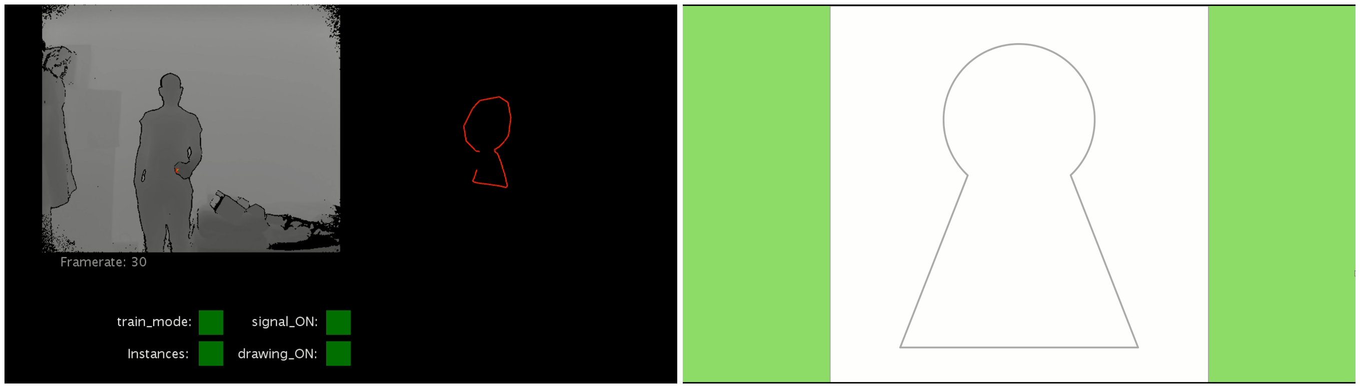
Another important aspect of the software application is the representation of the data in the resulting csv. From the OpenKinect depth image object, we were getting the raw depth that corresponds to every pixel of the image and has a



(a) Console (left) and display (right) screens of the application before making the signal to start drawing.



(b) Console (left) and display (right) screens of the application just after making the signal to start drawing.



(c) *Left*: Console screen that shows someone that has just completed the lock drawing. *Right*: Display screen that shows the celebratory feedback that the lock has been saved successfully - this happens a moment after the drawing csv is saved.

Fig. 2: The console (left) and display (right) screens in various phases of the data collection cycle.

range from 500mm to 4500mm, at 30 Frames per Second [8]. Also, the same library had some functionality to translate the x, and y pixel of the depth image to the real world dimensions given that the Kinect camera is the start of the axis. That translation was based on specifications of the Kinect physical device. In order to leverage this functionality to construct the csv with the drawing trace, for every frame where we had updated values from the Kinect, we were taking the x, y, z and a timestamp values and we were storing them in a row. Afterwards, with the user's signal that he finished drawing, we would produce the file.

Finally, another issue we had to deal with when building the software was to make sure that we get hold of the correct pixel in every frame - which corresponds to the closest point to the camera. That is not as straight forward as looking at every pixel of the OpenKinect depth image object, because there is noise in the form of random pixels that usually are at the edge of the screen. They sometimes appear to have a value remarkably close to the camera. However, since these pixels only sparsely appear to the depth image, the way we dealt with it was by checking a pixel's neighboring pixels before crowning it as the closest.

B. Lab Environment Setup

In terms of the setup of the lab environment, we used a spacious room where we put a camera four meters from a wall. The users were instructed to sit 1.3 meters away from the camera, behind a line that was drawn on the ground. Such a position, forces the drawing to happen in the area where the Kinect is most accurate [6]. A crucial part of the process was to familiarize the users with the system and how their drawings looked like, until they were to find a drawing rhythm that felt comfortable with all interaction aspects and the entire process. After some training the data collection process would start.

The interaction was confined within the lines of a straight standing pose with the movement freedom to start from the shoulder and onwards. Furthermore, we instructed the users to start drawing from the bottom left corner of the shape (green point in Figure 3b), which was a fixed point for every participant because as we have already said we wanted to remove the element of choice in the non-behavioral aspects.

In order to train the users we used the console window (right side of Figure 2) and we showed them some visual feedback of their trace. We then guided them on how to use the system and we encouraged them to play with it by making some locks or other shapes that would help them get a feel for it. When the users felt comfortable with the application, we stopped the training process and started the lock collection session. We then asked each user to make at least 60 locks with their index finger.

In addition to that, between every drawing, the participants were also asked to reset their hand in order to make a complete movement all from the beginning. In addition to that, four times during the session - approximately when a batch of 15 locks was completed - they were asked to pause lock making for a while and take a walk in order to have a break. The users are also encouraged in beforehand to target the camera like a door they want to open and to keep the rest of the fingers in a fist due to the fact that what gets tracked is the closest thing to the camera and therefore the index finger needs to stick out.

The data collection sessions were held in the span of five days with up to five users per day (a total of 16 users participated). All users were right handed, aged between 21 and 44, and among them there were 8 women and 8 men. Each person contributed from 71 to 104 drawings, from which not all were good locks. From each participant we ended up using 60 good locks to build models.

In addition to that, we had one extra participant (not in the pool of 16) that made 590 drawings within the span of four sessions that we would use when making the convergence rate analysis for the models. That person was not considered among the 16 for any other experiments because he was much more experienced than them. Finally, the conditions in the room regarding camera placement, drawing position etc were kept the same for all sessions.

IV. Data Analysis

After having the raw data for 16 participants we did our testing in three basic steps. First, we did some pre-processing on the raw lock data, then we constructed some high level features that we grouped up in various combinations, and finally we trained and evaluated a number of models on how they can predict locks from both the positive and the negative class. Also, in addition to the 16 participants that gave us at least 60 locks each, we had and an extra participant that gave us 590 drawings with which we analyzed how many training samples do the models we used require to converge. We did not include any data of that participant in any other case though, because he had much more experience in drawing than the other 16.

For the final step which is the model construction, we trained an one-class SVM and a Isolation Forest, for every participant and for every group of features. For the Research Questions 1, 2 and 3, we trained models with data from the 16 individuals (one one-class SVM and one Isolation Forest per user and feature group), and evaluated them with positive and negative samples. The positive ones were locks of the same person that were not used for training, and the negative ones were all the remaining locks from the other 15 users. More specifically, what we did for each of the first three questions was:

- RQ 1: For the first question we looked into the performance of each user across the different feature groups that were used, to see the ability of a model to verify a user.
- RQ 2: In the case of question 2 we looked at the groups of features and tried to identify the best performing ones.
- RQ 3: Regarding question 3 we looked at the difference between the performances of one-class SVM and Isolation Forest across all models.

At the model construction of the fourth question the approach was a little different. Our goal was to investigate the extent to which the models become better with more training samples. The problem was that with the samples from any of the existing 16 participants, we could only see it as far as 40 training samples (because we would at least need the remaining 20 positive samples to test with).

For that reason, as we explain later on, the data that we used for the convergence analysis was from the one additional participant that had contributed 590 drawings. Also, we did run the analysis under all the different feature groups, for it would be interesting to see how does the convergence rate fares in cases with high and low dimensional data.

A. Pre-processing

Before getting into the analysis of the data, the first step was to do some pre-processing in order to prepare them for the feature construction step. Given that the data already came in a nice format as we explained earlier we did not do more than four operations on the raw data. In addition to that, in this step we had to make make a first manual scan of the locks to see that they are indeed locks.

As a matter of fact the pre-processing steps that we took for every raw lock file were:

1) We reseted the coordinates of the data points so that the start of the axis was the front-bottom left corner of the interaction box,
2) chopped the front and the back tail of the drawing trace,
3) removed the duplicate points,
4) interpolated some values that were very much off the mark (sometimes there were glitches in the data collections that we accounted with this step)

Our process for the preprocessing step was very straight forward. By going through the locks of every participant separately, we were performing those actions for every lock. Afterwards we were plotting the lock, and we were judging whether to keep it or not. The decision was according to our intuition in regards to the extent some of our criteria were satisfied.

There were two criteria that we had for keeping a lock instance or not. The first was that the shape was indeed a lock (because there were some miss-drawings during the data collection process that we knew we had to toss out). The second one was that some Points of Interest (PoIs) that we were finding later in order to create features were correctly identified. We had to inspect the files for that, because our method to create those PoIs was not very robust and we could not be sure that it would work in all cases (in the end, in very few cases - less than 1% - it did not work).

It is out of this process that we created the dataset that had the locks we deemed eligible to train with. In the following paragraphs we will try to explain more in detail what we did for each pre-processing step, and also how did the selection looked like and how it worked out.

Step 1 - Coordinate reset: The reasoning behind the first step of the operations that we applied on the data, was only to make them easier to work with - be that plotting or making calculations. As has already been said, the raw data came with the physical device of the Kinect as the center of axis. In order to change that, we used the functionality of the OpenKinect library [8] to find the x dimension of the interaction box, and added half this value as offset value. Afterwards, we did the same for the y axis, but instead of adding we subtracted the y offset and took its absolute values. The process was different because the raw y axis was inverted.

Step 2 - Chopping the sides of the trace: The reason chopping the sides is important, is that as we explain before, the way to capture a trace is by having a user signaling to the system by keeping the finger steady for roughly 2 seconds. Some users, however, do not respond immediately and hold their hand longer. Because of that, there is the fear of leaking the reaction time as a characteristic in some of our features like for example the 'total drawing time'. However, the reaction time is a behavioral aspect we do not want because it refers to the response time and not the drawing behavior.

For that reason, in order to account for the response time which is between the start of the signal and the start of the actual drawing, we chopped a varying amount of points from the lock shape of each drawing. In order to find the correct amount of points to drop we made use of the fact that the person, while idling, is still holding the finger still. We set a threshold of 20mm and by going through the points of the trace (from the second and onwards), we would stop at the point whose euclidean distance with the very first point of the trace exceeds that threshold. Then, in order to not accidentally throw away any relevant information we tossed the trace up to three points behind the point with the high distance.

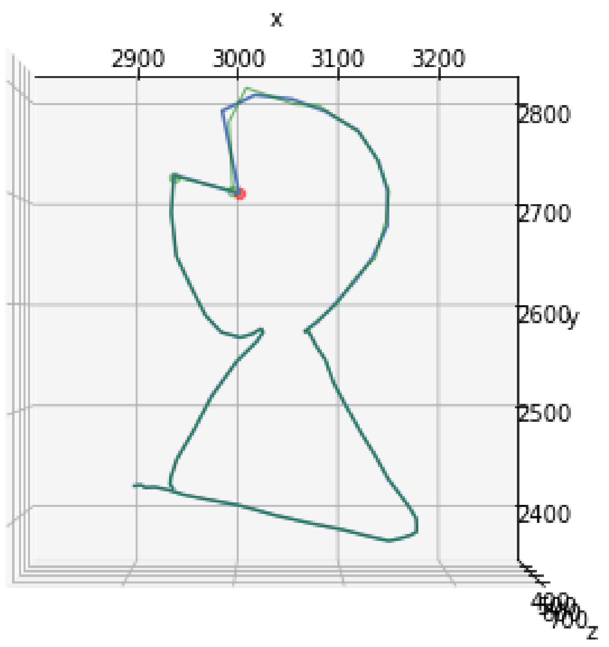
The way we set the threshold and took the decision what to toss was mainly through trial and error. Our main objective was to chop only when necessary and not throw away part of the trace that was relevant. In addition to the front chop we also made a back chop that was a fixed 50 lines. That was because they were redundant since they represented the signal that was 'closing' the lock (Figure 2c) and they were not part of the drawing.

