University of Tartu

Faculty of Science and Technology

Institute of Technology

Hassan Shehawy

Optical Tracking of Forearm for Classifying Fingers Poses

Master Thesis (30 ECTS)
Robotics and Computer Engineering

Supervisor:
Karl Kruusamäe
Christian Nissler

Tartu 2018

ACKNOWLEDGMENTS

I'd like to thank Christian Nissler and Claudio Castellini from the DLR for the continuous help and guidance they offered to me during the entire work. I'd also like to thank Gholamreza Anbarjafari for helping me to learn image processing and machine learning that made me able to do what I'm doing. I want to thank Heiki Kasemägi, for all the help and flexibility he showed to me. I'm grateful for Karl Kruusamäe for first showing me what a great teacher means and second for the knowledge, support and patience.

Hassan Shahbazi

Abstract

Prosthetic robotics is one of the most rapidly developing fields of robotics and providing solutions to many people around the world. Hand amputees can greatly benefit from prosthetic arms and can perform many daily tasks that would not be true if not for prosthetic arm. Despite the availability of commercial arms in today's world, the high cost of such products makes it unattainable for many people and the need for cost-effective solutions arises more and more. 3D printing technology has made it available to get a prosthetic arm at lower cost. However, one of the main challenges in this application is the reconstruction of the intended motion of the fingers. A new approach has been developed to enable for predicting the intended motion using just a camera and a combination of image processing and machine learning techniques. However, this setup implies a fixed position of the arm which is not practical. In this project, a more robust setup is designed and tested to enable for the free motion of the arm as a proof of concept. Instead of using the AR tags coordinates relative to the camera frame, the transformation between each tag relative to other tags is used. LDA, Decision Trees and SVM are used for classification and their performance is compared.

CERCS: T111 Imaging, image processing; T125 Automation, robotics, control engineering; P176 Artificial intelligence

Keywords: non-invasive rehabilitation, prosthetic robotics, April tags, image processing, machine learning

Table of Contents

1 Introduction

According to the World Health Organization (WHO), there are over a billion people who have physical disability in our world [1]. The significant cost of disability is not easy to quantify but it includes social and economic sides which raises the need of rehabilitation services. Limb loss is one of the disability forms that has such a high need of rehabilitation and its prevalence rates in various countries has been estimated in various works. In the USA, there were around 41,000 upper limb amputees in 2005 [2] while in UK there are approximately 5200 cases reported every year [3].

Hand loss is a severe experience and leads to considerable environment inaccessibility and prosthetics can greatly improve the quality of life for hand amputees. However, such a need for prostheses is not met in many cases especially in developing countries. Many barriers face the existence and development of rehabilitation and cost is the most challenging one. Commercial upper limb prostheses can cost between $4000 to $10000 for body-powered ones and $25000 to $75000 for externally-powered ones in the USA for example [4].

Based on the WHO and the International Society for Prosthetics and Orthotics statistics, disabled person in need of prostheses or orthotics are about 0.5% of developing countries population and the available services are inadequate. Waiting times for a service or device also can extend to months, especially in rural areas. Beside the high cost, the lack of prosthetics professionals contributes to the wide shortage in these services [1] [5]. In many publications, especially ones that focus on healthcare policies, there are suggested recommendations to tackle the unmet needs for amputees. Recommendations include local manufacturing of devices, adequate training for using them and the availability of enough professionals to maintain devices and provide support when needed [1] [6] [7]. For example, some countries encourage the local production of assistive devices or importing the components and assembling locally by offering low interest loans for enterprises that work on aids for disabled persons and others provide exemption from taxes like Viet Nam [1]. Therefore, providing a cost-effective solution that could be produced easily and with minimum required operational knowledge is highly demanded. Using 3D printing technology has offered a promising solution that is easy and at a very low cost comparing to other alternatives. In [8] there is a review of available 3D printed hand prostheses. Hand prosthetics in general are either passive (mainly used as a decorative prosthetic) or can be actuated to provide some motion. For the latter type, the motion control can be done by body through shoulder or elbow for instance or can be externally controlled by using electric motors or pressurized air.

Providing a method to detect the amputee's intention for a certain finger motion can greatly facilitate controlling the prosthetic hand and provide intuitive usage of the device. There are multiple methods to achieve this and the method this thesis is built on uses computer vision and called Optical Myography (OMG). Despite the promising results of OMG, it has been implemented only with the arm fixed in a particular position and not allowed to move. In this work, the OMG method is extended to allow for the free placement of camera on the arm and

free movement of the arm. It is achieved by utilizing the relative coordinates of AR tags placed on the forearm and machine learning.

In section 1 of this thesis, the topic is introduced. Section 2 provides a background for hand prosthetics and the methods used for reconstructing the intended fingers movement. Section 3 gives an overview of the OMG method and the possible extension of it. Section 4 explains the proposed method starting from testing the idea to selecting the best camera orientation and tags layout. Classification using machine learning is then introduced and classification using three algorithms is compared. Section 5 provides the results and conclusions as well as further ongoing work.

2 Background

Providing cost-effective solutions for prosthetic arms is attracting more interest because of the rapid development of several related technologies, e.g., 3D printing. For example, e-NABLE [9] is an open-source community offering many designs for prosthetics for free download and 3D printing (Figure 1). Being open-source, their designs have been used extensively not only by people in need, but also by students and researchers in various projects. A mechanical design course project at Washington University [10] has utilized an open-source design to provide better performance in grips whereas in Yuan Ze university, they helped a person with a 3D printed arm to play guitar [11].

Figure 1. *Raptor Reloaded hand by e-NABLE* [9].

Despite the significant cost reduction related to using 3D printing technologies in manufacturing and maintenance, identifying the intended fingers motion is still a challenge. This identification helps to achieve intuitive grasping. There are several options for estimating the intended finger motions: electromyography, pressure sensors, computer vision.

One of the more widely used approaches for estimating intended finger motion is Electromyography (EMG) which is using the muscles electrical activity in the arm to identify the required movement for fingers. By adopting different machine learning algorithms, researchers have been able to implement real-time systems that can predict and provide intended motion. In [12], Support Vector Machines were used to achieve hand control with a four-finger robotics hand shown in Figure 2 whereas in [13], they used 32 electrodes placed on the forearm and applied Neural Networks to detect the intended fingers pose for providing dexterous motion. Using force sensors can also be applied for predicting the intended motion [14] [15]. This kind of sensors utilizes the use of the Force Sensitive Resistor (FSR), which changes resistance depending on how much pressure applied to its sensing area. FSR can thus be used for sensing the pressure distribution on the forearm. Such a pressure distribution can be mapped to specific fingers pose using machine learning.

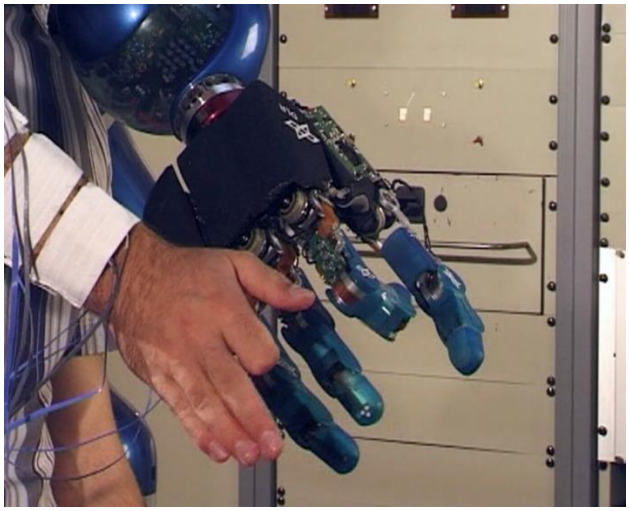


Figure 2. *Four-Finger hand controlled by EMG* [12].

An alternative to detect intended movement is the application of computer vision. In [16], a group of researchers have used a low-cost web camera with a computer to sense the posture of an arm (Figure 3). The project was mainly targeting stroke patients to facilitate reach and grasp exercise as a stroke rehabilitation system which is typically based on sending stimulating electrical signals to the forearm. These signals vary based on intended motion and an input is required to send the correct signals. So, they decided to use the elbow angle as the input and implemented a computer vision based system. They used color markers placed on 3 positions in the arm as shown in Figure 3 to sense the joint angles in the arm. In their approach, they used only a single 2D camera and then extracted the 3D information of positions. To make it more accurate and less sensitive to noise, they built a motion model for the arm that contains the patterns associated with arm positions in different movements. They then incorporate the model with the observed data from captured video and utilized a prediction and correction procedure. They could obtain an average accuracy of 91% and they also compared their approach to a commercial goniometer with an accuracy from 83% to 99% depending on angles.

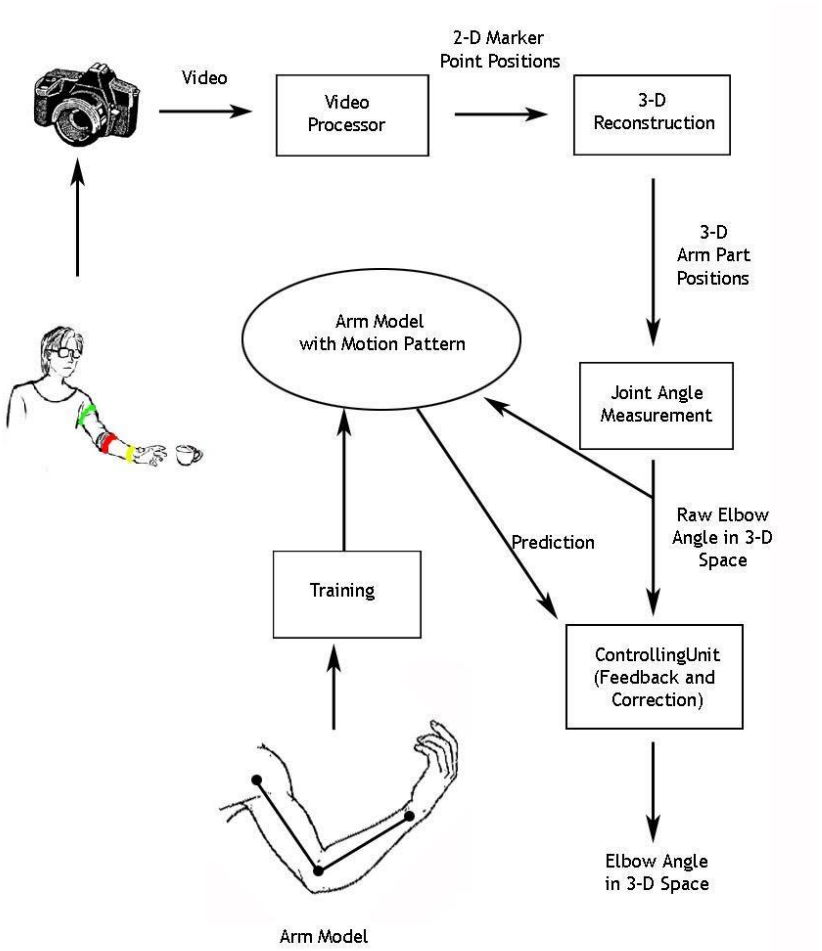


Figure 3. *Illustration for the arm posture sensing system* [16].

3 Previous Work

An approach called Optical Myograph (OMG) was proposed by a research group at the DLR (German Aerospace Center) for the reconstruction of intended fingers movement for a hand prosthetic application [17]. In OMG, computer vision and machine learning are used to detect surface deformations of the forearm and map them to fingers movement. The underlying principle of OMG is that the changes in the muscles activity of the arm related to fingers movements can be visually tracked because of the induced changes in forearm. In the proof-of-concept experimental setup (Figure 4) the forearm is strapped to a rig using a couple of bands. AR tags [18] are used as fiducial markers to track the changes of the forearm surface in different fingers poses.

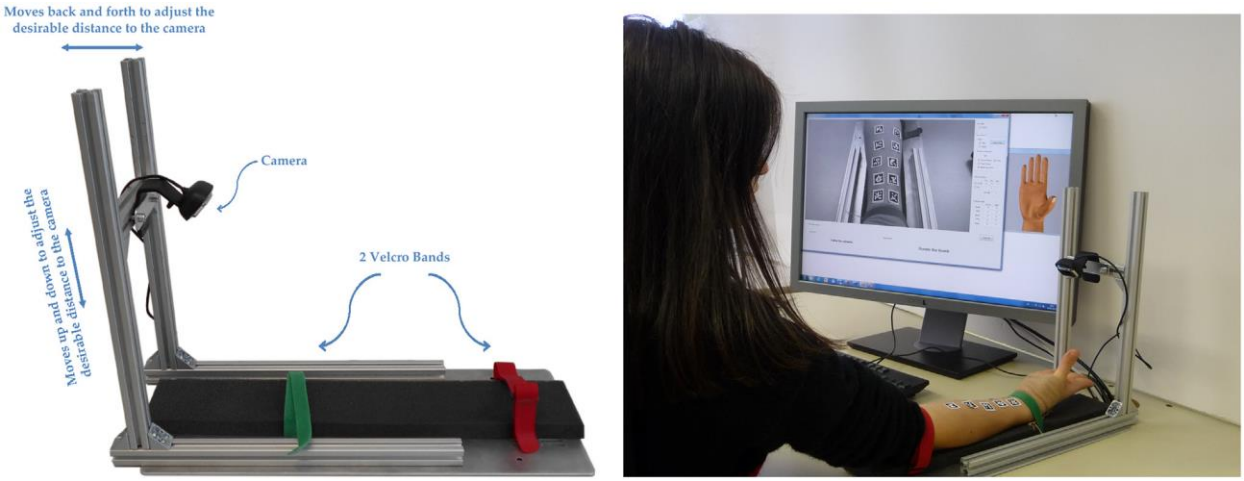


Figure 4. *Illustration for the OMG experimental setup* [17]

AR tags were printed and glued to the surface of the forearm and the person was asked to move their fingers based on a visual stimulus on the screen. The poses (position and orientation) of the 10 AR tags were tracked and saved as a dataset for the machine learning. For each AR tag, the corresponding translation is recorded in terms of both linear (x, y, z) and angular (yaw, pitch, roll) variables. Next step is filtering the signal and centering it around zero. Using the filtered data, two machine learning approaches were used (Ridge Regression (RR) as a linear one and Ridge Regression with Random Fourier Features (RRF) as a non-linear one). To test the approach, a combination of cross validation methods was used; the dataset was split into folds and the hyperparameters for the regression models were calculated.

Four different finger poses (thumb flexion, thumb rotation, index flexion, and combo flexion) were recorded for the machine learning part, and the detection of a given pose was then double-checked. Normalized Root Mean Square Error (NRMSE) is used to visualize the results and check for performance of the system in each case. the performance of OMG is compered to alternative methods, e.g., EMG, FSR, and Ultrasound (US) in Figure 5. Also shown the difference between

using RR and RFF models. The main conclusion that can be drawn for the comparison is that OMG stands well as an alternative method as the margins of errors are similar to other methods.