Step 3 - Duplicate points removal: The third pre-processing step that we took was to remove the duplicate points. Sometimes the Kinect would produce the same exact point in different timestamps. In those cases we kept the oldest point and threw away the other. The main reason that we did this was because it made calculations easier and also helped the filter to perform quite better when it came to the next step which was smoothing. We also had a concern that by removing the duplicate points we might slightly impact a few features that come out of the simple distributions of the pre-processed points. However the impact should be very small and it is hard to say if it would be for the better or worse.

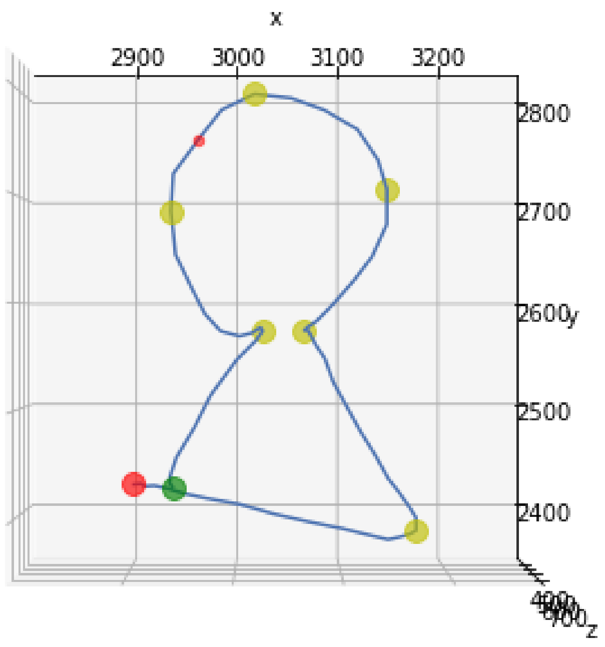
Step 4 - Interpolation on wrong values: The final operation on the data was interpolation. That was because in some cases there were glitches and the Kinect would pick an off point in the wrist or if someone extended slightly one of his other fingers for a brief moment or something like that. In these cases where one or two points in a row were very off we could use an interpolation technique to restore the shape of the lock. That way we could use more samples, and we could also improve others that we would have used in any case.

In order to do that, we made a new lock shape with a g-h filter [39] and we superimposed it on the raw data. Then by going through the points of the drawing traces one by one, we were finding the cases where the distance between the same point in the smooth image and the raw image was too large. Next, if the distance was large enough (over 3mm) we were finding a new position for that point, according to its adjacent points. Finally, if the distance between the new and the old positions would be substantial (over 20mm) we would go on with the interpolation.

The filter that we used had g parameter of 0.9 and h parameter of 0.9 so it only very slightly, in cases of big anomaly (like a very abrupt glitch as the one of Figure 3a),



(a) The raw lock in blue and the superimposed smoothened (almost on top of each other) in light green, with the red details which are points found to need interpolation.



(b) The interpolated lock with the points of interest (the small red point being the interpolated point).

Fig. 3: Images that determined the eligibility of a lock drawing for the final dataset.

it deviated substantially from the raw trace. Also, by looking at the conditions that we had set to make an interpolation, one can see that in a given drawing very few changes would be made - which was our goal. In fact with this smoothing technique less than 10% of the drawings received changes, and there was no image that had more than four of its points altered.

Selecting the eligible locks: After having processed an image out of these four steps, we made a software tool that displayed the images as shown in Figure 3. The tool then allowed to separate the good images from the bad ones with a yes/no user input. Only the good images were used going further.

In the images we can clearly see if the drawing is a lock in the first place, how the interpolation worked and how did our functionality for finding some Points of Interest (PoIs) worked. These PoIs were very important for the next section where we wanted to construct features. As we have said before, there was room for error in finding those points and that is part of the reason why checking manually if they were found correctly was important.

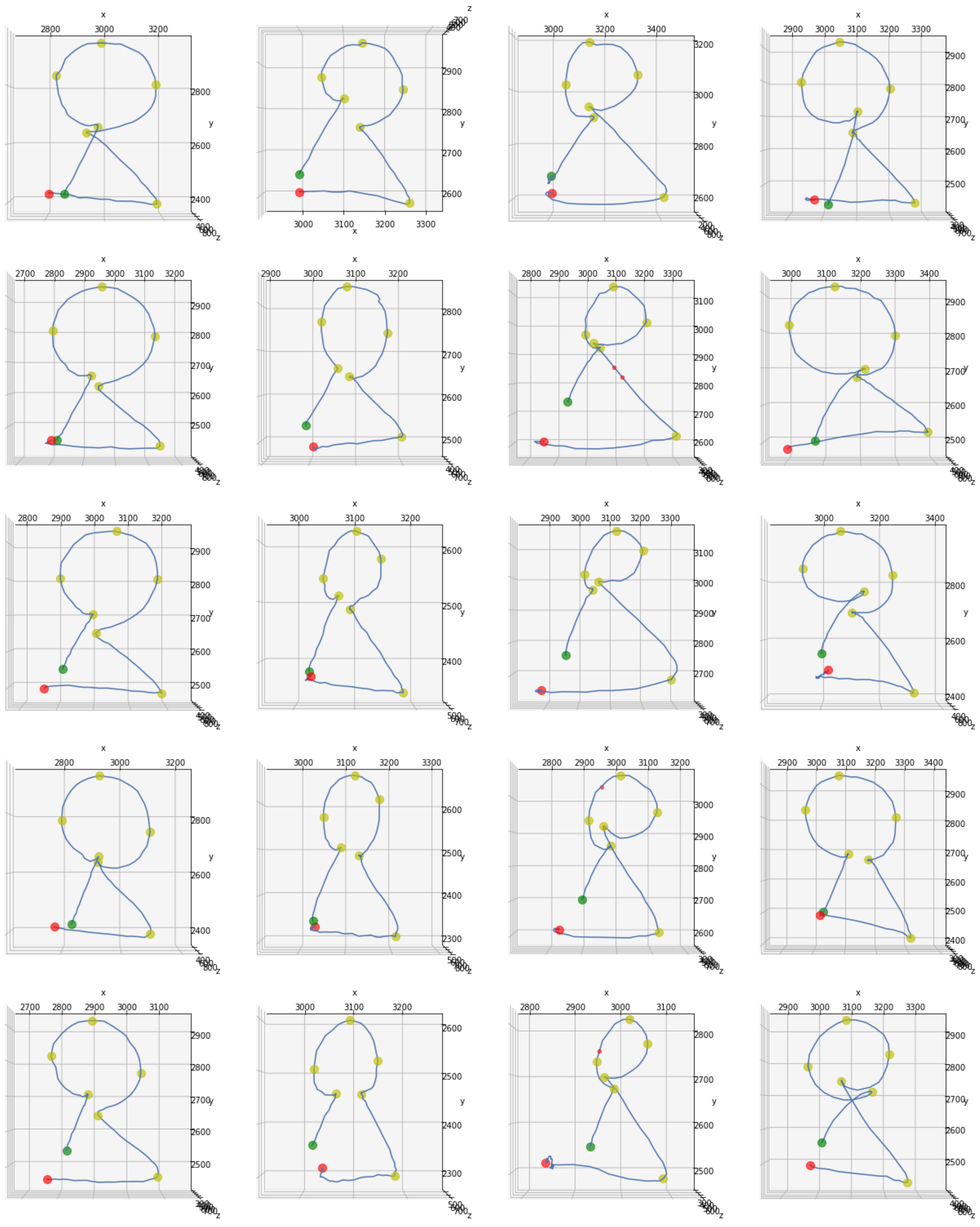
The criteria for selecting a drawing for the dataset that will be used later on, are exactly the qualities mentioned above. Mainly the drawing has to look like a lock and not garbage, and the PoIs algorithm needs to work. All in all we dropped 334 images out of the 1422 that we collected. In Figure 5 we can see a random sample of 12 locks that have been dropped.

On a final note, we need to mention that because we had more or less a varying number of locks for each participants, very rarely, we dropped drawings that were not too bad for the sake of better ones from the same participant. When this process was over, for each participant we had at least 60 locks and at most 79. In Figure 4 we can see examples of locks that were kept from four different participants as they were formed after pre-processing.

B. Feature construction

The next step after getting the pre-processed data was to extract more high-level features in order to do the classification with. As we have already said, the features we extracted used the geometric and temporal information of our data and we used them in different batches - in the end we tried 26 different groups as we explain below. The three main categories of groupings were: the global features, the distances, and the ratios. By all the groupings and experimentations we also aimed to find the most powerful kind of features for this particular problem.

The way we went to combine the different groups of features is the following. Based on the three aforementioned categories of features, we made 11 very basic groupings, which are for the most part clearly distinct from each other. Seven of those groups were from the global features as we derived them from the plain distributions of the points in the processed csv file. Two groups were comprised from distance measurements of some special points in the drawing that could describe it as lock (we hereby refer to those points as Points of Interest). Finally, the other two groups were ratios of the aforementioned distances.



(a) Participant 4. (b) Participant 5. (c) Participant 8. (d) Participant 13.

Fig. 4: Lock drawings that were kept from 4 different participants after pre-processing (different user in each column).