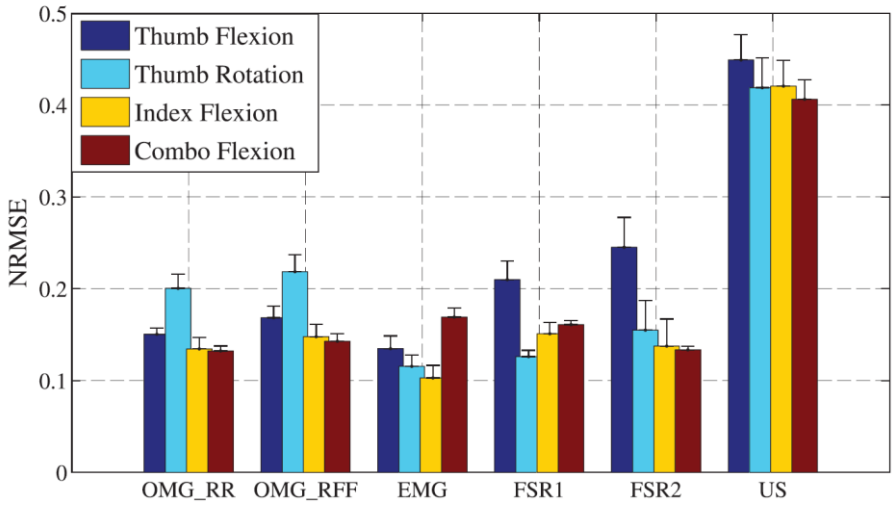


Figure 5. *Comparing OMG with 2 different regression models with other existing methods* [17]

In an enhancement to this approach, convolutional neural network and plain rectangular markers instead of AR tags were used [19]. In order to make the system more portable the camera was now attached to the arm instead of the fixed rig. OpenCV was used for image processing including the segmentation of the sticker, filtering the image, and applying morphological operations. First a median filter is applied to make the image smoother. Transforming the RGB channels into Log-Opponent-Chromaticity ones is done prior to segmentation. Application of morphological operations is utilized to compensate for the sticker's boundaries merging with forearm or sleeve. Figure 6 shows the experimental setup and segmentation process output.

(a) (b)

Figure 6. *Enhanced OMG setup (a) and output of segmentation process (b)* [19]

In order to do the classification of fingers poses, a convolutional neural network consisting of 2 layers, each having 16 filters was adopted. Here, 5 finger poses were fed for classification (same previous 4 plus the rest one) and accuracy level % is presented for each case of fingers in a box plot shown in Figure 7.

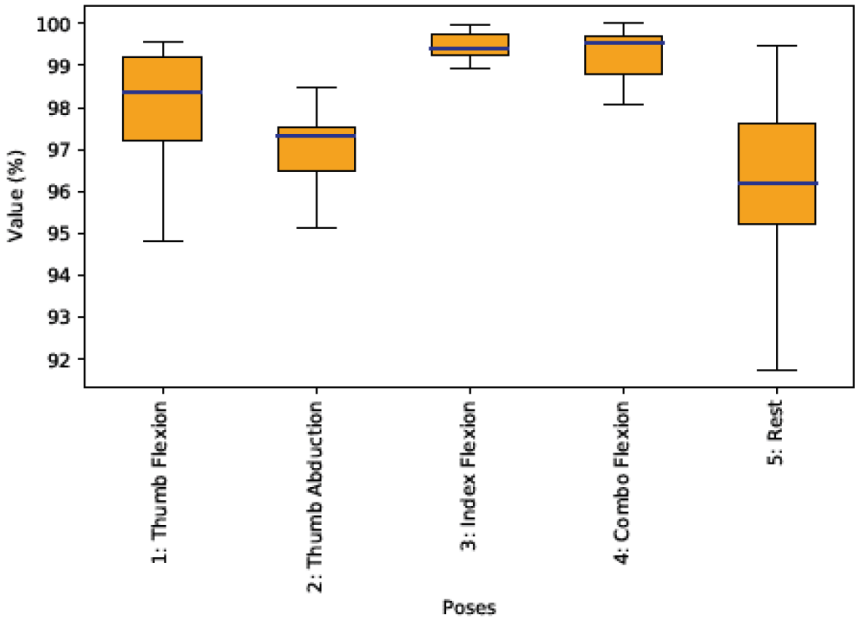


Figure 7. *Enhanced OMG classification accuracy* [19].

Results from OMG experiments show it can be an alternative method for predicting fingers pose. Classification of poses worked with a comparable accuracy to other methods. Using a simple machine learning method makes it computationally effective. Being robust to optical challenges (changes in contrast or brightness) is also an added advantage. The major breakthrough of this method is that the needed hardware for it is simply a camera and a computer which is widely available. However, in the described approaches, the arm was assumed to be in a fixed place during the experiments which forces the person not to move. Moreover, straps on the arm constrained it on the designed rig. In its next enhancement where the camera was moved to the arm, more robustness was added to the approach. Despite the flexibility offered by the placement of camera on the arm, the arm is still forced to be fixed on a plain surface.

4 Methodology

4.1 Problem definition and the proposed solution

In the OMG approach, the poses of AR tags relative to the camera frame were used for reconstructing the fingers poses. This approach works well with the assumption that the arm is always fixed in space, relative to the camera. However, this would not be valid if the arm is allowed to move freely, and a different method is needed. The proposed solution is using the coordinates of the tags, relative to each other. The coordinates are then fed to a machine learning algorithm to map the coordinates changes to fingers poses.

Initially AR tags were printed on a plain sheet of paper and pictures were taken of it from different angels to check for the accuracy of relative coordinates acquisition (Figure 12(a)). Smaller tags were then tested similarly. Tags were then placed on the forearm (Figure 15) and pictures were taken while the camera is in different orientations and changes in coordinates with fingers poses were analyzed to seek the best setup. With the camera placed on the arm, a video was captured of the forearm in different fingers poses and then was split into images that were fed to a software for detecting and localizing the AR tags. Based on this set of images, a time-vs-coordinates plot was made, and a pattern was observed. The pattern is analyzed by three machine learning algorithms for classification. Figure 8 summarizes the methodology.

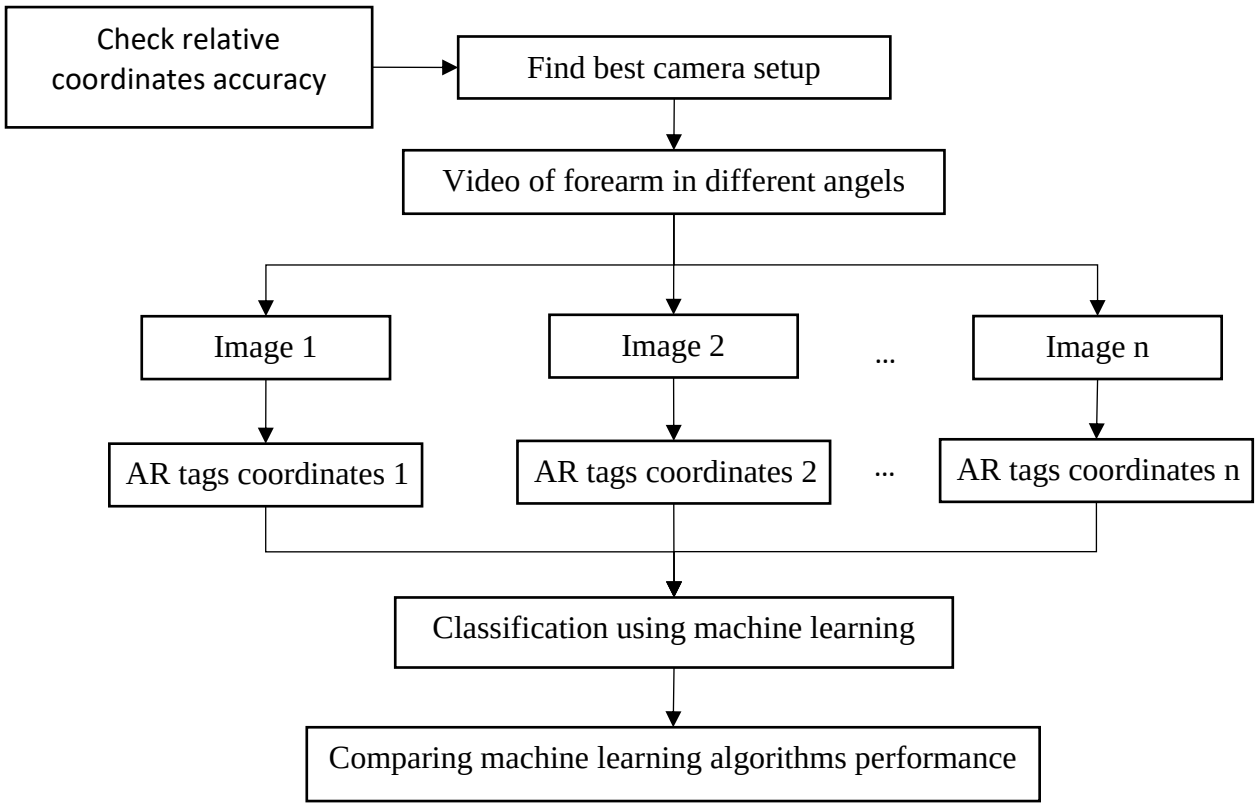


Figure 8. *Flowchart for the methodology*

4.2 AR tags detection and relative coordinates

The available code implementation of AR tags was used for detecting and localizing the tags [20]. Figure 9 shows interaction with a sheet of paper having some AR tags printed on it where tags are detected and identified. The output also includes poses of all detected tags and was saved in a text file. The details of detecting and identifying tags are described in [18].

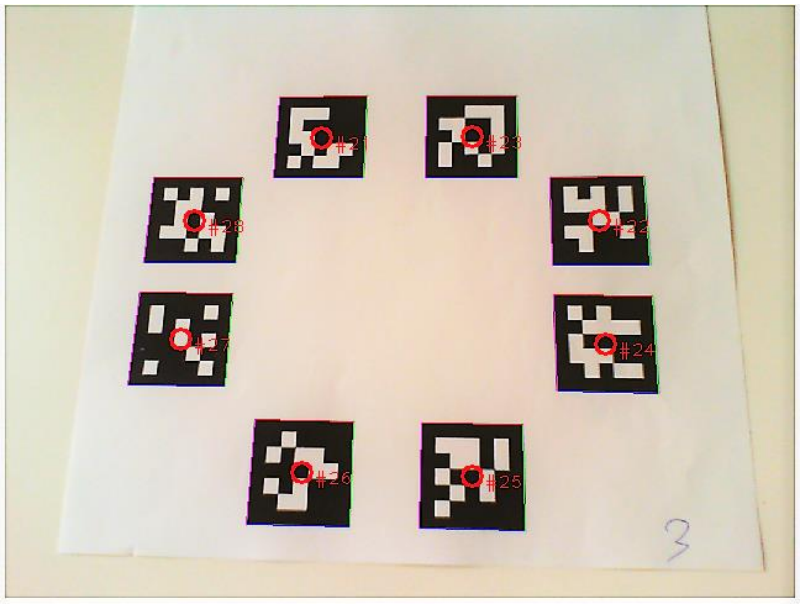


Figure 9. *AR tags detected and identified by their id (21 to 28)*

The mathematical relationship between coordinates of a point in an arbitrary 3D coordinate system (known as object or world coordinate system) and the camera 3D coordinate system (Figure 10) can be described by the following equation:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = R \begin{bmatrix} W \\ V \\ U \end{bmatrix} + t = [R \mid t] \begin{bmatrix} W \\ V \\ U \\ 1 \end{bmatrix} = \begin{bmatrix} r_{00}\, r_{01} r_{02}\, t_x \\ r_{10}\, r_{11} r_{12}\, t_y \\ r_{20}\, r_{21} r_{22}\, t_z \end{bmatrix} \begin{bmatrix} W \\ V \\ U \\ 1 \end{bmatrix} \qquad (1)$$

(X, Y, Z) describes the point in camera coordinate system and (W, V, U) in the world coordinate one. R is the rotation matrix and t is the translation vector between the two coordinate systems [21] .

The image is formed on an imaging surface where the image coordinate system is located and it's a 2D system (Figure 10). The transformation between the 3D camera coordinate system and the 2D image plane is given by the following equation:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = s \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \qquad (2)$$

(x,y) is the point coordinates in the image 2D plane, fx and fy represent the focal length of the camera and cx and cy compensate for the offset of the camera optical center and image plane center. Distortion effects are ignored here. s is a scaling factor [21].

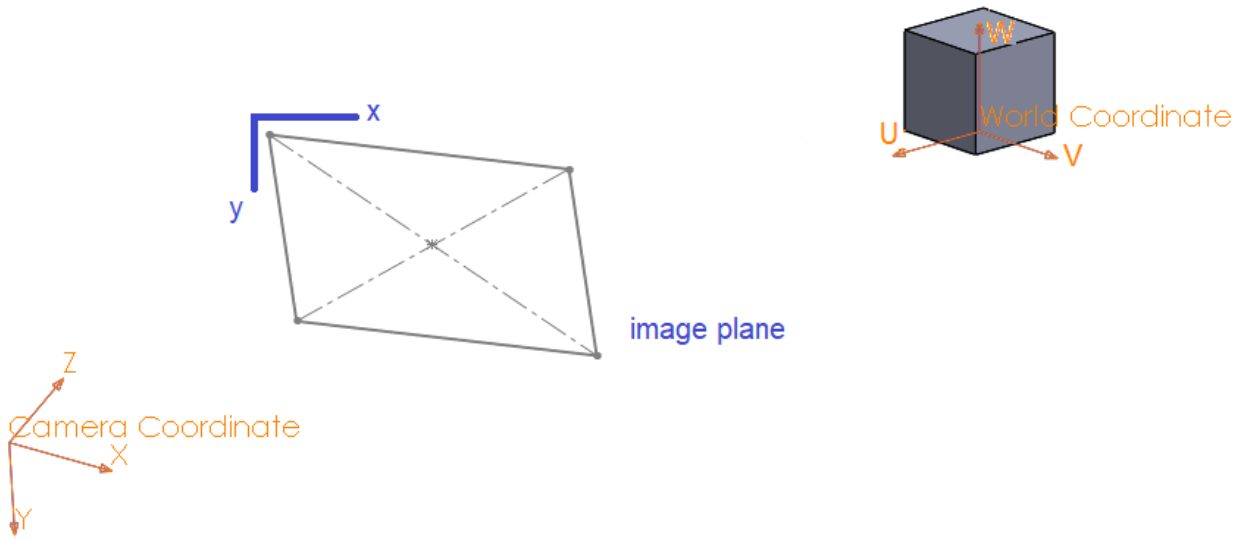


Figure 10. *Three coordinate systems (world, camera and image)*

Since the world coordinate system is arbitrary, it can be selected to have its origin at AR tag's center (Figure 11). Consequently, the four points defining an AR tag can have the coordinates in world systems of: (-h, -h,0), (-h, h,0), (h, h,0) and (h, -h,0) where h is half the tag size.

Given the (W, V, U) and the camera properties (fx, fy, cx, cy) combined with the detected four points locations on the image plane, equations (1) and (2) can be solved for the 3D pose of the tag using the Linear Direct Transform method [22]. There is a function to solve this problem inside the OpenCV library, named solvePnP and it was used in this work [23].

In this work, the interest is in the transformation of tags relative to each other and not to the camera. Consequently, a relative transformation between any two tags is expressed in terms of three directions and will be denoted as X, Y, Z. This is not to be confused with camera coordinates. It's used instead of W, V, U to give more intuition.

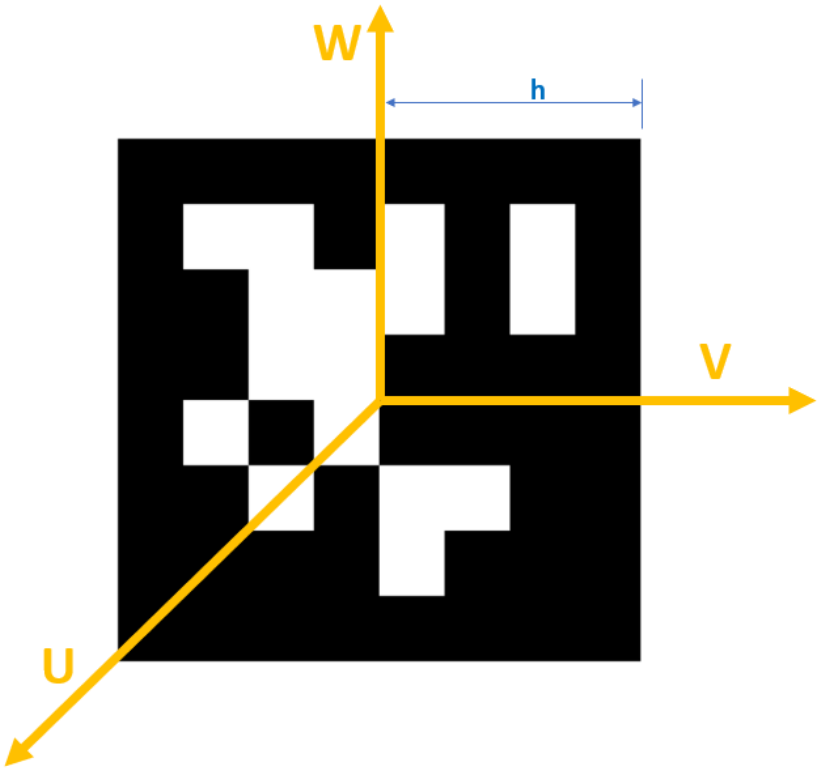


Figure 11. *World Coordinate System centered on an AR tag*

The code in [20] was modified to compute the tags pose relative to each other. With the [R|t] matrix computed for two tags (will be named M for short), their pose relative to each other can be calculated by following the same procedure. To achieve this, a matrix is needed that makes the transformation between the two tags.

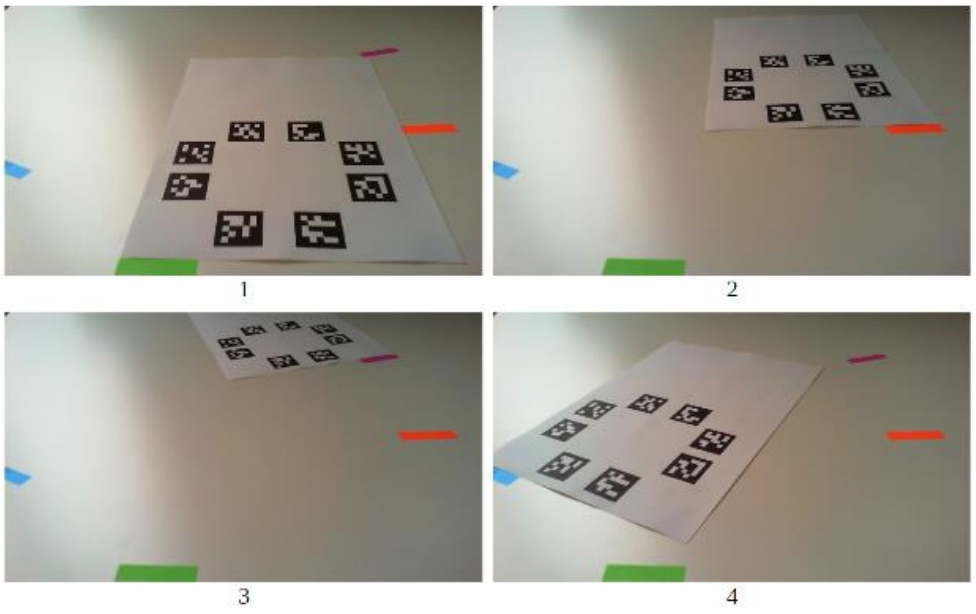
The two tags are assumed to have same size and therefore, their 4 points have same coordinates in world coordinate system. Assuming the coordinates vector in world coordinate system is P, then $P_1 = M_1 P$ and $P_2 = M_2 P$ where M_1 and M_2 are transformation matrices for the two tags, P_1 and P_2 are coordinates vectors in camera coordinate system.

If the required transformation matrix between the two tags coordinate systems is called M_r, then $M_r P_1 = P_2$ and it follows that:

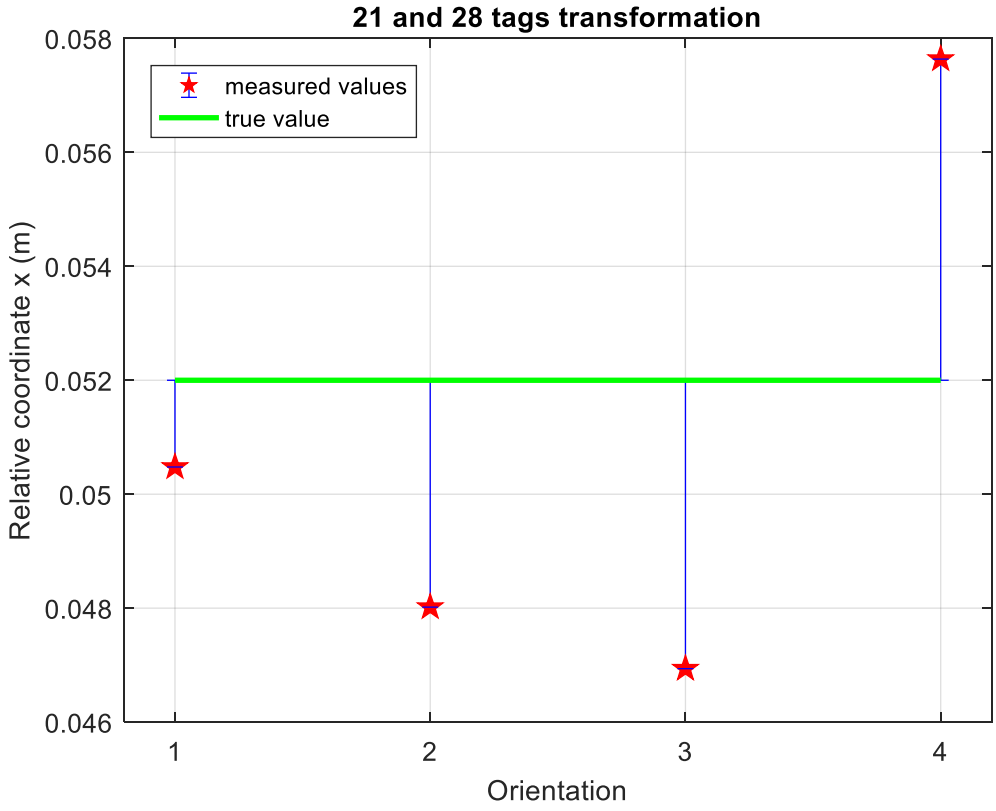
$$M_r M_1 P = M_2 P \;\Rightarrow\; \therefore M_r M_1 = M_2 \qquad \therefore M_r = M_2^{-1} M_1$$

The accuracy and precision for the software was tested by capturing photos of the sheet of paper from different angels. Figure 12 (a) shows the printed tags in 4 different orientations (numbered 1 to 4) and Figure 12 (b) shows the the measured value for the relative transformation between tags 21 and 28 in x direction in each orientation. For comparison, the true value is added to the

graph. Orientation was changed to allow for translational and angular changes. Results show an error range of 2.9% to 10.8%.



(a)



(b)

Figure 12. *AR tags on plain sheet of paper in different orientation (a) and relative transformation between 2 tags (21 and 28) in x direction (b)*

The low value of the error suggests the practicality of the idea. However, change of coordinates in different fingers poses is expected to be small and if the error gets high, the system may be confused with another pose than the true one. Therefore, the system must be designed carefully, and the range of error should be checked against values related to poses changes.

The code was used as the main software for detection of AR tags and computing the relative coordinates.

4.3 Arm and forehand muscles

The movement of fingers and wrist is a complicated type of motion because of the various degrees of freedom involved. The finger motion is controlled by muscles located in the forearm. Attached to the muscles are the tendons which are fibrous cords that also attach to bones. As the muscles contracts, the tendons move the bones and joints they cross. Figure 13 illustrates how the flexor pollicis longus muscle (in bold) allows the bending of thumb tip [24]. The muscles actions create clear and distinguishable deformations in the forearm surface which can be recognized easily.

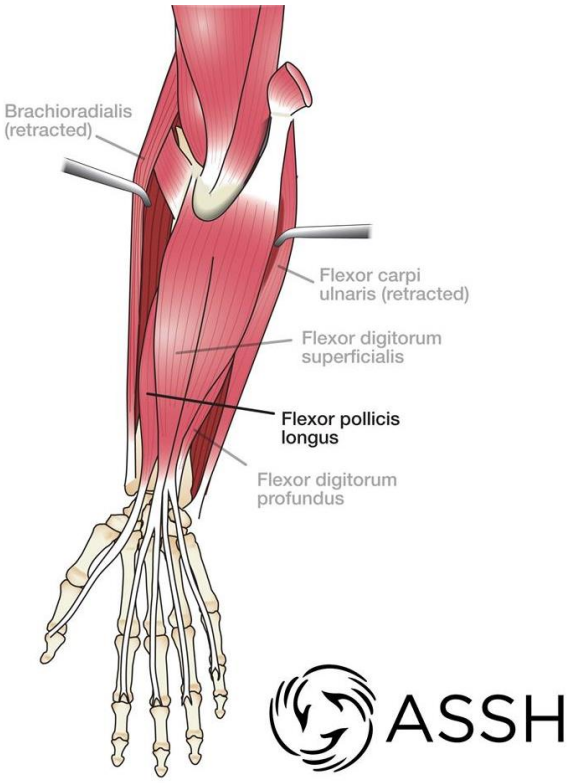


Figure 13. *Forearm muscles* [24]

4.4 Experiments

In the experiments, a web camera attached to a computer was used. Video was captured using VLC software and then divided into shorter videos, each for a different camera orientation. Each video was then split into images. These images represent the database used for machine learning.

The camera (Figure 14) is Microsoft LifeCam HD-3000 which is a very low cost (~30 €) device offering reasonable video quality and rate (1280 x 750 pixels at 30 fps). The camera was calibrated for obtaining its optical properties. Calibration was done using checkerboard and an in-house software at DLR.

Figure 14. *Camera used*

4.5 Tags on arm

An array of AR tags was then placed on the forearm as shown in Figure 15 and the camera was used to acquire pictures for it while held in hand and in different angels. Here two fingers poses were captured; stretching the five fingers and fisting them, named as stretch and fist respectively (numbered as (1) and (2) respectively in Figure 15). Camera orientation was changed four times (a – d).

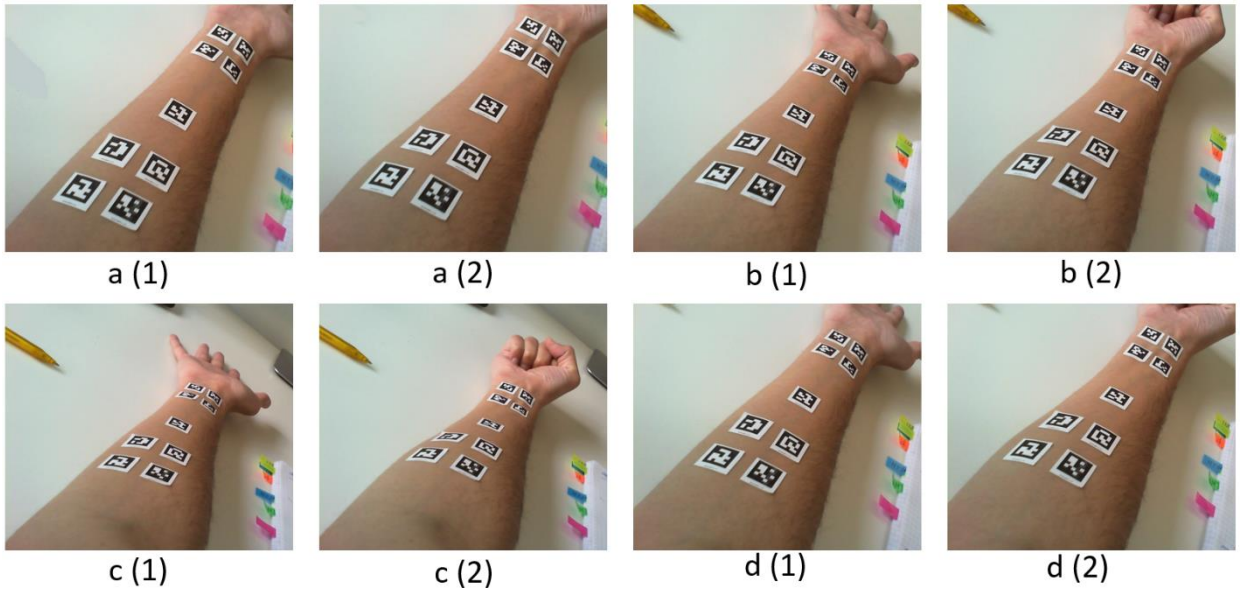


Figure 15. *AR tags on forearm in different angels while fingers in 2 different poses*

These images were fed to the software and relative coordinates between detected tags were obtained. Figure 16 shows coordinates changes with the fingers pose for relative transformation between two tags. On x axis is the camera orientation (a – d) and y axis gives the value of coordinate in both stretch and fist cases. Despite the difference in camera angel between a, b and d, a pattern is observed, and we can have a good threshold value to differentiate between the two poses. However, in the case of orientation c, values were not consistent.

Figure 17 shows another example of coordinates change for the setup of Figure 15. Using another combination of tags could improve the results for classification for orientation c. The challenging nature of classification in case of orientation c can be explained by looking into figrue15-c which shows that not all tags can be easily detected, if compared to other orientations.