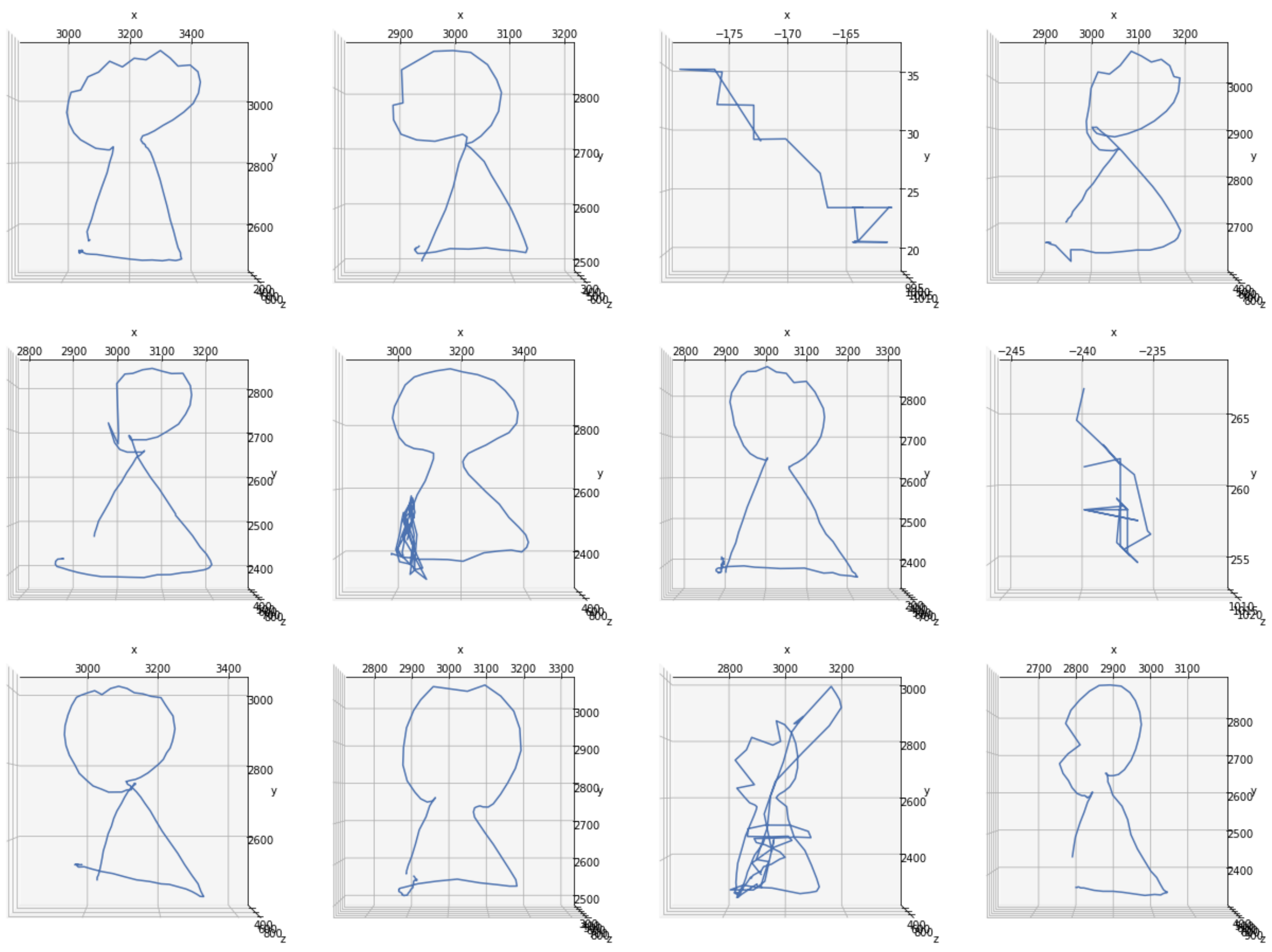


Fig. 5: Raw lock drawings that were tossed from various participants sampled at random.

On top of these 11 basic groups, we tried combinations according to their performance. For example if we would see that group 4 and 5 have a good score we would try them together, or if group 13 performs well but is very high dimensional we would try to see if a subgroup could also perform well. All in all, we experimented with 15 combinations of the 11 basic groups, which resulted in trying 26 groups of features in total. The details of each one can also be seen in Table I, where the groups of basic features are in bold letters.

In addition to that, we evaluated each group of features by their mean AUC performance across our participants, and we also tried to see if there were different participants performing better with different groups in particular. Following up with this section we will go to more specifics as to how we calculated the features of the groups of each category.

The Global Features: The reason we call this category 'the global features' is because in order to make those features, we just used the plain distributions of the xs, the ys and the zs from the entire csv of a lock drawing. We also made the distributions of their intervals, and then by dividing the intervals with the time differential (from the timestamp column) we got the speed distributions. All that resulted in 9 distributions. In addition to that, in the case of the position intervals and the speed intervals, we also calculated the magnitudes as $||m|| = \sqrt{x^2 + y^2 + z^2}$, which gave us two additional distributions (a total of 11).

The way we turned each distribution into features, was by calculating 9 or 10 statistics for it: the mean, median, variance, standard deviation, kurtosis, skewness, min value, max value and range (sum was also included for the four distributions of intervals). With the 11 distributions we mention above, we made 7 basic groups of features for the global features category (Table I).

As can be seen in Table I, for the first basic group of features we used the statistics from the simple positional distributions of x, y and z, for the second we took the 4 corresponding intervals distributions, and for the third group we used the 4 speed distributions. Afterwards, we recombined all those features for four more basic groups. For group 4 we used the aforementioned features that refer to the x axis, then for group 5, 6 and 7 the ones that are about the y axis, z axis and magnitudes respectively.

The Distance Features: For the second category of features, inspired by the structure of the lock shape, we identified some points on it that are important to our perception.

Category	Groups	Features	Description
global	***Group 1***	27	Positional features from the distributions of x, y and z points of a drawing.
global	***Group 2***	40	Features from the distributions of the position intervals of the x, y, z and magnitude.
global	***Group 3***	36	Features from the distributions of the position intervals of the x, y, z and magnitude, combined with temporal information.
global	***Group 4***	28	All the features from groups 1, 2 and 3 that refer to the x axis.
global	***Group 5***	28	All the features from groups 1, 2 and 3 that refer to the y axis.
global	***Group 6***	28	All the features from groups 1, 2 and 3 that refer to the z axis.
global	***Group 7***	19	All the features from groups 2 and 3 that refer to magnitude.
global	*Group 8*	56	The fine performing groups of features regarding x and y (groups 4 and 5).
global	*Group 9*	103	All the unique, global features of groups 1, 2, 3, 4, 5, 6, 7, 8.
distances	***Group 10***	28	Distances that correspond to euclidean distance between the Points of Interest.
distances	***Group 11***	28	Distances that correspond to temporal distance between the Points of Interest.
distances	*Group 12*	56	All the distances of groups 10 and 11.
ratios	***Group 13***	378	Ratios that correspond to all possible ratios of the euclidean distances of Group 10.
ratios	***Group 14***	378	Ratios that correspond to all possible ratios of the temporal distances of Group 11.
ratios	*Group 15*	756	All ratios of groups 13 and 14.
ratios	*Group 16*	30	The best ratios of group 13 as found by a feature importance assessment based on multi-class classification with Random Forest.
ratios	*Group 17*	30	The best ratios of group 14 as found by a feature importance assessment based on multi-class classification with Random Forest.
ratios	*Group 18*	60	All the best ratios of groups 16 and 17.
mix	*Group 19*	83	The positional features of group 1 together with the distances of group 12.
mix	*Group 20*	112	The x and y features of group 8 together with the distances of group 12.
mix	*Group 21*	87	The positional features of group 1 together with the best ratios of group 18.
mix	*Group 22*	116	The x and y features of group 8 together with the best ratios of group 18.
mix	*Group 23*	116	The distances of group 12 together with the best ratios of group 18.
mix	*Group 24*	143	The positional features of group 1 together with the distances of group 12 and the best ratios of group 18.
mix	*Group 25*	172	The x and y features of group 8 together with the distances of group 12 and the best ratios of group 18.
mix	*Group 26*	916	All the unique, features of all other groups.

TABLE I: Performance of the different feature groups as the mean AUC score over 16 participants.

These are points where the motion has to change, or points at the edge of the circle that far enough from each other and should have some symmetry. We identified 8 of those points that can be seen in the lock of Figures 3b and 4, as big colored dots (green, yellow and red).

To find those points we made an algorithm tailored to our case and our kind of data, and we manually checked it in every lock that it works, else we dropped the drawing. However, in the end, from the locks that were dropped, none of them was dropped because the algorithm failed, but rather because they had other issues like deformations. Also, regarding the algorithm, it was made of simple instructions like locating the highest point in the drawing, the left-most or right-most point in the circle and so on.

Moreover, the reason we have different color-coding in the pictures, and not just yellow to every PoI, is because the green signals where the drawing starts and the red where the drawing ends. In the ideal lock of Figure 1, the green and the red should be the same exact point but in our case they were treated as different, because in practice it is extremely rare that a user would close the lock perfectly. Having those points we can then construct $\binom{8}{2}$ euclidean distances and another $\binom{8}{2}$ time intervals, as two basic groups of features (each of the groups would then have 28 features).

The Ratio Features: Following that, two additional basic groups can be made by computing the ratios between those distances to extract even more features. That is $\binom{28}{2}$ features for each of the aforementioned distance groups, which is equal to 378 features for each case (both the euclidean and the temporal distances). One concern we had at that point was that such a feature space is very high dimensional, and that might cause problems into training the algorithms. For that reason we tried to reduce the space by finding the most effective ratio features according to a feature importance assessment of of a multi-class classification with a Random Forest [40].

As can be seen in Figure 6 where the results of the feature assessment are displayed, in both the euclidean and the temporal ratio cases the importance quickly falls off after the first 20% of the features. In the euclidean ratios case, features after 30 quickly drop below 0.006 importance or 0.004 in the temporal ratio case. For that reason we feel confident that using less features would not have a dramatic impact on the performance - since the impact of each one is so small.

Also, between many of these features there is high degree of correlation with each other and to a large extent they can predict one another - so probably not all of them are needed. However, one could say that due to the fact that we used multi-class classification, we are biased towards the features that perform best in our group and that these importances would not necessarily generalize with the broader population. So in a sense, selecting the best features here is kind of a subtle way of overfitting.

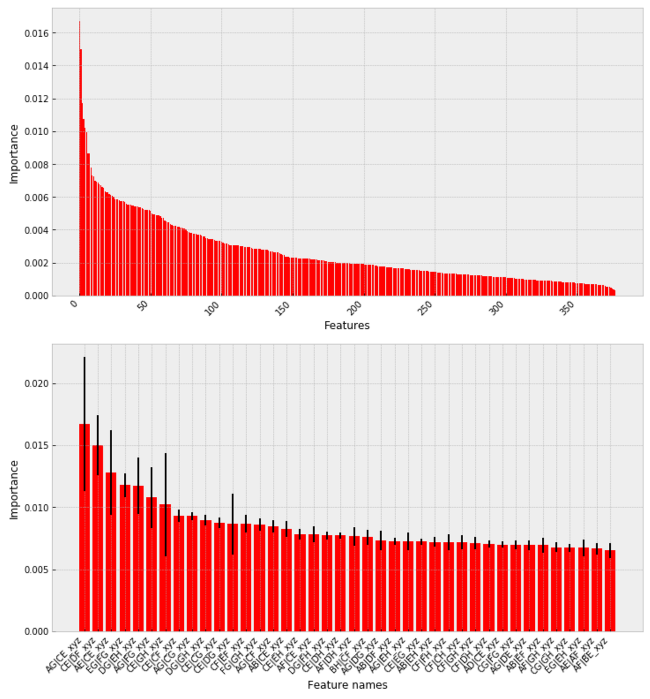
That criticism would be correct but given the fact that this is one-class classification that we are doing here, there might be a particular group of ratios a user is most consistent with, and if we had larger amounts of data and a way to identify these features (perhaps by training many models) we could legitimately use them for training. However, there is still no guarantee for how long they will remain as the best ratios if the behavioral patterns change as we have said early on. That being said, it would be interesting to see the performance of the smaller group, but take these results with a grain of salt.