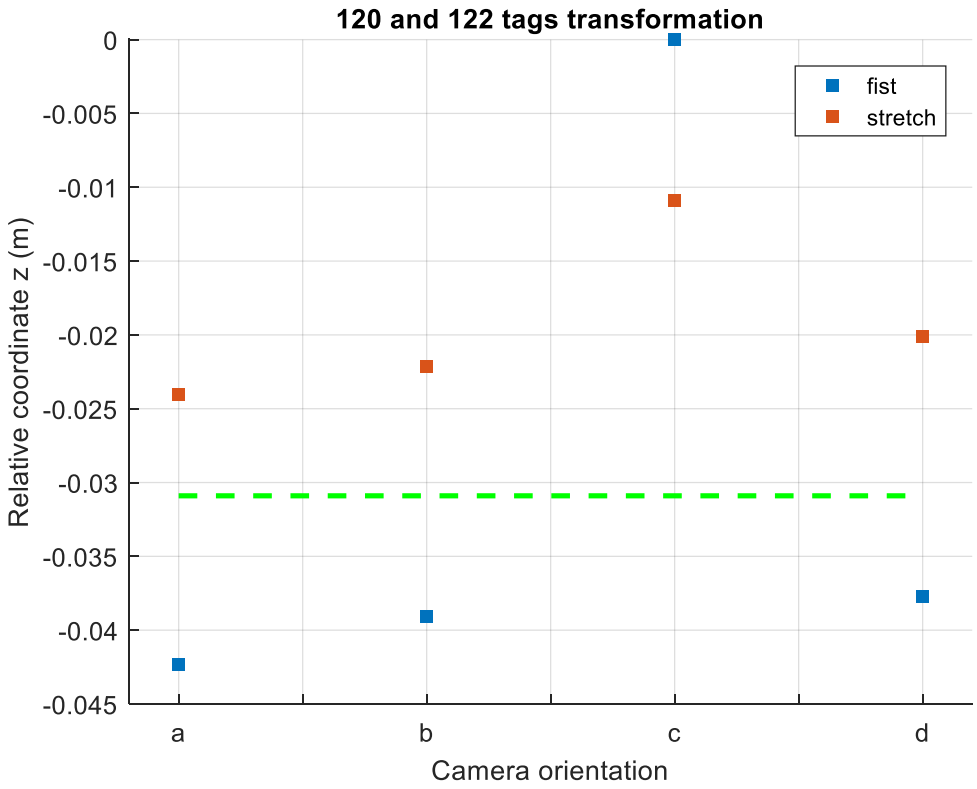


Figure 16. *Relative coordinates between tags in two fingers poses and 4 camera angels*

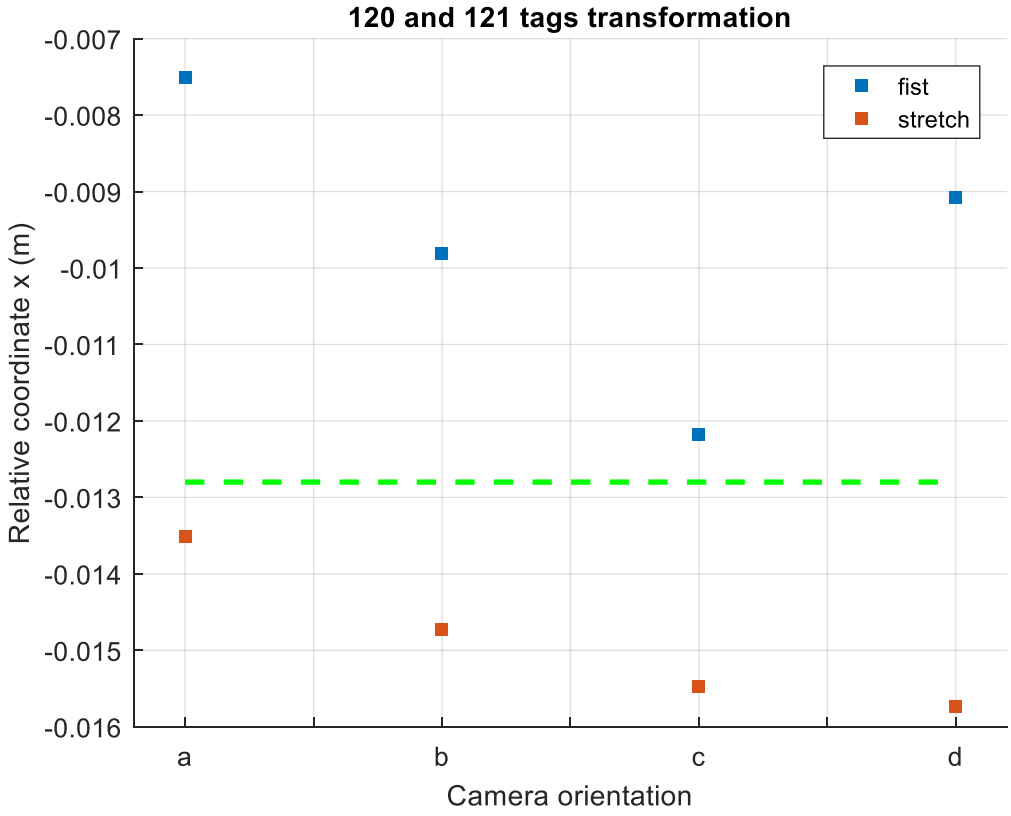


Figure 17. *Relative coordinates between tags in two fingers poses and 4 camera angels*

4.6 Camera Setup

The camera was placed on a rig and pictures were taken of the forearm in different fingers poses. Experiment was repeated while the camera was in different orientations and comparison was made between changes in coordinates with orientation to poses. Initially, four setups were considered during this work as illustrated in Figure 18. However, the second and third setups were omitted because they do not give much freedom for the person.

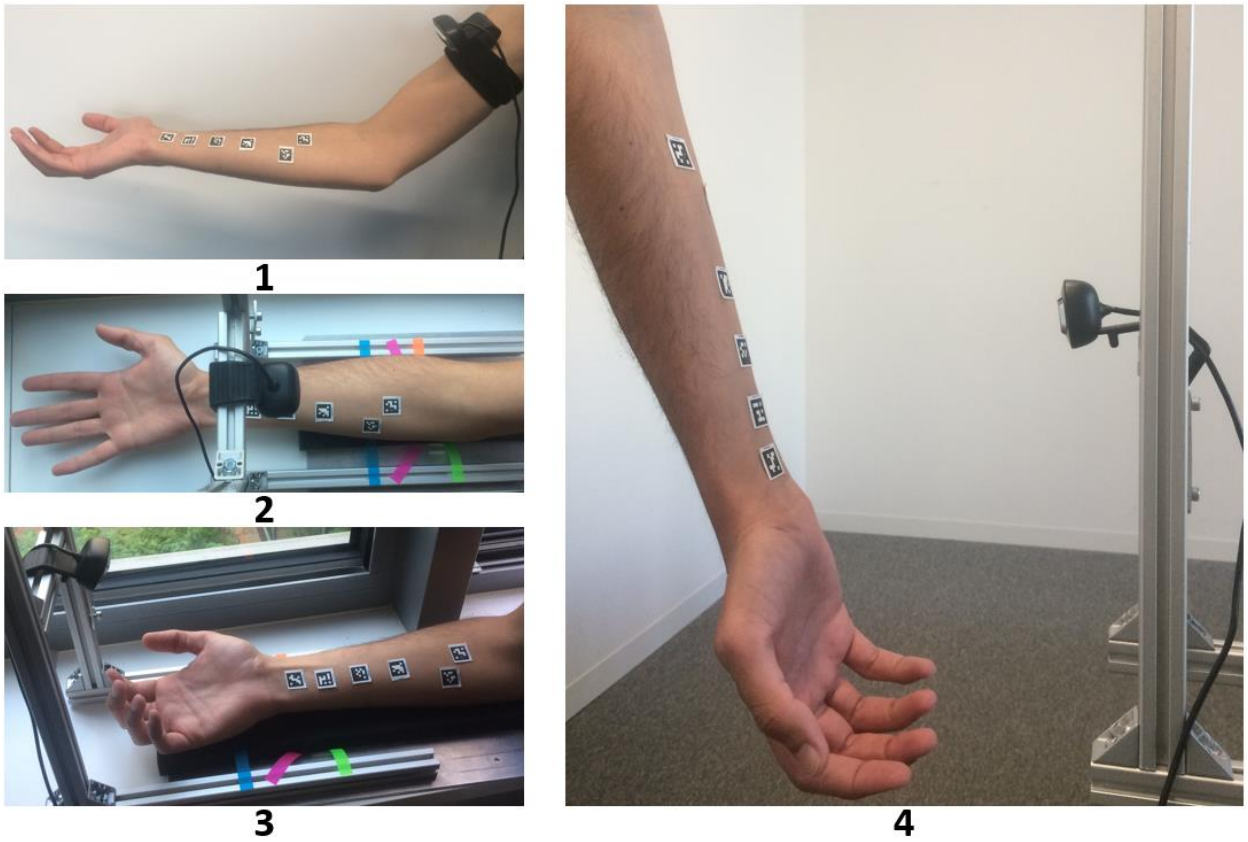


Figure 18. F*our camera setups*

The two selected setups were tested, and multiple pictures were taken for the forearm to compensate for any possible noise. Images were fed to the software and coordinates were calculated. Principal Components Analysis (PCA) was applied to remove outliers and find a statistical mean. PCA is a statistical procedure which is basically based on finding the principal components, the first of which has the largest variation in data [25]. Figure 19 illustrates it.

For comparison, video was captured for 2 fingers poses, fist and stretch, while the camera was in two orientations. Obtained results for 2 tags relative transformation are shown as a graph, one using x coordinate (Figure 20-a) and the other using z (Figure 20-b). The specific setup is shown on x axis with the number beside the name refers to the first or second orientation. For example, arm1 and arm2 mean the camera was on arm and in orientation 1 and 2, respectively. In the case of camera on arm, it is clear that the 2 poses can still be differentiated while the camera is in

different orientations. Moreover, the difference in values between the two poses is the same in the different orientations. However, when the camera is on rig the pattern is not consistent. This stems from the inability to obtain a threshold that applies for both rig1 and rig2. Moreover, the difference between stretch and fist values is changing between rig1 and rig2.

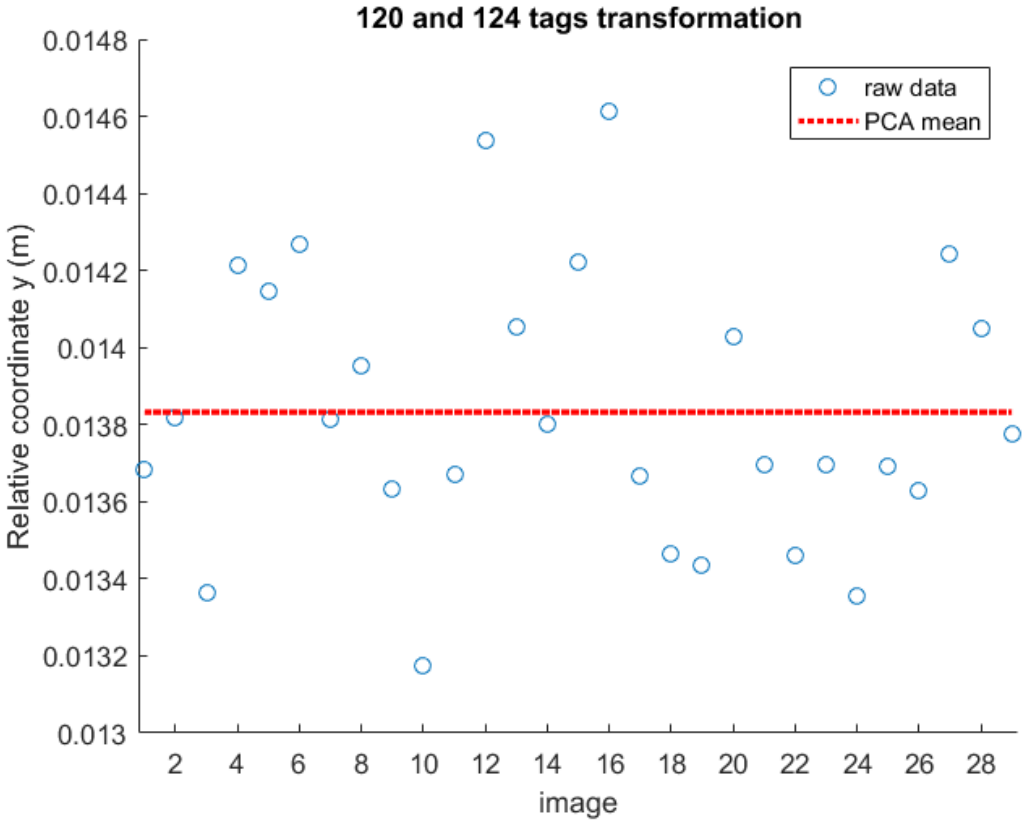


Figure 19. *Relative coordinates between tags after PCA*

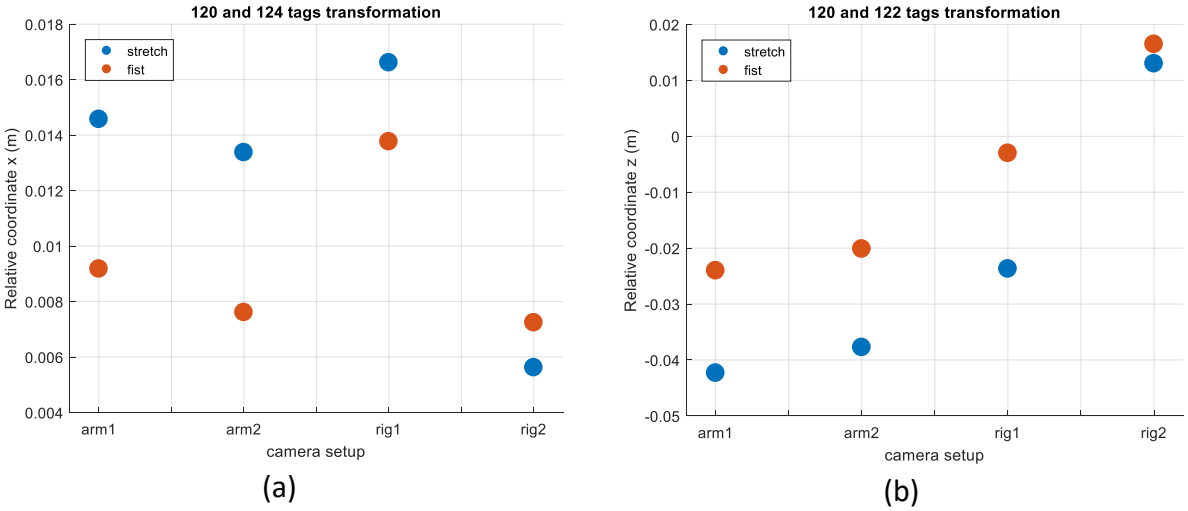
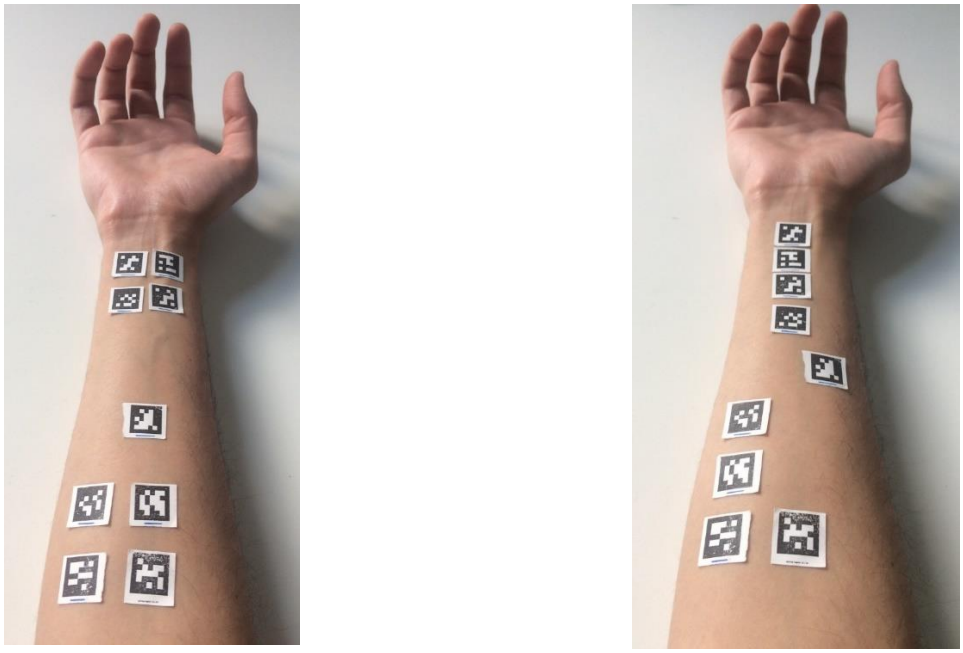


Figure 20. *Changes of coordinates in 2 poses with different camera setups and orientations*

4.7 Layout of tags

The layout of tags over the forearm was changed to seek a better one. Two layouts are shown in Figure 21. The layout in Figure 21(b) was preferred over the one in Figure 21(a) because it had less rate of not detecting the upper tags near the hand and it provides more information about the forearm dynamics.



(a) (b)

Figure 21. *Two tags layouts*

Different tags were used during the changes of testing and/or changing algorithms or experiments. So, to avoid the confusion with tags identification by their id, numbering scheme was adopted. The tag in the middle is titled the master tag and assigned the number 0 while other tags are numbered from 1 to 8 as shown in Figure 22.

Moreover, the relative transformation between any two tags will be named as *Tlmn* where l and m are the tags numbers and n is direction (x, y or z). For example, to refer to the relative translation between tag 0 and 1 in x direction, the term T01x is used.

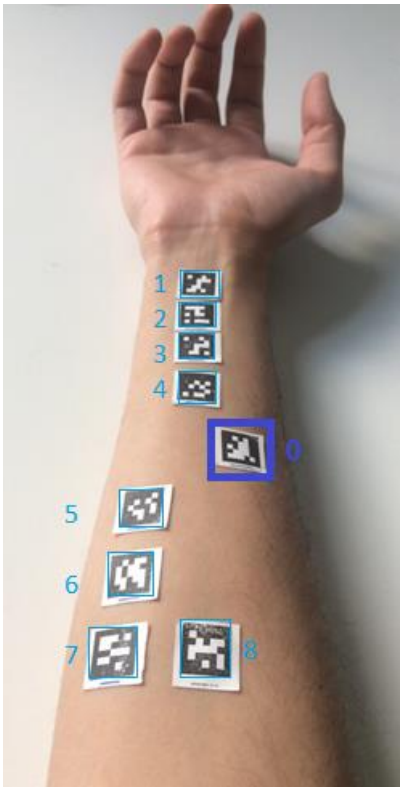


Figure 22. *Tags numbering*

4.8 Poses identification

Four hand configurations (Figure 23) were added as a further complication for the identification. Images were taken of every pose and transformations were computed. It was noted that some coordinates gave very clear distinction while others were more challenging.

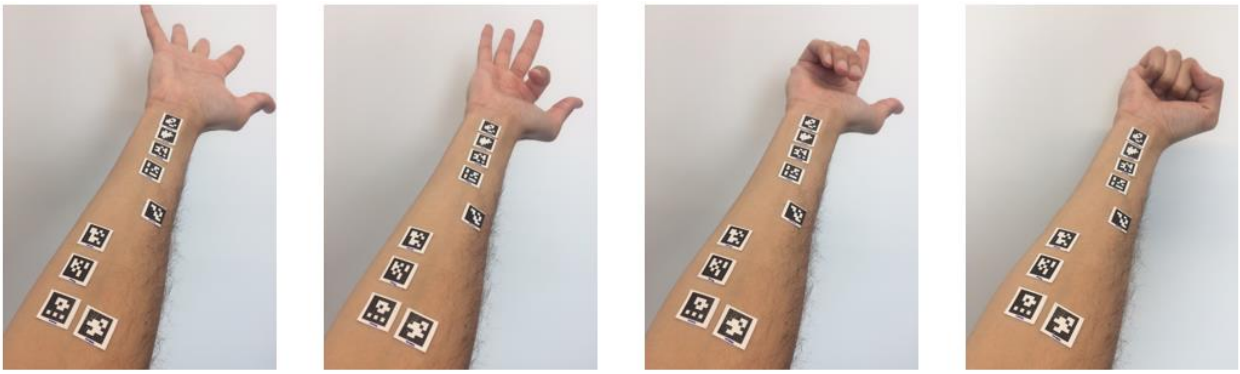


Figure 23. *Four fingers poses*

T02z is an example for a distinctive feature of the four poses. Outliers and transition data were removed from results and plotted (Figure 24). An average value per each pose can be found and learned for identification.

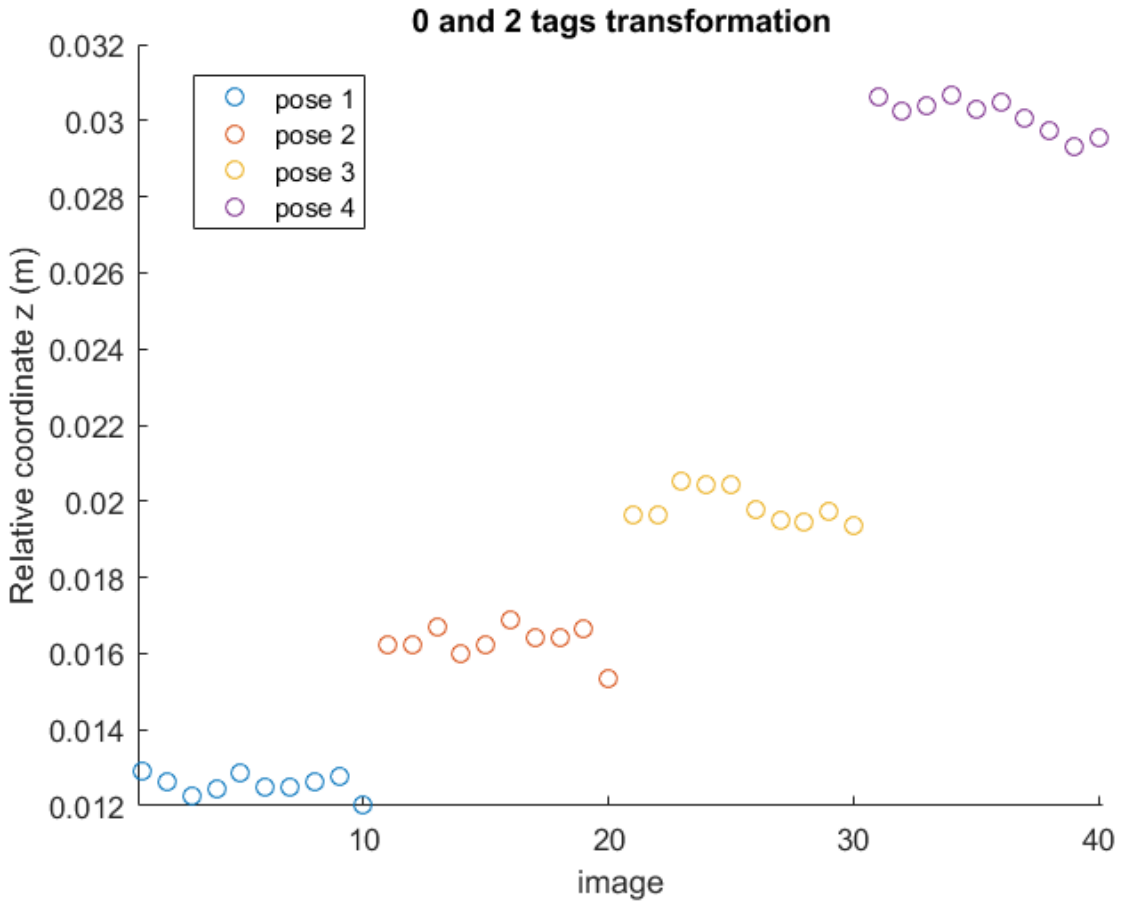
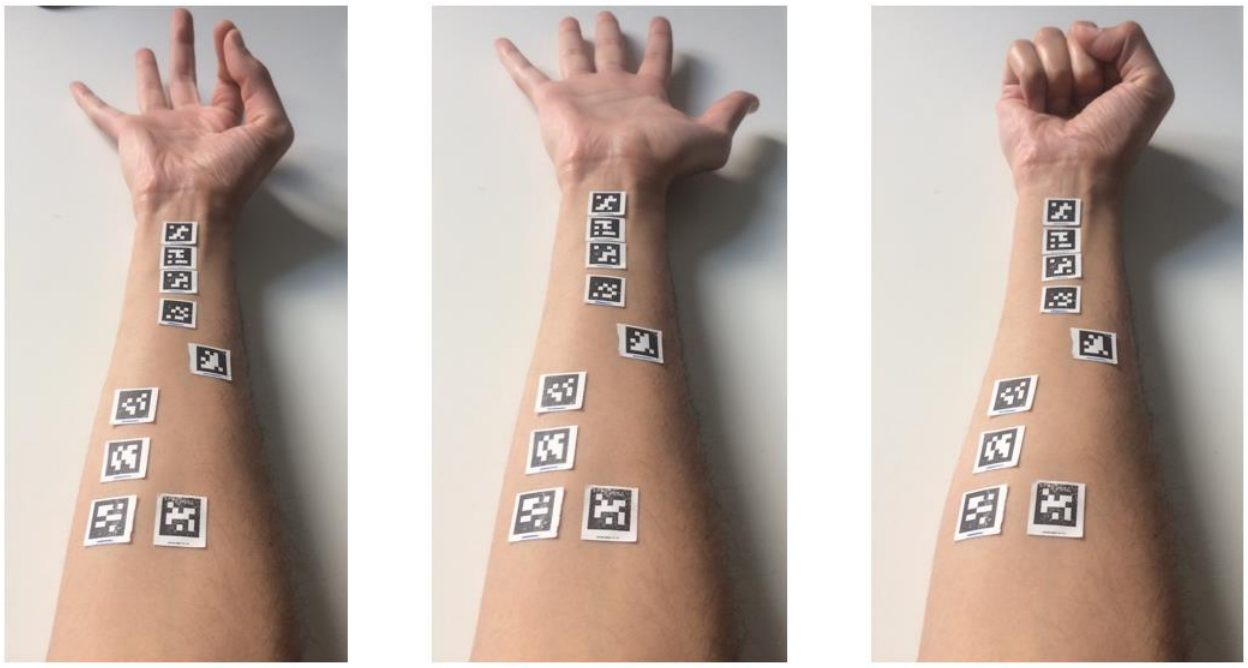


Figure 24. *T02z changes in four fingers poses*

4.9 Initial pattern

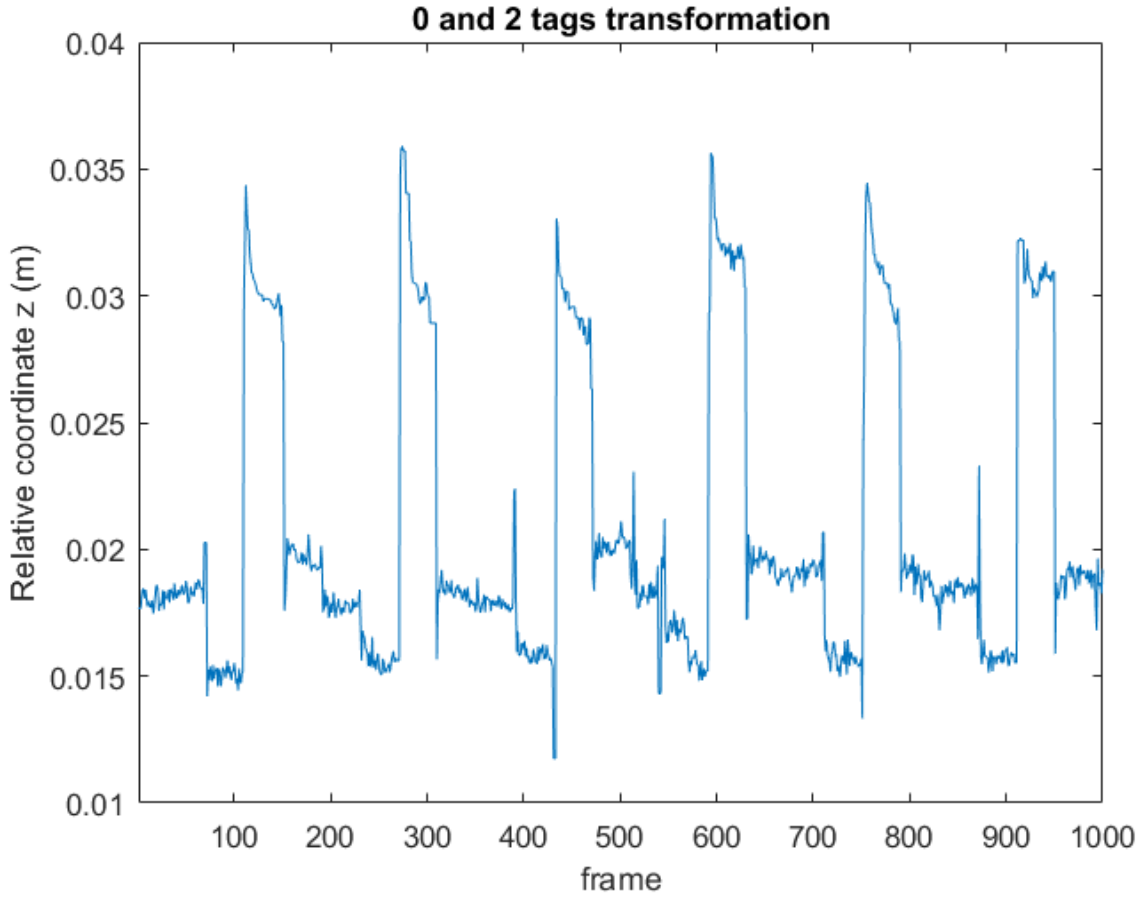
In searching for a pattern, a video was captured for the forearm while doing three poses (two grasps - fist and pinch - and stretch) repeatedly and the camera orientation was changed in the middle of the video. Video was then split into frames to be analyzed for tags identification and localization. Similarly, some features were better than others. It was noted that T02z also provided a better pattern than other ones. Figure 25 shows the three poses.