Feature Construction Step in Practice: As has been said before, the entire amount of features that we made to describe a lock was 916. The outcome of the feature construction process was a csv file for each participant with rows equal to the number of the person's locks, and 916 columns for every feature. We then used these files for the next step which was classification. To try different combinations of features we sub-sampled these files.

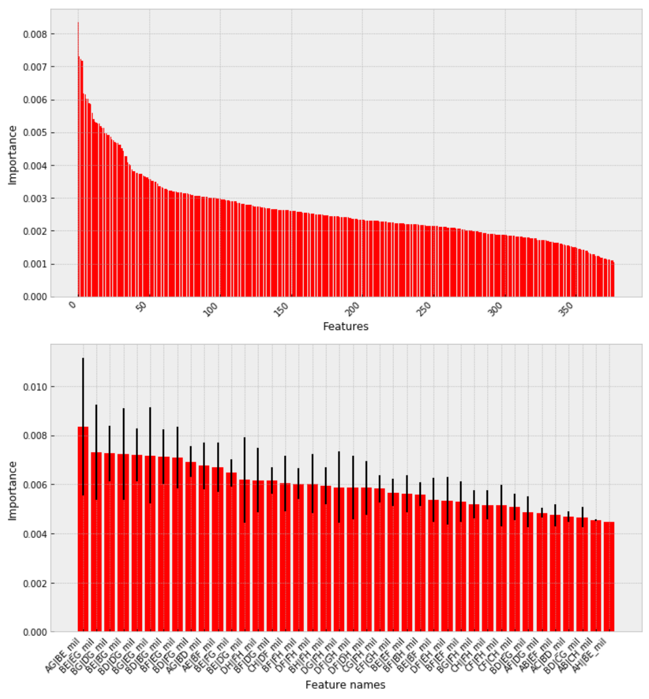
C. *Classification*

In this study it is two kinds of models that we used for classification: one-class SVM [17] and Isolation Forest [18]. For the one-class SVM we used the popular LIBSVM implementation [41] through the sklearn API [42]; for Isolation Forest we also used the implementation in sklearn [42]. For every participant and every group of features, we built these two kinds of models to be able to compare them. The way we evaluated them was mostly with Area Under the ROC Curve (AUC) [21] or with some error rate like EER, FRR, FAR.

The main difference between the two models is that the one-class SVM aims to find a decision boundary within which the observations of the correct class should be falling. In other words it is trying to find an area at the input space where the underlying probability density of the class lies [17]. On the other hand, instead of looking to profile the distribution of a class, Isolation Forest works by looking for anomalies into the way instances get isolated by a tree. It works under the premise, that the earlier an instance is isolated, by a binary tree, the more probable it is that it is an outlier [18], [19]. In that sense, the final anomaly score is computed by multiple tree estimators that produce shorter path lengths for when an instance is isolated [18], [19].



(a) Feature importances of euclidean ratios. *Top*: All 378 ratios. *Bottom*: 40 best ratios.



(b) Feature importances of temporal ratios. *Top*: All 378 ratios. *Bottom*: 40 best ratios.

Fig. 6: Feature importances for euclidean ratios (on the left) and temporal ratios (on the right).

Usually the two algorithms are mentioned in the context of novelty detection - mostly for one-class SVM - and anomaly or outlier detection - mostly for Isolation Forest [18]. The difference is that in the anomaly or outlier detection the train set is 'contaminated' with outliers/anomalies, but the terms sometimes are also used interchangeably [43]. Here our context is closer to novelty detection - and therefore one-class SVM - as we do have clean train sets. Isolation Forest however, according to the authors [18], can also perform well with train sets that do not have any anomalies.

To do classification, the data that we used for almost every case (except in the convergence rate analysis where we used data from the extra user) was 60 drawings from each of the 16 participants. In the cases where for some participants we had more than 60 samples available, we sampled randomly 60 drawings and dropped the others. The same sampling we used across all the different feature groups. Furthermore, because we also had to test with positive samples, and 60 is not too much data, we decided to use 5-fold cross-validation instead of splitting the dataset into something like 30-30 for train and test (or even further if we also wanted a validation set).

The way we implemented the cross-validation for each model was first by splitting the 60 positive samples into 5 folds of 48+12 locks and then we used the 48 samples to train and the 12 to test. In the case of negative samples we had plenty of drawings since we used all the locks from the remaining 15 participants in each fold (which was 900 drawings). The final step was to get the mean ROC curve out of the 5 curves and that represented the curve for a participant. The mean ROC curve was also calculated in cases where we wanted to see the performance across all participants.

In addition to that, before training any models, we tried to tune them in order to build them in the most efficient way. In order to do that we first specified the kind of parameters we needed to tune, and then we selected potential values. The combinations of those values represented a grid of possible parameters. Then by comparing models with parameter values across the grid, we could find the best performing set of parameters. We did this kind of grid search in two steps due to the large number of possible parameter sets. The first step was to make the search on a coarse set of parameters and then a finer one.

The way one set of parameters was selected was by comparing the mean of the FRR and FAR of the model trained with that set, with other models trained with different sets of parameters - we also used cross validation for finding the FRR and FAR of each model. One issue that we had with this kind of grid search was with which participant should we try to optimize, and with which feature set, because it seemed that each case could have slightly different results. In the end we made a long winded search across 10 feature sets (including all the 7 basic feature sets) and all participants and we selected the values that reported the lowest error across all combinations of settings (160 different models for each set of parameters).

More specifically in terms of which parameters we looked at, for one-class SVM we used the 'rbf' kernel, and we searched a grid of different values for the parameters nu and gamma. The best combination we found to be was 0.1 for nu and 0.001 for gamma. In the case of isolation forest the parameters that we investigated were the number of estimators (which is the number of trees), the contamination level and max features. The best parameters were found to be 350 estimators, 0.15 contamination and 1.0 max features.

In addition to cross-validation and parameter tuning, another aspect of the classification process was scaling the data. We did that before building any model. Prior to feeding the data to every model, the process was to train a scalar object with the data we would use to train the model itself, and then we transformed the positive and negative data accordingly. The scalar was trained to map the values for every feature between 0 and 1 based on the minimum and maximum value of that feature on the train set. The main reason we did that, was because not all our features were in the same metric (in some cases we had euclidean distances and in sometimes temporal).

Convergence Rate Analysis: When it comes to the methods that we used for the fourth research question which is the analysis of how fast the algorithms converge to the optimal performance, from the aforementioned techniques of cross-validation and scaling we only used scaling. We avoided cross-validation because we had plenty of data to train and test with, since we used the drawings of another participant from whom we collected 590 locks. Also, we always evaluated the algorithms under the optimal parameters as found from the preceding grid search.

The way we went by with the process, was by building models trained with sample counts from 1 until 400, and then finding their AUC scores. In every case, for testing we used 190 positive samples and 1088 negative samples which is all the samples we had in total for the other 16 participants. Also, to avoid the randomness of sampling too good or too bad locks - especially in the cases where sample count was low - we repeated the process multiple times for each sample count and we kept the mean AUC score.

The latter means that in a case where for example we are trying to evaluate models with 6 samples, we would randomly pick 6 train and 190 test samples (the negative samples were fixed to 1088) to evaluate a model, and then repeat this process to finally have a mean AUC score of several models that have been trained with the same number of samples. We run 20 and 4 repetitions for one-class SVM and Isolation Forest convergence rates respectively. The reason we were running less repetitions for Isolation Forest was because it was much more expensive to run. The result of less repetitions is for the convergence rate to be somewhat more coarse because the AUC estimation of each point is from a smaller sample of models.

V. Results

In this section we will try to present the results in a way that they, as directly as possible, answer back to the research

	Feature Groups														
	Global									Distances			all Ratios		
Participants	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Participant 1	0.88	0.86	0.86	0.90	0.88	0.75	0.82	0.91	0.90	0.96	0.86	0.97	0.98	0.78	0.97
Participant 2	0.98	0.85	0.81	0.94	0.89	0.85	0.79	0.96	0.95	0.98	0.86	0.98	0.97	0.70	0.95
Participant 3	0.95	0.89	0.87	0.89	0.96	0.78	0.81	0.97	0.95	0.97	0.91	0.98	0.95	0.82	0.93
Participant 4	0.98	0.95	0.89	0.93	0.97	0.81	0.92	0.98	0.98	0.98	0.96	**0.99**	**0.99**	0.95	**0.99**
Participant 5	0.97	0.96	0.95	0.97	0.97	0.90	0.93	0.98	0.97	0.96	0.91	**0.99**	0.83	0.64	0.79
Participant 6	0.98	0.88	0.80	0.95	0.93	0.84	0.78	0.96	0.95	0.95	0.35	0.91	0.90	0.63	0.85
Participant 7	0.94	0.82	0.77	0.88	0.81	0.82	0.70	0.89	0.88	0.83	0.92	0.94	**0.99**	0.92	**0.99**
Participant 8	0.79	0.76	0.73	0.74	0.72	0.76	0.68	0.75	0.78	0.81	0.86	0.88	**0.99**	0.81	**0.99**
Participant 9	0.97	0.95	0.93	0.97	0.93	0.89	0.93	0.97	0.98	0.95	0.89	0.98	0.96	0.81	0.95
Participant 10	**0.99**	0.95	0.93	0.96	0.92	0.96	0.94	0.97	0.97	0.96	0.94	0.97	0.96	0.93	0.98
Participant 11	0.91	0.87	0.86	0.79	0.90	0.84	0.81	0.91	0.92	0.96	0.81	0.96	0.93	0.76	0.92
Participant 12	0.98	0.96	0.95	0.96	0.95	0.97	0.93	0.97	0.98	0.94	0.90	0.98	0.89	0.89	0.95
Participant 13	0.93	0.78	0.81	0.80	0.87	0.79	0.83	0.88	0.91	0.91	0.91	0.97	0.91	0.75	0.90
Participant 14	0.96	0.90	0.92	0.92	0.94	0.94	0.91	0.95	0.96	0.95	0.88	0.97	0.95	0.72	0.92
Participant 15	0.98	0.91	0.88	0.91	0.91	0.95	0.87	0.95	0.97	0.96	0.89	0.97	0.93	0.79	0.93
Participant 16	0.98	0.94	0.91	0.95	0.98	0.64	0.94	**0.99**	0.98	0.96	0.91	**0.99**	0.94	0.86	0.92
Group Avg.	0.95	0.89	0.87	0.90	0.91	0.84	0.85	0.94	0.94	0.94	0.86	0.96	0.94	0.80	0.93

TABLE II: Performance of the different participants with one-class SVM in terms of AUC.