Figure 26 (a) shows raw data collected from all frames that shows a repetitive pattern over the entire video. Filtered data is shown in (b) and the pattern is obvious. The lighter colored points refer to the first camera orientation while the darker ones refer to the second orientation.

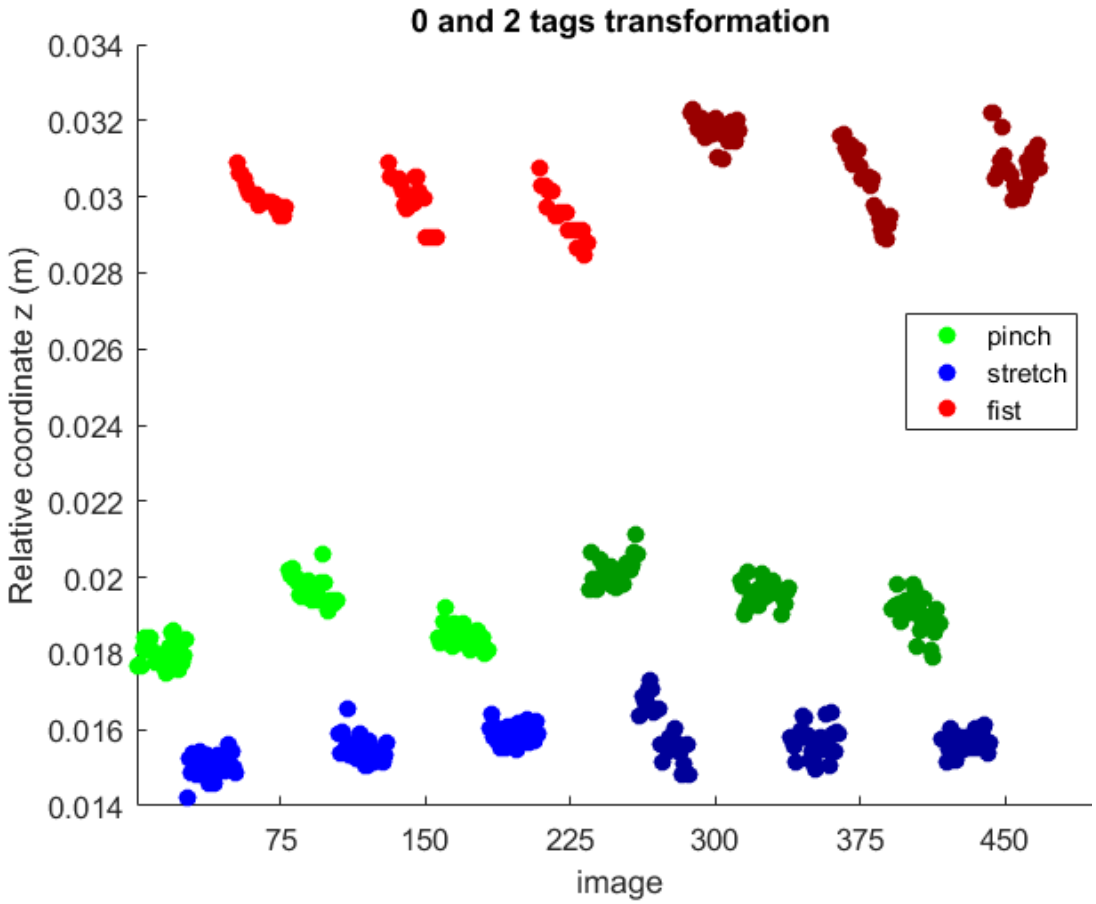


pinch stretch fist

Figure 25. *Three poses (two grasps and stretch)*



(a)



(b)

Figure 26. *T02z in two grasps and stretch, all values (a) and filtered (b)*

4.10 Machine learning

For classification using machine learning, video was captured of the three poses (pinch, stretch and fist) from three different camera orientations. The video was then split into images and analyzed by AR tags software to detect and localize the tags. Relative transformations between tags are used as features to create the training model. Three different algorithms were tested for classification.

4.10.1 Database

Different poses were recorded as videos and split into images. The resulting database contains approximately 10,000 images. It is divided into subsets, each for a different combination of poses. Moreover, for the two grasps and stretch poses, two independently captured videos were split into two sets of images, each of 2000 images.

4.10.2 Filtration

Various noise sources can result in odd data and not detecting one or more tags in different images. This is particularly happening in the case of occlusion. Therefore, some filtration is needed. The output of the software is saved in a text file and contains a list of all detected tags with their relative transformations. A simple regular expression [26] was used to check if there are any tag(s) that are not detected. In the filtration step, if a certain tag is not detected, its coordinates are replaced with zeros for that image. Prior to the machine learning step, zeros and outliers are replaced with previous values. This could be achieved in a real-time system by providing last know coordinates of a tag if it is not detected.

4.10.3 Linear Discriminant Analysis (LDA)

Given a dataset, the LDA tries to find the directions that could provide best separation between the classes of data. As illustrated by Figure 27, one direction could provide a very good separation between classes if data is projected along it (direction 1) while another may not be a good one (direction 2). LDA helps in identifying such direction(s). Even though this is illustrated for two-dimensional data, it can be extended for higher dimensions. Given n vectors (representing n features) for the data, each is of d-dimensions as input, mean vectors for each class are calculated and scatter matrices, both within class and between classes are computed. Eigenvectors and eigenvalues are found for these to obtain the direction that gives maximum distance between classes and minimum one within the class. Finally, original data is projected along these directions. [27]

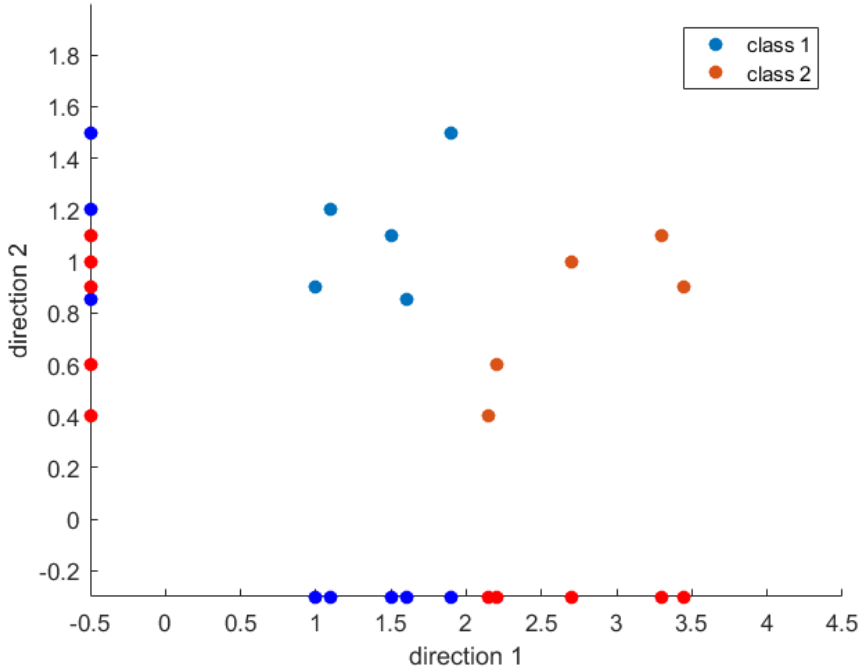
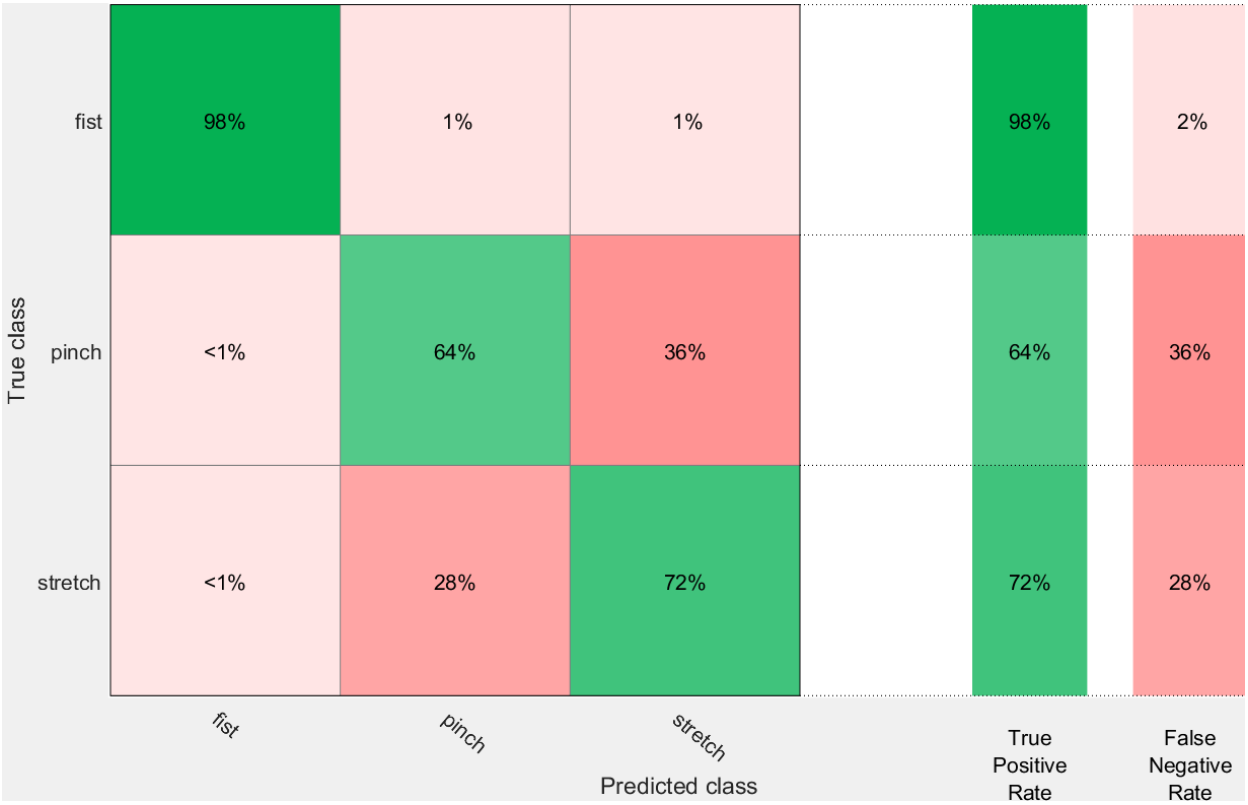
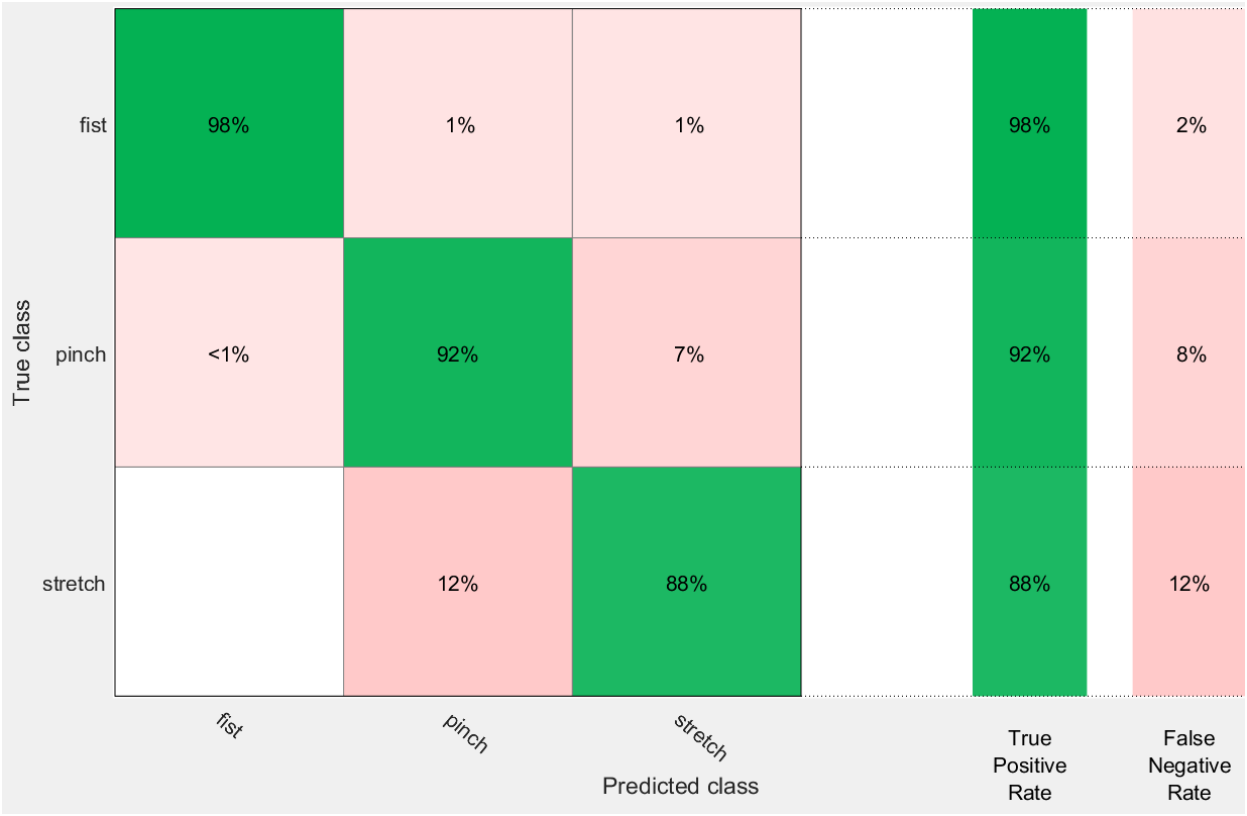


Figure 27. *Illustration for LDA*

Initially, all or most tags transformations were used as features. However, poor results were obtained (appendix 7.1). This is probably because not all the transformations can be linearly separated. Therefore, three transformations (z coordinate) were selected; tag 0 to 1, tag 0 to 2 and tag 0 to 3 because they showed best pattern among others. With these 3 features, classification was possible and accuracy level is around 92%. Figure 28 shows the classification results which indicates that both stretch, and fist classes can be easily classified using this method.