	Feature Groups											
	best Ratios			best Combinations							all	
Participants	16	17	18	19	20	21	22	23	24	25	26	**mean score**
Participant 1	0.97	0.85	**0.99**	0.96	0.95	**0.99**	0.98	**0.99**	**0.99**	**0.99**	0.98	0.920 (0.069)
Participant 2	0.95	0.70	0.96	**0.99**	0.98	**0.99**	0.98	**0.99**	**0.99**	**0.99**	0.98	0.922 (0.087)
Participant 3	0.95	0.74	0.91	0.98	0.98	0.96	0.97	0.97	0.98	0.98	0.96	0.924 (0.066)
Participant 4	**0.99**	0.91	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.965** (0.042)
Participant 5	0.89	0.68	0.89	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	0.98	0.927 (0.093)
Participant 6	0.89	0.68	0.90	0.95	0.95	0.98	0.97	0.96	0.97	0.97	0.93	0.877 (0.137)
Participant 7	**0.99**	0.90	**0.99**	0.95	0.95	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	0.916 (0.080)
Participant 8	**0.99**	0.83	**0.99**	0.90	0.89	**0.99**	**0.99**	**0.99**	**0.99**	0.98	0.98	**0.868** (0.107)
Participant 9	0.97	0.80	0.98	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	0.98	0.951 (0.051)
Participant 10	0.92	0.92	0.98	0.98	0.98	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	0.964 (0.024)
Participant 11	0.95	0.75	0.94	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.96	0.904 (0.069)
Participant 12	0.81	0.85	0.91	0.98	0.98	0.97	0.98	0.97	0.98	0.98	0.98	0.946 (0.044)
Participant 13	0.83	0.74	0.83	0.97	0.95	0.94	0.93	0.97	0.97	0.96	0.96	0.885 (0.072)
Participant 14	0.96	0.70	0.94	0.97	0.97	0.97	0.97	0.98	0.98	0.98	0.95	0.930 (0.069)
Participant 15	0.95	0.76	0.94	**0.99**	0.98	0.98	0.98	0.98	**0.99**	0.98	0.96	0.934 (0.057)
Participant 16	0.98	0.70	0.95	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	0.98	0.934 (0.085)
Group Avg.	0.94	0.78	0.94	0.97	0.97	0.98	0.98	0.98	0.99	0.98	0.97	0.923 (0.028)

TABLE II: Performance of the different participants with one-class SVM in terms of AUC. (Continuation)

questions we have set in section I. In that sense, in order to answer to the first question which asks for the extent to which we can verify a user with our system, we need to look at the models we made for each one of the participants, which is 52 models per person. Of those, 26 were one-class SVMs and 26 were Isolation forests. Also, each of those 26 in both algorithms is on a different group of features.

In tables II & III we can see the results for one-class SVM and Isolation Forest respectively. Each Table displays the AUC score of every one of the participants in each of the feature groups. Given that if building a system like that, we would build it with the best performing feature group, we

	Feature Groups														
	Global									Distances			all Ratios		
Participants	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Participant 1	0.92	0.81	0.83	0.88	0.88	0.73	0.82	0.90	0.89	0.96	0.85	0.97	0.95	0.78	0.93
Participant 2	0.93	0.81	0.81	0.91	0.85	0.81	0.78	0.92	0.90	0.97	0.89	0.96	0.95	0.73	0.88
Participant 3	0.93	0.88	0.87	0.87	0.93	0.78	0.82	0.95	0.93	0.97	0.89	0.97	0.93	0.77	0.88
Participant 4	0.96	0.88	0.82	0.88	0.93	0.82	0.87	0.94	0.92	0.97	0.94	0.98	**0.99**	0.95	**0.99**
Participant 5	0.97	0.96	0.96	0.96	0.95	0.95	0.94	0.97	0.97	0.97	0.92	**0.99**	0.83	0.75	0.84
Participant 6	0.97	0.86	0.83	0.95	0.94	0.79	0.82	0.96	0.95	0.96	0.78	0.93	0.87	0.65	0.81
Participant 7	0.95	0.79	0.76	0.87	0.84	0.80	0.75	0.89	0.86	0.85	0.92	0.93	0.98	0.91	0.97
Participant 8	0.83	0.76	0.77	0.79	0.80	0.70	0.70	0.82	0.81	0.90	0.84	0.90	0.98	0.79	0.94
Participant 9	0.96	0.87	0.90	0.96	0.93	0.82	0.88	0.97	0.96	0.94	0.91	0.98	0.94	0.83	0.93
Participant 10	0.93	0.92	0.90	0.91	0.91	0.93	0.90	0.95	0.95	0.95	0.89	0.95	0.95	0.91	0.97
Participant 11	0.89	0.84	0.84	0.82	0.88	0.83	0.80	0.91	0.90	0.96	0.79	0.93	0.92	0.72	0.90
Participant 12	0.97	0.96	0.96	0.97	0.95	0.95	0.93	0.98	0.97	0.94	0.87	0.96	0.87	0.83	0.89
Participant 13	0.92	0.78	0.75	0.83	0.87	0.75	0.77	0.88	0.87	0.93	0.89	0.96	0.92	0.76	0.88
Participant 14	0.93	0.92	0.94	0.89	0.93	0.93	0.92	0.95	0.96	0.95	0.91	0.97	0.94	0.67	0.87
Participant 15	0.98	0.89	0.84	0.88	0.89	0.94	0.83	0.95	0.96	0.97	0.90	0.97	0.95	0.85	0.96
Participant 16	0.97	0.92	0.91	0.94	0.96	0.83	0.92	0.97	0.95	0.98	0.91	0.98	0.94	0.83	0.92
Group Avg.	0.94	0.87	0.86	0.89	0.90	0.84	0.84	0.93	0.92	0.95	0.88	0.96	0.93	0.80	0.91

TABLE III: Performance of the different participants with Isolation Forest in terms of AUC.

	Feature Groups											
	best Ratios			best Combinations							all	
Participants	16	17	18	19	20	21	22	23	24	25	26	**mean score**
Participant 1	0.96	0.84	0.95	0.97	0.96	0.97	0.96	**0.99**	**0.99**	0.98	0.95	0.909 (0.070)
Participant 2	0.95	0.71	0.92	0.97	0.97	0.94	0.94	0.96	0.97	0.97	0.91	0.897 (0.076)
Participant 3	0.92	0.71	0.87	0.97	0.98	0.91	0.94	0.95	0.96	0.98	0.94	0.904 (0.068)
Participant 4	**0.99**	0.92	0.98	0.98	0.97	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.947** (0.052)
Participant 5	0.90	0.76	0.88	**0.99**	0.98	0.98	0.98	**0.99**	**0.99**	**0.99**	0.95	0.935 (0.067)
Participant 6	0.90	0.67	0.88	0.96	0.95	0.96	0.97	0.92	0.95	0.95	0.88	0.887 (0.088)
Participant 7	**0.99**	0.86	0.97	0.95	0.95	**0.99**	0.98	0.98	**0.99**	0.98	0.97	0.911 (0.075)
Participant 8	**0.99**	0.82	0.94	0.93	0.91	0.96	0.95	0.96	0.95	0.95	0.96	**0.871** (0.087)
Participant 9	0.97	0.80	0.94	**0.99**	**0.99**	0.98	0.98	**0.99**	**0.99**	**0.99**	0.97	0.937 (0.055)
Participant 10	0.92	0.93	0.97	0.95	0.97	0.98	0.98	0.98	0.98	**0.99**	0.98	0.944 (0.030)
Participant 11	0.95	0.71	0.90	0.94	0.94	0.93	0.93	0.93	0.94	0.95	0.92	0.884 (0.068)
Participant 12	0.83	0.79	0.87	0.97	0.98	0.97	0.98	0.96	0.98	0.98	0.96	0.934 (0.055)
Participant 13	0.88	0.72	0.84	0.97	0.95	0.92	0.92	0.95	0.96	0.96	0.92	0.875 (0.075)
Participant 14	0.95	0.61	0.90	0.98	0.98	0.94	0.96	0.97	0.97	0.98	0.93	0.917 (0.085)
Participant 15	0.95	0.83	0.94	0.98	0.98	0.98	0.97	0.98	**0.99**	**0.99**	0.98	0.936 (0.052)
Participant 16	0.98	0.75	0.95	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	0.97	0.943 (0.058)
Group Avg.	0.94	0.78	0.92	0.97	0.97	0.96	0.96	0.97	0.98	0.98	0.95	0.914 (0.025)

TABLE III: Performance of the different participants with Isolation Forest in terms of AUC. (Continuation)

should look at the performance per participant on the best group only. According to Table IV the best groups are group 24 for one-class SVMs and group 25 for Isolation Forest.

In the case of one-class SVM that is displayed in Table II, we can see that for group 24, 10 out of 16 users have an AUC score of 0.99 and the remaining 6 users score 0.97 or higher. Not much different is the case of Isolation Forest where for group 25, 6 out of the 16 users have an AUC score of 0.99, and the remaining 10 participants score at least 0.95.

The mean AUC score across all participants for group 24 with one-class SVM is 0.986, and for group 25 with Isolation Forest is 0.977. The respective Equal Error Rates (EERs) for

	Basic Feature Sets											mean AUC scores (std inc.)	
Feature Groups	positional	intervals (pos)	speed	x axis features	y axis features	z axis features	magnitude features	euc distances	temp distances	euc ratios	temp ratios	**oc-SVM**	**IF**
Group 1	+											0.948 (0.051)	0.939 (0.037)
Group 2		+										0.889 (0.061)	0.865 (0.061)
Group 3			+									0.867 (0.064)	0.856 (0.064)
Group 4				+								0.904 (0.066)	0.894 (0.051)
Group 5					+							0.908 (0.066)	0.903 (0.045)
Group 6						+						0.843 (0.088)	0.835 (0.078)
Group 7							+					0.850 (0.083)	0.841 (0.069)
Group 8				+	+							0.936 (0.058)	0.931 (0.042)
Group 9	+	+	+	+	+	+	+					0.939 (0.052)	0.921 (0.045)
Group 10								+				0.940 (0.048)	0.946 (0.031)
Group 11									+			0.861 (0.137)	0.882 (0.044)
Group 12								+	+			0.963 (0.029)	0.959 (0.023)
Group 13										+		0.942 (0.043)	0.932 (0.042)
Group 14											+	0.798 (0.094)	0.796 (0.083)
Group 15										+	+	0.933 (0.052)	0.910 (0.049)
Group 16										+		0.938 (0.054)	0.939 (0.044)
Group 17											+	0.782 (0.081)	0.778 (0.085)
Group 18										+	+	0.943 (0.044)	0.920 (0.041)
Group 19	+							+	+			0.972 (0.022)	0.968 (0.017)
Group 20				+	+			+	+			0.966 (0.025)	0.965 (0.021)
Group 21	+									+	+	0.982 (0.015)	0.963 (0.025)
Group 22				+	+					+	+	0.979 (0.016)	0.963 (0.021)
Group 23								+	+	+	+	0.983 (0.011)	0.969 (0.021)
Group 24	+							+	+	+	+	**0.986** (0.008)	0.975 (0.016)
Group 25				+	+			+	+	+	+	0.983 (0.011)	**0.977** (0.015)
Group 26	+	+	+	+	+	+	+	+	+	+	+	0.972 (0.017)	0.949 (0.029)
											Mean scores:	0.923 (0.057)	0.914 (0.055)

TABLE IV: Performance of the different feature groups as the mean AUC score over 16 participants.

the two cases are 4.7% and 7.2%. This, except for question one, also answers the research questions two and three to some extent. That is because the two best groups of features are identified as 24 and 25, and because one-class SVM seems to have the edge over Isolation Forest.

However that is not the best we can do. Earlier we have said that we run our one-class SVM models with the 'rbf' kernel, 0.1 nu and 0.001 gamma, which was a result of a tuning process across many groups of features and all users. But given that we know the best performing feature group we could also tune the models specifically with that group. When we did that for the one-class SVM specifically for group 24, we found out that with kernel 'rbf', and the nu, and gamma parameters set to 0.001 and 0.15 respectively, the AUC score could improve to 0.988 and the EER to 4.05%.

Moreover, it also needs to be mentioned here that when we attempted to do tuning specifically for with Isolation Forest and group 25, we did not manage to get a better result than the one we had with the previous set of parameters. Part of the reason is that it was maybe harder to find these parameters, but it is also the case that the implementation of Isolation Forest that we used was more expensive to run in terms of time complexity, and we could not run it with as many parameter sets. One additional thing as far as tuning goes for both algorithms, is that we could also tune the models specifically for each user, but according to the logs of the tunning process, there was hardly any variation between the AUC scores of the same users with different sets of parameters.

Another way to answer research question 1, would be to look at each participant under his best performing group. That would be interesting in case we were building a system while knowing the best performing feature group for each person. However that means that we would have to make a feature group analysis for every user we enroll which might not be as efficient. It would be interesting though to see the extent of the improvement, if there is any.

For that case we can look at the best AUC scores for a participant across all groups of features. In Tables II and III we can see that 10 out of 16 users have at least one group of features with AUC score of 0.99. Also, for the other 6 users there is at least a group of features with an AUC score of 0.97. Not much different is the case of Isolation Forest where 9 out of the 16 participants have at least a group of features with 0.99 AUC score, and the remaining 7 participants have at least a group of features with AUC score of 0.96. In comparison with the case where we selected the same feature group for everyone, it seems that by selecting someone's best group separately, the performance increase is very small for either algorithm.

Furthermore, among our 16 participants there does not seem to be any individual that performs better in the one category of features or the other in isolation although there are a few exceptions. There are some isolated incidents like participant 8 who performs poorly with distances but performs excellently with ratios - which is against the norm. However all these differences are ameliorated when it comes to the mixed feature groups where all participants seem to perform best.

In addition to that, it would be interesting to see the locks of the most and least successful participants and speculate for what might make a participant successful or not. In that sense, probably the most successful participant across all categories of features is participant 4, and the least successful is participant 8. By looking at a hand-picked sample - where we aimed for a representative pick - of each one's drawings in Figure 4, we can tell that the displayed locks of participant 4 clearly seem more consistent than the ones of participant 8.

Moreover, specifically for participant 8, it seems that distances are not such a good group, but ratios are; so among his drawing inconsistency, he is keeping some things stable. Opposite to that, participant 5 (whose locks can be seen in Figure 4b), is the worst performer when it comes to ratios and among the best in the case of distances, while most people perform better in distances than ratios. These are interesting cases that show that not all the participants will perform best at the same group of features. On the other hand, they are very isolated cases in a small pool of participants where it clearly seems that most people perform well at specific sets of features.

In terms of the second research question, as we have already mentioned while discussing question one, the best feature groups seem to be group 24 for one-class SVM and group 25 for Isolation Forest. The results across all groups as well as the components of each group are summarized in Table IV - as we have said most of the feature groups are combinations of some basic feature groups. As discussed earlier, the description of the features of each group are summarized in Table I.

In that sense, looking at the results of Table IV, which are the mean scores across participants, we can see first of all that the global features are lagging behind the others, except when it comes to simple positional features (group 1), the combination of xs and ys (group 8), and all the global features together (group 9). Notably the positional features of group 1, perform better than when we add the speed and interval information for group 9.

Moving to the distance features, the euclidean distances (group 10) seem to do quite well. Also, the euclidean distances along with the temporal distances of group 11, seem to complement each other well with a performance of around 0.96 (in both models) for group 12. Ratios in groups 13-18 perform also ok but slightly worse than the distances - by 0.02 or more. Also, it seems that the euclidean and temporal ratios, do not combine well when it comes to the very high dimensional groups, as adding the 378 euclidean rations with the 378 temporal ones (group 15) performs worse than the euclidean alone (group 13). Even in the lower dimension ratio groups (the ones selected by a feature assessment from multi-class classification as the 30 best), adding the temporal ratios does not account for a significant improvement in performance.

The best performance all across the features board though, comes from the combinations of the best performing groups that have been tried and can be seen at the Table IV as groups 19, 20, 21, 22, 23, 24, 25 and 26. For these groups we combined group 1 (x, y, z positional distributions), group 8 (features for x and y axis), group 12 (the distances) and group 18 (selected best ratios) into the possible combinations, and the best seemed to be the cases of using groups 1, 12 and 18 for one-class SVM and 8, 12 and 18 for Isolation Forest with respective AUC scores of 0.986 and 0.977 as has already been mentioned.

The main takeaways from the feature group analysis is that the simple positional features, the distances and ratios are very good features for this problem, and when combined they are complementary to each other. In addition to that it seems that temporal distances are worse predictors than euclidean ones, but they can be complementary for each other. That is not the case though when it comes to ratios. Also it seems that sometimes there are feature groups that perform much better or worse with one or two participants in particular, but they are not as good with the others. That is not a very frequent occasion though.

By all the discussion for the two questions so far, it seems that research question three has been partially answered as well. In general it seems that across all models, the one-class SVM performs very slightly better and as we will discuss later on, the algorithms have no real difference in terms of converging to their best score with more training samples. Regarding the performance though, in order to give a more comprehensive answer we need to look at the performance of the two algorithms across the participants and across the different feature groups. That way we could see if an algorithm performs better than the other in let's say high dimensional feature groups, or in a specific participant for some reason.

However, that doesn't seem to be the case when we look at the mean performance across participants in Tables II and

III. More specifically, we can see that one-class SVM has the edge on 11 of them and the Isolation Forest performs better only in 5 cases. The differences when one-class SVM is better are around 0.015 and for Isolation Forest they are around 0.006. This small superiority of one-class SVM can also be seen by the mean of the mean scores of the participants in the two tables, where one-class SVM scores 0.923 and Isolation Forest 0.914.

When it comes to the performance of the two algorithms across feature groups, the situation as we can see in Table IV is quite similar. From the 26 different feature groups perspective, Isolation Forest has a very slight edge only in 3 of them (for around 0.009), while one-class SVM performs better for a very small margin in the remaining 23 groups (for a mean margin of 0.011). That is also reflected if we take the mean scores (of the mean scores for each group), where one-class SVM seems to score 0.923 and Isolation Forest 0.914. That is the same score with the participants case, with the only difference that it has a somewhat bigger standard deviation.

Despite the fact that all across the board one-class SVM seems to be better, all the differences are so small that probably a more in-depth parameter tuning could wipe out. In that sense, and given both the algorithms ability to converge fast, we would say that while one-class SVM seems very slightly better, there is no clear winner as far as performance goes from the two algorithms.

Finally, for the fourth and final research question, we will be looking at the evaluations of the models we built for an additional participant that is not among the 16 - as we have described in previous sections. With the drawings of that user, we looked for the convergence rate of the AUC score across all 26 groups of features as we have discussed earlier. In Figure 7 there are the curves for the AUC convergence of the one-class SVM and Isolation Forest for 6 of those groups that are indicative for the remaining groups. Because of the fact though that this is another user, there might be small differences with the results we have discussed so far that are computed for across the participants. For example in Figure 7a we can see that for him the Isolation Forest performs better than the one-class SVM which is not the case for the other participants.

The first and most impressing thing that we observe from those plots is with how few samples the algorithms converge. In most cases the algorithm needs no more than 5 samples to converge, and in the worst cases no more than 10 to 15 are needed. In general, between the two kinds of models we do not see any big difference in terms of one converging faster than the other although the one-class SVM seems to have a small edge by converging in most cases with about 5 samples, whereas the Isolation Forest needs around 10.

One peculiar thing we do notice though, is that in the case of the Isolation Forest, the convergence is very coarse at the beginning and it always dips down in the fourth sample. Finally, another thing that seems remarkable is that the extremely high dimensionality does not prolong the conversion rate whatsoever. This becomes evident in Figures 7a & 7f that refer to groups with 27 and 916 features respectively and they both converge with about 5-10 samples.

VI. Discussion and Future Work

In this section we will discuss the contribution of this research, potential areas for future work and limitations we came across for this research. Our goal regarding the discussion is mainly to discuss some thoughts and expectations and to put our findings into a perspective relative to other work close to this field.

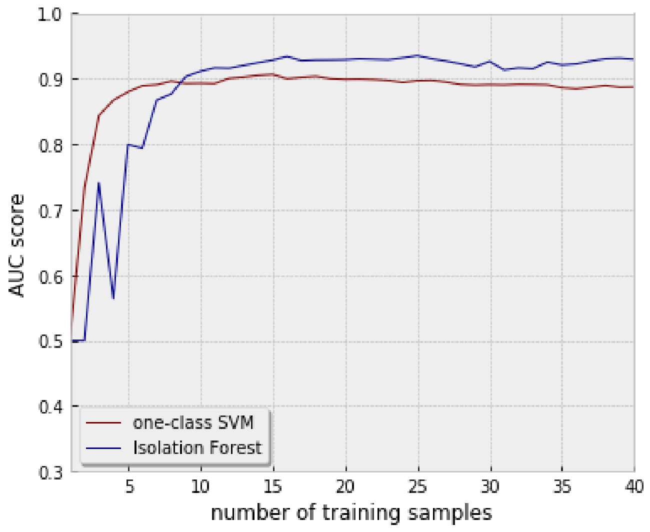
A. Contribution of this Research

The contribution of this research is to show that a simple air-drawn symbol could be considered as a behavioral biometric with comparable results. For that reason we set out our research questions the way we did in section I. In agreement with those questions, the displayed results of the previous section report that it is possible to build a verification system for a simple symbol such as the lock of Figure 1 with an EER of approximately 4.05%, by using a certain set of features and an one-class SVM model. It also seems that no more than 5 samples are needed to get near optimal performance of such a model and that, in general, one-class SVM outperforms Isolation Forest by a little.