(a)

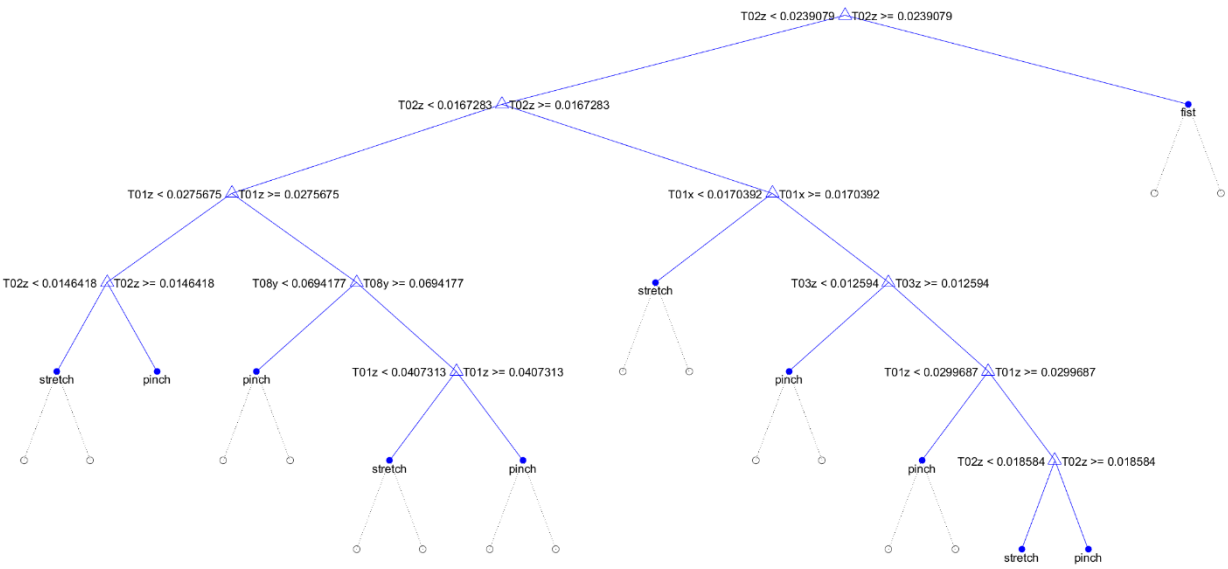


(b)

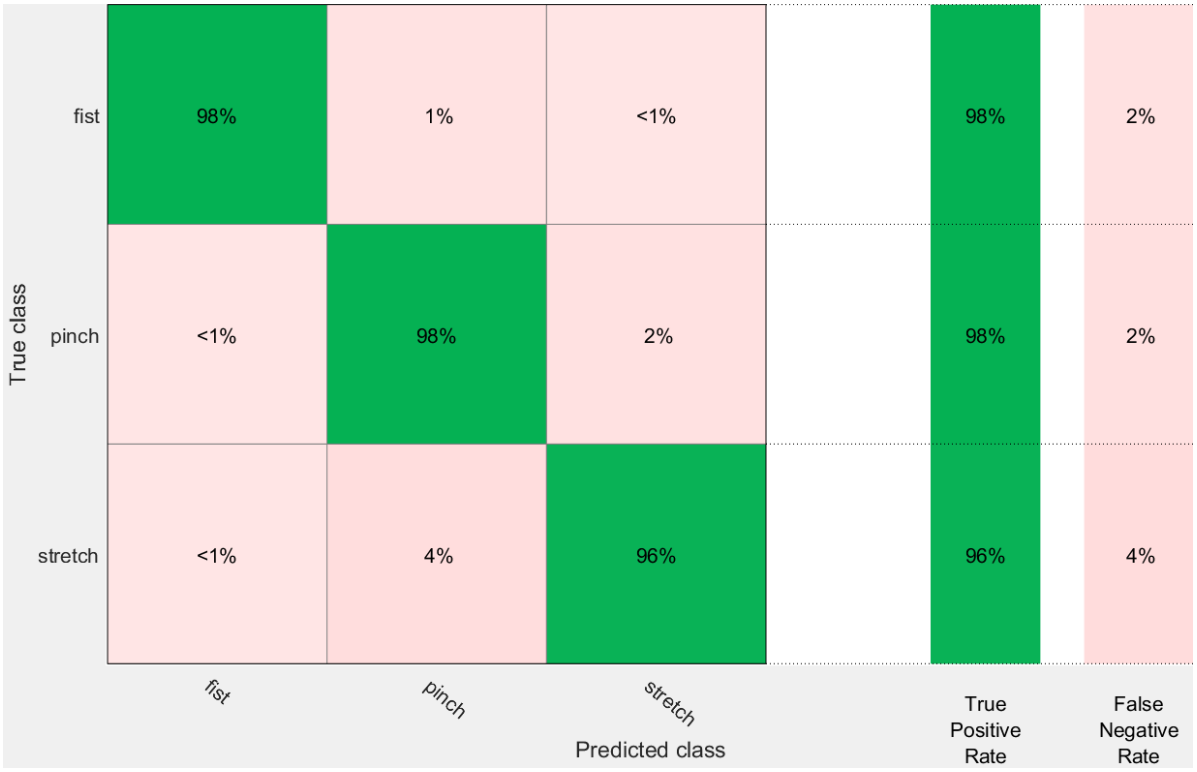
Figure 28. *LDA classification, with all features (a) and with only 3 (b)*

4.10.4 Decision Trees

Decision trees can be used as a statistical method for predicting a class value based on input variables. The tree is composed of nodes, branches and leaves. The nodes have the input variables and based on their value, a branch is selected. At the end of the branch there is another node, which can either have another variable for decision or carry a class label and in that case, it is called a leaf [28]. Figure 29(a) shows part of the tree structure representing the model for classifying the fingers poses.



(a)



(b)

Figure 29. *Decision Trees structure (a) and classification results (b)*

Figure 29(b) shows classification results using decision trees and utilizing all the features. Accuracy level was around 97% and it outperformed the LDA.

4.10.5 Support Vector Machines (SVM)

Given data in 2D space as shown in Figure 30 where each axis represents a certain feature, SVM tries to find a separating line between these points representing two classes. The algorithm tries to find the best line that could separate the data (Figure 30). Working in higher dimensions means instead of finding a line, finding a hyperplane. In some cases, data points can not be linearly separated. However, if projected on a higher dimension function (kernel function), linear separation is possible. [29]

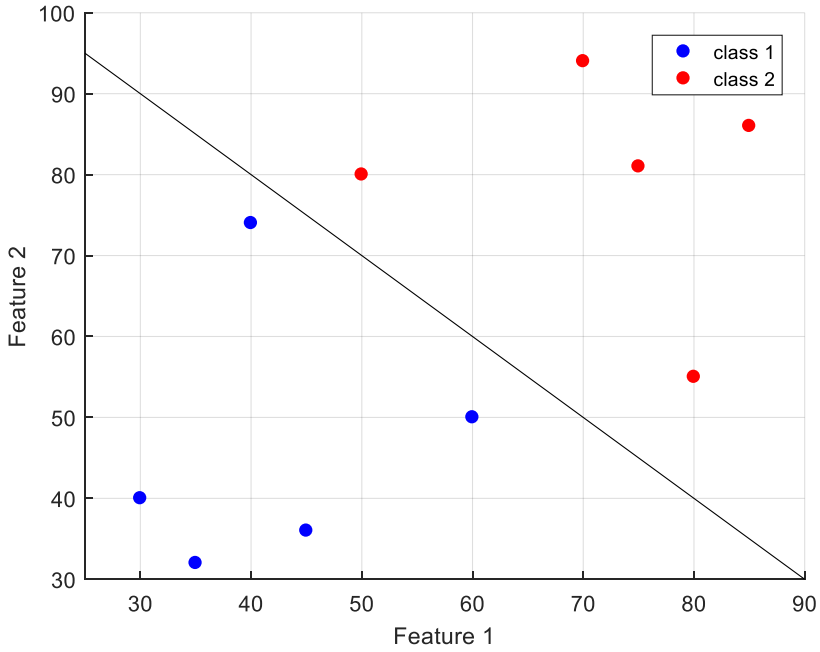


Figure 30. *SVM Illustration*

All features were used for SVM classification and accuracy level was, in average, 98%. By changing the kernel function to quadratic and cubic, even better results were obtained. Figure 31 shows classification results when using a cubic kernel.

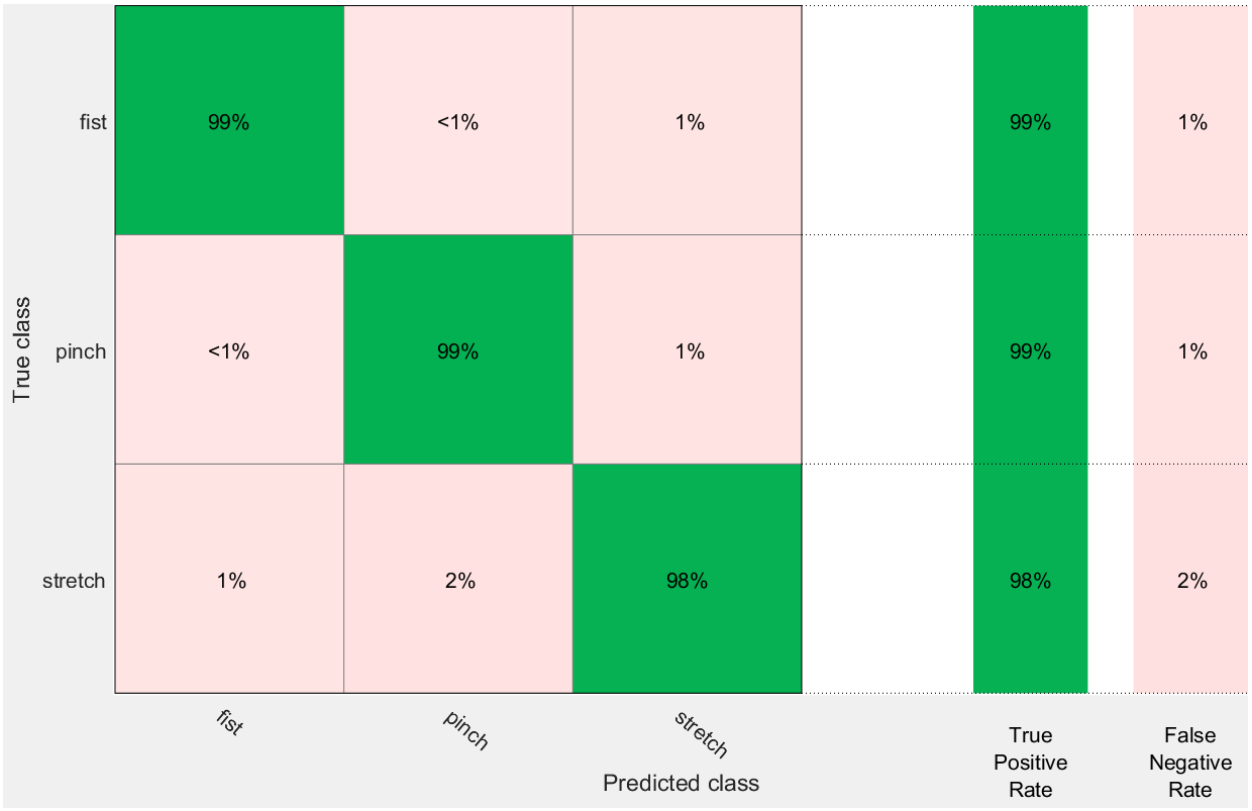


Figure 31. *SVM classification using cubic kernel*

4.10.6Validation

When using statistical methods for predicting values for new data, a validation method must be used to ensure that the model fits the data well. K-fold cross-validation is a method used for this where the data is randomly divided to k subsamples, equal in size. Initially, one subsample is used for testing the model and the other k-1 subsamples are used for training. Then, the process is repeated k times to check for all subsamples [30]. In this work, 5-fold and 10-fold cross validation were used to check for the model.

5 Results and Conclusions

Using relative coordinates of AR tags to allow the application of OMG with the free camera setup was tested in this work and proposed as a solution for enhancing OMG. The idea was tested using big markers on a sheet of paper and then on the forearm. When used on the forearm, smaller tags were needed, and it resulted in reduced accuracy of detection. However, smaller tags allow more features of the forearm to be analyzed.

5.1 Tags relative transformations

For each image, and in the case of detecting all tags, 28 relative coordinates are computed for the tags. As mentioned earlier, some coordinates provide good potential for linear separation while others are not. Moreover, some coordinates are more robust than others with camera orientation changes. Figures 32, 33 and 34 show some examples for this using T01x, T01y and T01z. On the x axis, the number of image is marked from 1 to 1500 where each set of 500 images represent an orientation. An ideal signal would be a periodic signal with same three distinctive values that correspond to three poses. In Figure 32, T01x signal has a repetitive pattern, but with an offset. The y-transformation in Figure 33 had a worse offset that completely gives a different range of numbers in different orientations. In Figure 34, T01z gives the most robust one among them.