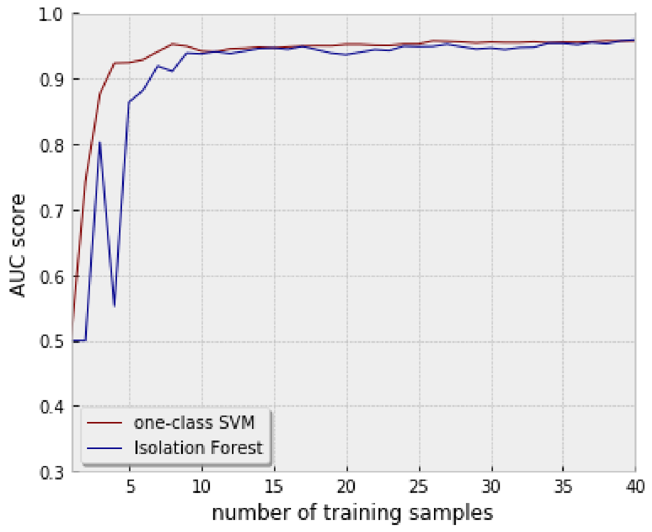
As have been already mentioned, the 4.05% EER is achieved with parameter tuning specifically for the best performing group of features across all participants (group 24 of Table I). It would also be possible to tune the model parameters separately for each participant. That could also be realistic for a practical application, but we would first need to have enough data for that participant. However, by observing the logs of the tuning for the features of group 24 across all participants, it seems that the performance improvement of tuning for a specific participant would be extremely small, if any.

As far as features go, from the three categories of features that we used here (global, distances and ratios), the best kind of features appeared to be the distances. These distances were between points on the lock shape that are important to our perception (the large colored points in Figures 3b and 4). However, given the fact that behavioral biometrics suffer from consistency issues as we mention earlier, one could say that the size of the drawing would be the first thing someone is inconsistent with (e.g. if the user is tired he might make a smaller drawing). Ratios (of those distances) on the other hand, could be a better kind of feature since they might be more robust to behavioral changes. In any case we did not take samples over many sessions with each participant, which would probably increase the intra person variability and could probably bring this issue to the surface.

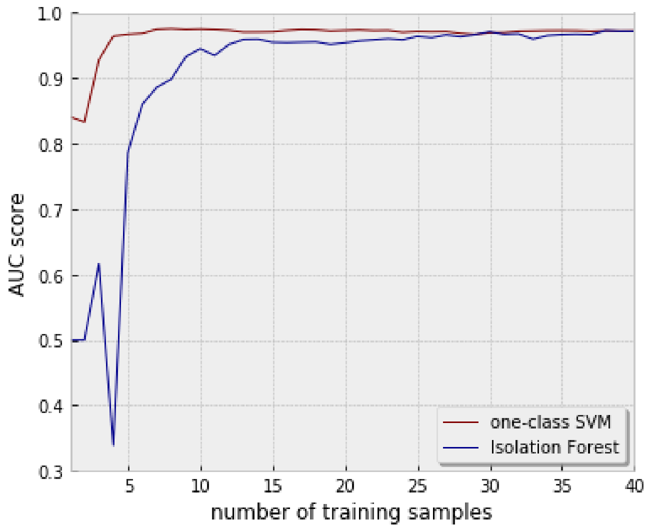
Finally, we think that the results presented here show that a simple air-drawing such the one we experiment with, could be thought of as a suitable biometric that can potentially yield similar results to the ones seen in online signature verification and keyboard or mouse dynamics. In Table V we summarize the results of some of the leading studies in these Behavioral



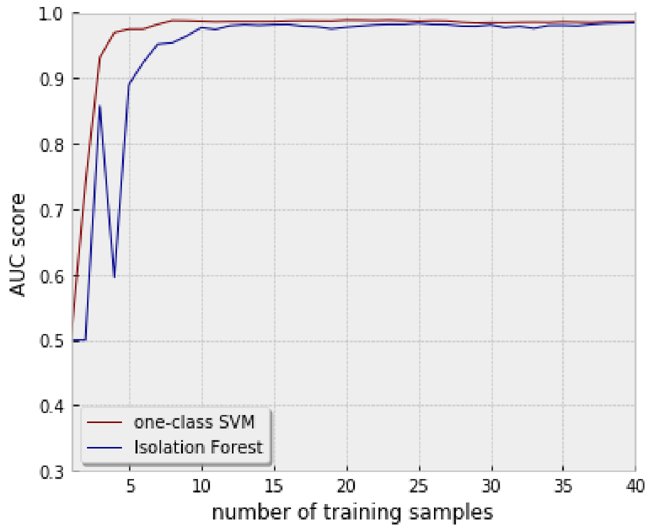
(a) Trained with 27 features of group 1.



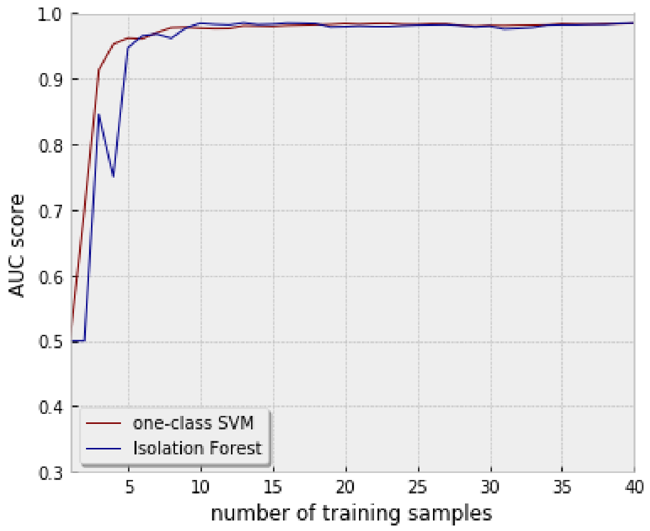
(b) Trained with 56 features of group 12.



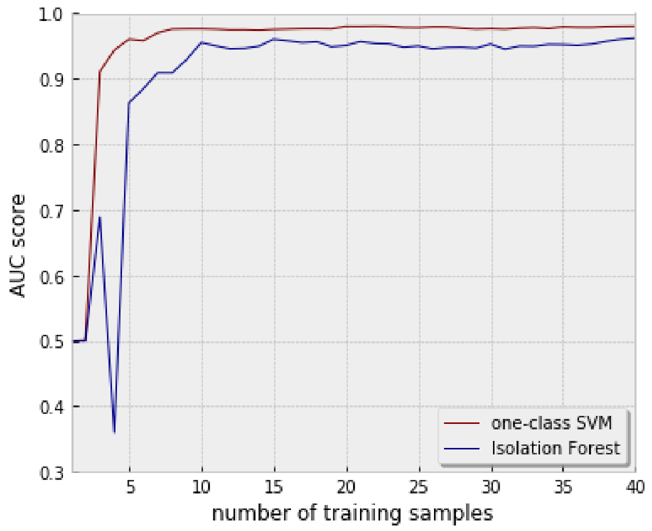
(c) Trained with 60 features of group 18.



(d) Trained with 143 features of group 24.



(e) Trained with 172 features of group 25.



(f) Trained with 916 features of group 26 (all features).

Fig. 7: Convergence rate of one-class SVM and Isolation Forest.

	Error Rate	Score
Signatures in 2-d		
SVC2004 [30]	2.84% (EER)	-
Kholmatov, Alisher, and Berrin Yanikoglu (SVC2004 winners) [31]	1.64 (FRR)% & 1.28% (FAR)	-
Nakanishi, Isao, et al. [32]	4% (EER)	-
Nyssen, Edgard, Hichen Sahli, and Kui Zhang [33]	5.8% (FRR) & 0% (FAR)	-
Ortega-Garcia, Javier, et al. [34]	0.35% (EER)	-
Kashi, Ramanujan S., et al. [35]	2.5% (EER)	-
Air-drawn signatures		
Tian, Jing, et al.[3]	-	100% (precision) & 70% (recall)
Behera, Santosh Kumar, Debi Prosad Dogra, and Partha Pratim Roy[4]	-	95.5% (accuracy)
Bailador, Gonzalo, et al. [22]	4.58% (EER)	-
Air-drawn Characters		
Our approach	**4.05% (EER)**	**-**
Keyboard Dynamics		
Hwang, Seong-seob, Hyoung-joo Lee, and Sungzoon Cho [16]	1% (EER)	-
Mouse Dynamics		
Shen, Chao, et al. [1]	7.96% (FRR) & 8.74% (FAR), in 11.8 seconds*	-

*The FRR and FAR in [1] can be greatly reduced even below 1%, with more samples and increased authentication time up to 10 mins.

TABLE V: Performance of leading studies in different areas of Behavioral Biometrics.

Biometric areas. We have also discussed those studies in section II.

More specifically, it seems that our approach is in the same ballpark with the air-drawn signatures, but falls short when it comes to the performances in 2-d signatures or keyboard dynamics. The referenced mouse dynamics approach seems to do worse than our approach, but given an increased number of samples and more time for verification their results improve significantly (even below 1% but that comes with almost 10 minutes of authentication) [1]. However, it should also be noted that these areas are much more well researched and plenty of techniques have been applied before the error rates to be dropped so low. As a result, we think that given more data or some other improvements, such results are not far off for simple air-drawn symbols as well.

B. Suggestions for Future Work

As far as directions for future work, the possibilities are numerous. That is because air-drawing is not very much thought of as a behavioral biometric that could be used practically, given that there is no other application for it. However, about the lock symbol that we have used, the most obvious suggestion for further research would be to investigate more techniques and features that would increase the performance from 4.05% EER. That could be features that exploit symmetries, function based techniques and much more things that have been tried for signature verification and other neighboring fields.

As far as more features is concerned, that would probably require more kinds of data. Two obvious kinds of data are acceleration and inclination which would probably require the data collection to be done with IMUs. However, that introduces some other kinds of problems such as possible movement restrictions due to the fact that the accelerometer would have to be mounted on the nail or the issue of drift when turning the acceleration into position that we have discussed earlier. In addition to that, more features could be extracted even from the positional data that we had. For example since the lock shape is a circle on top of a triangle, we could use its angles, centroids and other things.

Another suggestion would be to see how starting drawing from anywhere could impact the results. Such a thing might improve the results because there is an additional choice from each participant which adds variability, but it might add to drawing inconsistency over time, since the user has more things to deal with, that require making extra choices that might influence the behavior. Except from choice though, other thing that might influence change of behavior in the long term could be posture changes or the different psychological states whose impact is also worth exploring.

Furthermore, in terms of adding complexity, it would be interesting to see if the results improve in the case where the drawing is not made by the finger but with a pen-like tool or what would be the impact of drawing a different, more complex, symbol than the lock. As far as the tool goes, it would probably have a positive impact on performance since there is an additional joint that makes the movement more complex. Similarly, a more complex symbol (e.g. a G-clef instead of a lock) would most likely increase the drawing variability between the participants, and therefore it would also have a positive effect on performance. In fact this hypothesized correlation between shape complexity and performance would be interesting to be explored further.

In addition to that, it would also be interesting to see if, for a given participant, there are other participants that are consistently being miss-interpreted. Such a 'relatedness' between a group of participants could probably reveal families of drawing styles. However, probably a lot of data would be needed for such a thing. On a final note, another hard but interesting task would be to make a more generic algorithm that could find the Points of Interest by itself in any given shape. That seems like a difficult problem for the computer to solve though, given that these points are important to human perception.

C. Limitations

The two main limitations of this study come from the fact that generally in behavioral biometrics there is the issue of inconsistency over long periods of time and also because of the fact that our data comes from a lab environment which might be different to the real world. The latter is usually the case in such studies. Other than that, there were also some issues that might have affected the results of the research and they have to do with the data collecting device.

As far the drawing inconsistency that we also discussed in the introduction section, it is an issue intrinsic to behavioral biometrics. Some studies like [22] try to collect data from the same participants over prolonged periods and make a permanence analysis. However, in our case due to time and budget limitations we could not do that. Our efforts in inducing this intra-person variability were confined in reseting the process after each drawing for each participant but always within the same session.

In that sense our results are useful to the extent that each user is consistent with the drawings, which might not always be the case. In spite of that though, it seems that this effect of consistency is not very well studied, and that there are some evidence that show both stability and instability of behavior. That is the case for example in [22], where according to the permanence analysis the makers of air-drawn signatures are split into two groups, one that tends to keep a stable pattern and another that does not.

In addition to that, for our research it needs to be mentioned that the data was collected in a controlled environment which is different than the real world where there might be all kinds of causes to influence the behavior. Environmental factors such are the distance from the object a user is casting on, the presence of others and other things might have an impact on the behavior that we could not see in the lab. In addition to that psychological factors such as mood or the state of mind of the participant might also affect the behavior. This might also be interconnected with the consistency issue we raised earlier.

Finally, there were two issues with the Kinect device that might had some small impact on the results. First of all the Kinect followed the closest point and not the finger, which meant that the users had to have the finger prolonged and that put a small constraint in their movement in the sense that they had to have the rest of the fingers in a fist and were instructed to avoid some extreme angles with the elbow. Moreover the Kinect was a little noisy, especially in terms of the z axis which probably had some effect on the results.

VII. Conclusions

Air-drawing as a behavioral biometric has a lot of potential the smarter items around us are getting. In this research, we studied a simple lock symbol and the possibility to verify the person who did its drawing. We have also proposed that a system like that could potentially be used from a person to open a door. Our results report an EER of 4.05% that was achieved by an one-class SVM across a sample of 16 participants. The AUC score of the mean ROC curve across all participants from which the EER was calculated was 0.988. This seems quite promising, however more evidence needs to support such systems before they come close to becoming reality. More important than that though, would be the existence of connected items that allow such messaging.

Also, depending of the task, the demands for security can be quite different. A very secure system would be needed in order to be trusted with giving access to a private space through a door, but that is not so much the case when it comes to communicating to a lamp to open, or to ask the coffee machine to make coffee. Also, the utility of such a system could be extended to not only give a command, but to also make sure that the other device, e.g. the coffee maker, knows who made the symbol and therefore probably know what kind of coffee to make. That can potentially be extended further into other things though, such as items to be able to understand someone's mood, or whether that person is in a hurry by the way of the drawings.

In our approach we first collected the data with an application that we made for this purpose and then we built a user-dependent verification system that we used with 16 users. More specifically, from the data of each user we created simple features with which we trained models for each participant. In that sense, we did not make use of template related techniques that is often the case for behavioral biometrics. In addition to that, along with testing the models for their performance across users, we also tested various groups of features to find the most important ones. The features we used were made either by the raw distributions of the points of the drawing trace, or by the distances and ratios that we could produce by some Points of Interest we identified for that purpose. These Points of Interest are points on the drawing that are important for our perception of the lock.

As this study shows, features that have to do with those Points of Interest that are important to our perception of the symbol, can yield good results. Also, it seems that there is small differences between one-class SVMs and Isolation Forests, with the SVM to have a slight edge in most cases. In addition to that, both those algorithms seem to need incredibly few samples to learn a pattern of such a drawing, even when it comes to very high dimensional features (up to 5 samples for the one-class SVM and 10 for IF). Finally, it seems that verification by the drawing of a simple symbol, with a mean EER of 4.05% across participants, is in par with other kinds of biometrics such as online signature verification or mouse and keyboard dynamics.

References

[1] Shen, Chao, et al. ”User authentication through mouse dynamics.” IEEE Transactions on Information Forensics and Security 8.1 (2013): 16-30.

[2] Plamondon, Rjean, Giuseppe Pirlo, and Donato Impedovo. ”Online signature verification.” Handbook of Document Image Processing and Recognition. Springer London, 2014. 917-947.

[3] Tian, Jing, et al. ”KinWrite: Handwriting-Based Authentication Using Kinect.” NDSS. 2013.

[4] Behera, Santosh Kumar, Debi Prosad Dogra, and Partha Pratim Roy. ”Analysis of 3D signatures recorded using leap motion sensor.” Multimedia Tools and Applications (2017): 1-26.

[5] Lachat, E., et al. ”First experiences with Kinect v2 sensor for close range 3D modelling.” The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 40.5 (2015): 93.

[6] Yang, Lin, et al. ”Evaluating and improving the depth accuracy of Kinect for Windows v2.” IEEE Sensors Journal 15.8 (2015): 4275-4285.

[7] Reas, Casey, and Ben Fry. ”Processing: programming for the media arts.” AI & SOCIETY 20.4 (2006): 526-538.

[8] ”OpenKinect Repository” GitHub. 21 May 2018. https://github.com/shiffman/OpenKinect-for-Processing

[9] Zafrulla, Zahoor, et al. ”American sign language recognition with the kinect.” Proceedings of the 13th international conference on multimodal interfaces. ACM, 2011.

[10] Thong, Y. K., et al. ”Numerical double integration of acceleration measurements in noise.” Measurement 36.1 (2004): 73-92.

[11] Roetenberg, Daniel, Per J. Slycke, and Peter H. Veltink. ”Ambulatory position and orientation tracking fusing magnetic and inertial sensing.” IEEE Transactions on Biomedical Engineering 54.5 (2007): 883-890.

[12] LeCun, Yann, et al. ”Gradient-based learning applied to document recognition.” Proceedings of the IEEE 86.11 (1998): 2278-2324.

[13] Vikram, Sharad, Lei Li, and Stuart Russell. ”Handwriting and Gestures in the Air, Recognizing on the Fly.” Proceedings of the CHI. Vol. 13. 2013.

[14] Srihari, Sargur N., Aihua Xu, and Meenakshi K. Kalera. ”Learning strategies and classification methods for off-line signature verification.” Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on. IEEE, 2004.

[15] Eskander, George S., Robert Sabourin, and Eric Granger. ”Hybrid writer-independentwriter-dependent offline signature verification system.” IET biometrics 2.4 (2013): 169-181.

[16] Hwang, Seong-seob, Hyoung-joo Lee, and Sungzoon Cho. ”Improving authentication accuracy using artificial rhythms and cues for keystroke dynamics-based authentication.” Expert Systems with Applications 36.7 (2009): 10649-10656.

[17] Schlkopf, Bernhard, et al. ”Estimating the support of a high-dimensional distribution.” Neural computation 13.7 (2001): 1443-1471.

[18] Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou. ”Isolation forest.” Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on. IEEE, 2008.

[19] Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou. ”Isolation-based anomaly detection.” ACM Transactions on Knowledge Discovery from Data (TKDD) 6.1 (2012): 3.

[20] Impedovo, Donato, Giuseppe Pirlo, and Rejean Plamondon. ”Handwritten signature verification: New advancements and open issues.” Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on. IEEE, 2012.

[21] Fawcett, Tom. ”An introduction to ROC analysis.” Pattern recognition letters 27.8 (2006): 861-874.

[22] Bailador, Gonzalo, et al. ”Analysis of pattern recognition techniques for in-air signature biometrics.” Pattern Recognition 44.10 (2011): 2468-2478.

[23] Matsuo, Kenji, et al. ”Arm swing identification method with template update for long term stability.” International Conference on Biometrics. Springer, Berlin, Heidelberg, 2007.

[24] Jain, Anil K., Arun Ross, and Salil Prabhakar. ”An introduction to biometric recognition.” IEEE Transactions on circuits and systems for video technology 14.1 (2004): 4-20.

[25] Siradjuddin, Indrazno, et al. ”A position based visual tracking system for a 7 DOF robot manipulator using a Kinect camera.” Neural Networks (IJCNN), The 2012 International Joint Conference on. IEEE, 2012.

[26] Plamondon, Rejean, and Guy Lorette. ”Automatic signature verification and writer identificationthe state of the art.” Pattern recognition 22.2 (1989): 107-131.

[27] Leclerc, Franck, and Rejean Plamondon. ”Automatic signature verification: The state of the art19891993.” International journal of pattern recognition and artificial intelligence 8.03 (1994): 643-660.

[28] Plamondon, Rjean, and Sargur N. Srihari. ”Online and off-line handwriting recognition: a comprehensive survey.” IEEE Transactions on pattern analysis and machine intelligence 22.1 (2000): 63-84.

[29] Impedovo, Donato, and Giuseppe Pirlo. ”Automatic signature verification: The state of the art.” IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 38.5 (2008): 609-635.

[30] Yeung, Dit-Yan, et al. ”SVC2004: First international signature verification competition.” Biometric Authentication. Springer, Berlin, Heidelberg, 2004. 16-22.

[31] Kholmatov, Alisher, and Berrin Yanikoglu. ”Identity authentication using improved online signature verification method.” Pattern recognition letters 26.15 (2005): 2400-2408.

[32] Nakanishi, Isao, et al. ”On-line signature verification based on discrete wavelet domain adaptive signal processing.” Biometric Authentication. Springer, Berlin, Heidelberg, 2004. 584-591.

[33] Nyssen, Edgard, Hichen Sahli, and Kui Zhang. ”A multi-stage online signature verification system.” Pattern Analysis & Applications 5.3 (2002): 288-295.

[34] Ortega-Garcia, Javier, et al. ”Complete signal modeling and score normalization for function-based dynamic signature verification.” International Conference on Audio-and Video-Based Biometric Person Authentication. Springer, Berlin, Heidelberg, 2003.

[35] Kashi, Ramanujan S., et al. ”On-line handwritten signature verification using hidden Markov model features.” Document Analysis and Recognition, 1997., Proceedings of the Fourth International conference on. Vol. 1. IEEE, 1997.

[36] Yampolskiy, Roman V., and Venu Govindaraju. ”Taxonomy of behavioural biometrics.” Behavioral Biometrics for Human Identification: Intelligent Applications (2009): 1.

[37] Lager, Peter. ”G4P”. 31 May 2018. http://www.lagers.org.uk/g4p/

[38] ”libfreenect2”. 31 May 2018. https://github.com/OpenKinect/libfreenect2

[39] Labbe, R. R. ”Kalman and bayesian filters in python.” (2014).

[40] Breiman, Leo. ”Random forests.” Machine learning 45.1 (2001): 5-32.

[41] Chang, Chih-Chung, and Chih-Jen Lin. ”LIBSVM: a library for support vector machines.” ACM transactions on intelligent systems and technology (TIST) 2.3 (2011): 27.

[42] Pedregosa, Fabian, et al. ”Scikit-learn: Machine learning in Python.” Journal of machine learning research 12.Oct (2011): 2825-2830.

[43] Pimentel, Marco AF, et al. ”A review of novelty detection.” Signal Processing 99 (2014): 215-249.