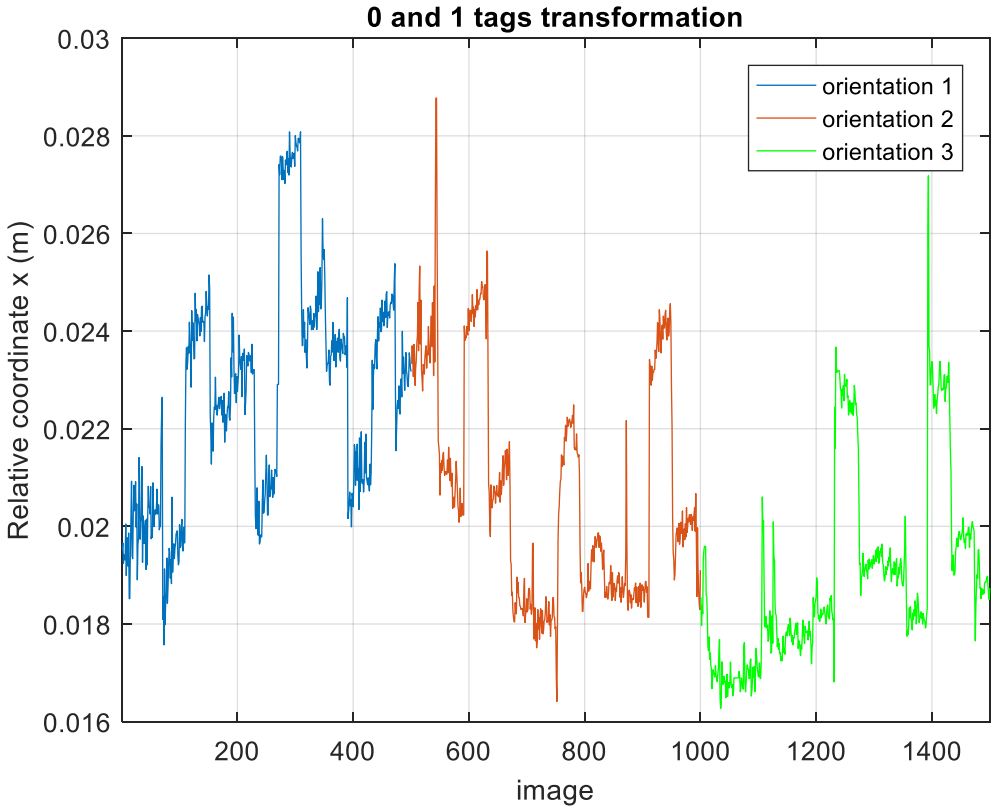


Figure 32. *T01x*

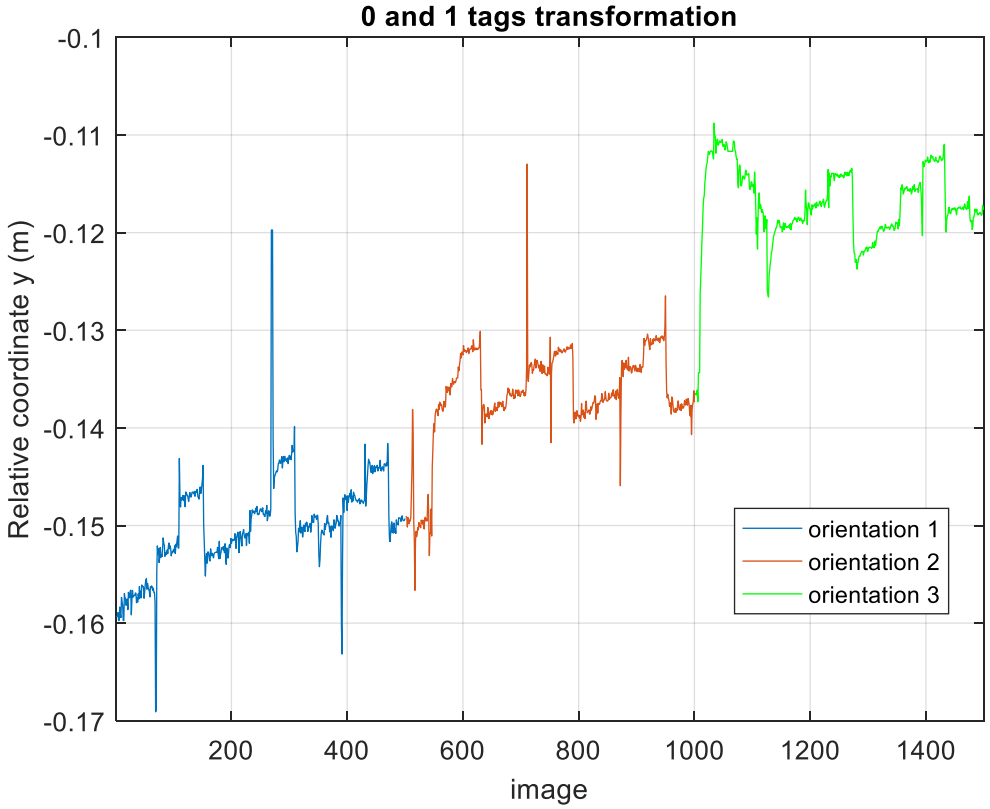


Figure 33. *T01y*

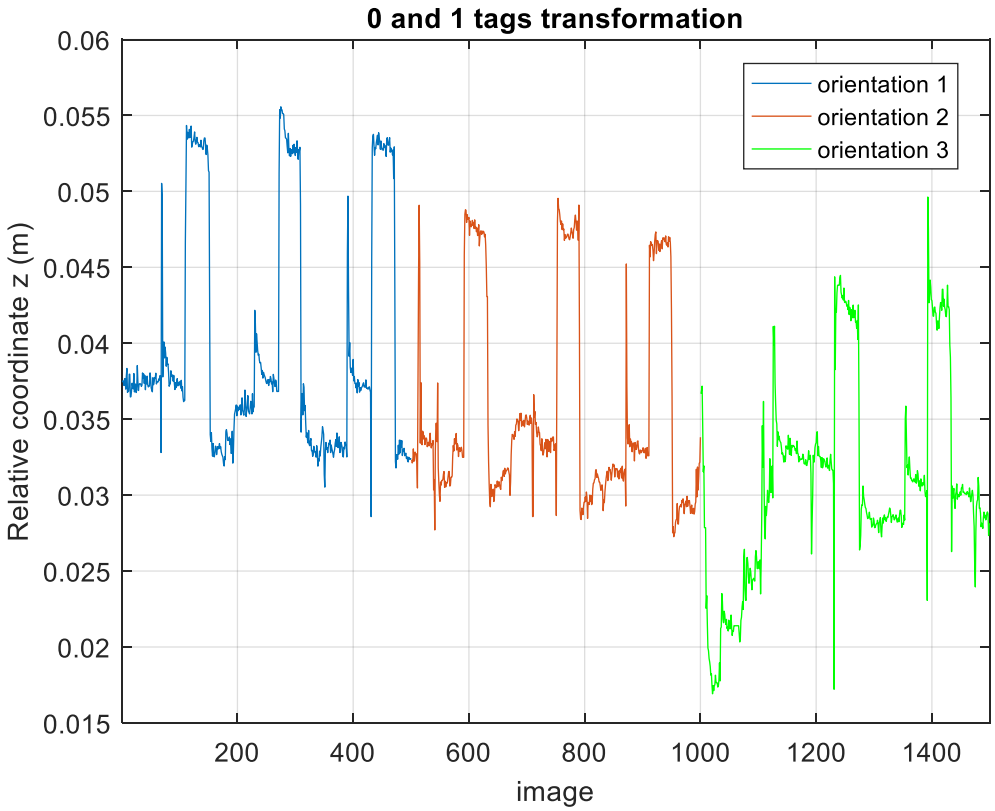


Figure 34. *T01z*

5.2 Machine learning results

Different number of features were tested against three machine learning algorithms. For the LDA, T02z was essential to obtain a reasonable accuracy. When combined with T03z as only two features, an accuracy of 90% was obtained. Slightly better accuracy (92%) was obtained with the addition of T01z and that was the best result with that algorithm. Any addition of features resulted in decreasing the accuracy. Moreover, in the case of having all transformations except tags 0-1 and 0-2, the accuracy level was 61%.

While using decision trees, the absence of specific transformations didn't affect the performance the way it did with LDA. Adding more features resulted in better accuracy and absence of certain tags did not result in a poor performance. However, not detecting tags 1 and 2 led to an accuracy level of 93% even with all other tags detected.

Using SVM, high accuracy levels were obtained, and it proved robust against the absence of any tag. The algorithm was tested in case of misdetection of one tag and two tags and accuracy didn't go below 97%.

Results are summarized in figure 32 where LDA_2 refers to the average value of using LDA with combinations of T01z, T02z, T03z and T04z as input features and LDA_1 otherwise. DT_1 refers to using Decision Trees while T02 is excluded from input features and DT_2 otherwise. SVM_1 refers to lowest value obtained with SVM and SVM_2 is the average value for using SVM.

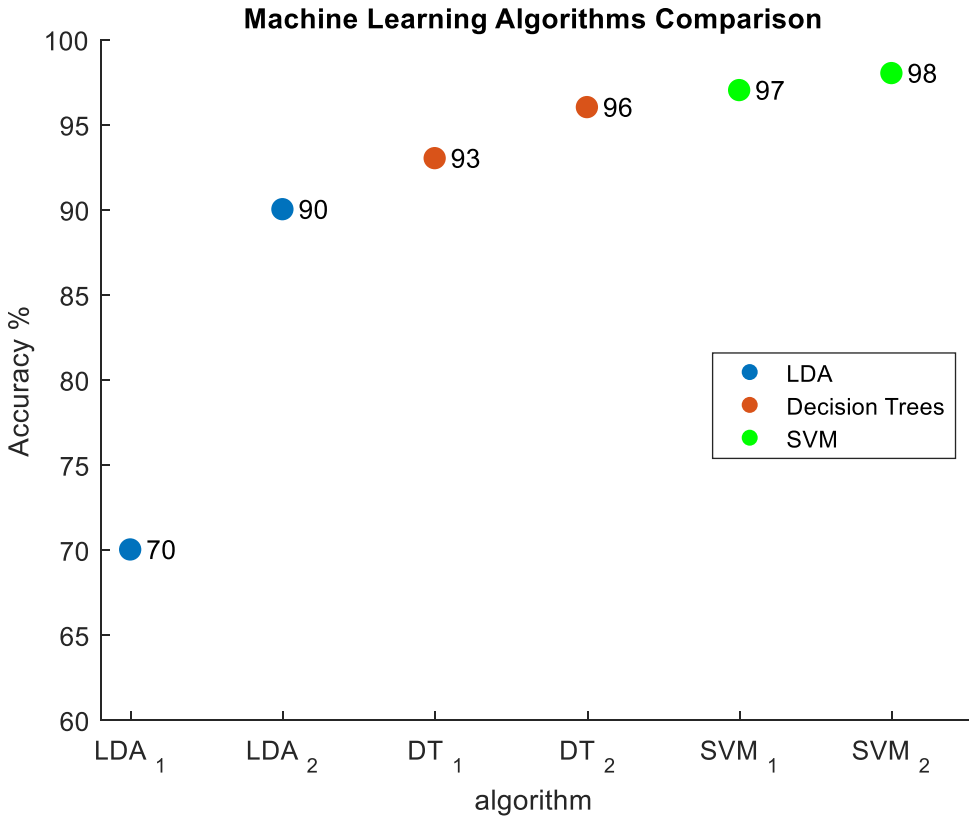


Figure 35. *Three Machine Learning Algorithms Performance Comparison*

LDA was critically dependent on tags 1 and 2 while decision trees were dependent on them to obtain a higher accuracy level. However, SVM does not depend on any particular tag(s) to give very good results.

Appendix 7.1 shows the performance of the three algorithms when tested in case of not detecting one tag, two tags or three tags respectively. Comparison shows that SVM is not affected by not detecting tags, even in case of missing three tags.

5.3 Further work

The presented work is part of an ongoing project at the DLR to enhance the OMG for better mobility. This work constitutes as a proof of concept for a method of using transformations of AR tags relative to each other and not to the camera. This is proposed as a solution to allow free placement of camera on arm and free movement of the arm. The next stage is to extend this study to larger set of human subjects at the DLR. For a wider scale user study, a stimulus software developed at DLR is used to ask the person for a certain pose and then capture the corresponding movement of forearm. As the software is coded in C#, the AR detection and coordinates computations code needs to be in C#. However, currently it is only available in C++ and Java. There exists one implementation in C# that can only detect the tags, but no further computations [31]. So, writing the code in C# has to be done to allow for the integration with the other components and further experiments.

6 References

[1] World Health Organization and The World Bank, "World Report on Disability," Malta, 2011.

[2] K. Ziegler-Graham, E. J. MacKenzie, P. L. Ephraim, T. G. Travison and R. Brookmeyer, "Estimating the Prevalence of Limb Loss in the United States: 2005 to 2050," *Archives of physical medicine and rehabilitation,* vol. 89, no. 3, pp. 422-429, 2008.

[3] F. Cordella, A. L. Ciancio, R. Sacchetti, A. Davalli, A. G. Cutti, E. Guglielmelli and L. Zollo, "Literature review on needs of upper limb prosthesis users," *Frontiers in neuroscience,* vol. 10, p. 209, 2016.

[4] L. Resnik, M. R. Meucci, S. Lieberman-Klinger, C. Fantini, D. L. Kelty, R. Disla and N. Sasson, "Advanced upper limb prosthetic devices: implications for upper limb prosthetic rehabilitation," *Archives of physical medicine and rehabilitation ,* vol. 93, no. 4, pp. 710-717, 2012.

[5] World Health Organization, "Guidelines for training personnel in developing countries for prosthetics and orthotics services," Geneva, 2004.

[6] D. Cummings, "Prosthetics in the developing world: a review of the literature," *Prosthetics and orthotics international,* vol. 20, no. 1, pp. 51-60, 1996.

[7] J. Borg, A. Lindström and S. Larsson, "Assistive technology in developing countries: national and international responsibilities to implement the Convention on the Rights of Persons with Disabilities," *The Lancet,* vol. 374, no. 9704, pp. 1863-1865, 2009.

[8] J. t. Kate, G. Smit and P. Breedveld, "3D-printed upper limb prostheses: a review," *Disability and Rehabilitation: Assistive Technology,* vol. 12, no. 3, pp. 300-314, 2017.

[9] "e-NABLE," [Online]. Available: www.enablingthefuture.org.

[10] M. Sullivan, B. Oh and I. Taylor, "3rd Printed Prosthetic Hand," Washington University Open Scholarship, 2017.

[11] P.-H. Wu and J.-S. Shieh, "3D Printed Prosthetic Hands," in *International Conference on communication Problem-Solving*, Taipei, Taiwan, 2016.

[12] S. Bitzer and P. v. d. Smagt, "Learning EMG control of a robotic hand; Towards Active Prostheses," in *International conference on Robotics and Automation*, Orlando, Florida, 2006.

[13] F. Tenore and A. Ramos, "Towards The Control of Individual Fingers of a Prosthetic Hand Using Surface EMG Signals," in *29th Annual International Conference of the IEEE EMBS*, Lyon, France, 2007.

[14] M. Kuttuva, G. Burdea, J. Flint and W. Craelius, "Manipulation practice for upper-limb amputees using virtual reality," *Presence: Teleoperators & Virtual Environments,* vol. 14, no. 2, pp. 175-182, 2005.

[15] N. Li, D. Yang, L. Jiang, H. Liu and H. Cai, "Combined use of FSR sensor array and SVM classifier for finger motion recognition based on pressure distribution map," *Journal of Bionic Engineering,* vol. 9, no. 1, pp. 39-47, 2012.

[16] D. Han, D. Kuschner and Y. Wang, "Upper limb position sensing: A machine vision approach," in *EMBS conference on Neural Engineering*, 2005.

[17] C. Nissler, N. Mouriki and C. Castellini, "Optical Myography: Detecting Finger Movements by looking at the Forearm," *Frontiers in Neurorobotics,* vol. 10, 2016.

[18] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *International Conference on Robotics and Auomation*, Shangahai, China, 2011.

[19] C. Nissler, C. Castellini and N. Navab, "Improving Optical Myography via convolutional neural networks," in *MEC17*, 2017.

[20] " AprilTags public svn repository from MIT," [Online]. Available: https://svn.csail.mit.edu/apriltags/.

[21] A. Kaehlr and G. Bradski, "Camera Models and Calibration," in *Learning OpenCV 3*, O'Reilly Media, 2016.

[22] R. Urtasun, "Computer Vision: Calibration and Reconstruction," February 2013. [Online]. Available: https://www.cs.toronto.edu/~urtasun/courses/CV/lecture09.pdf.

[23] "Camera Calibration and 3D Reconstruction," OpenCV documentation, [Online]. Available: https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html#solvepnp.

[24] "American Society for Surgery of the Hand," [Online]. Available: http://www.assh.org/handcare/Anatomy/Muscles.

[25] L. I. Smith, "A tutorial on Principal Components Analysis," February 2002. [Online]. Available: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf.

[26] "Regular Expressions," [Online]. Available: http://www.pnotepad.org/docs/search/regular_expressions/.

[27] S. Elhabian and A. Farag, "A Tutorial on Data Reduction Linear Discriminant Analysis (LDA)," September 2009. [Online]. Available: http://www.sci.utah.edu/~shireen/pdfs/tutorials/Elhabian_LDA09.pdf.

[28] C. Kingsford and S. L. Salzberg, "What are decision trees?," *Nature biotechnology,* vol. 26, no. 9, pp. 1011-1013, September 2008.

[29] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, "16.5. Support Vector Machines," in *Numerical recipes 3rd edition: The art of scientific computing*, Cambridge university press, 2007.

[30] J. Schneider, "Cross Validation," Carnegie Mellon University, [Online]. Available: http://www.cs.cmu.edu/~schneide/tut5/node42.html.

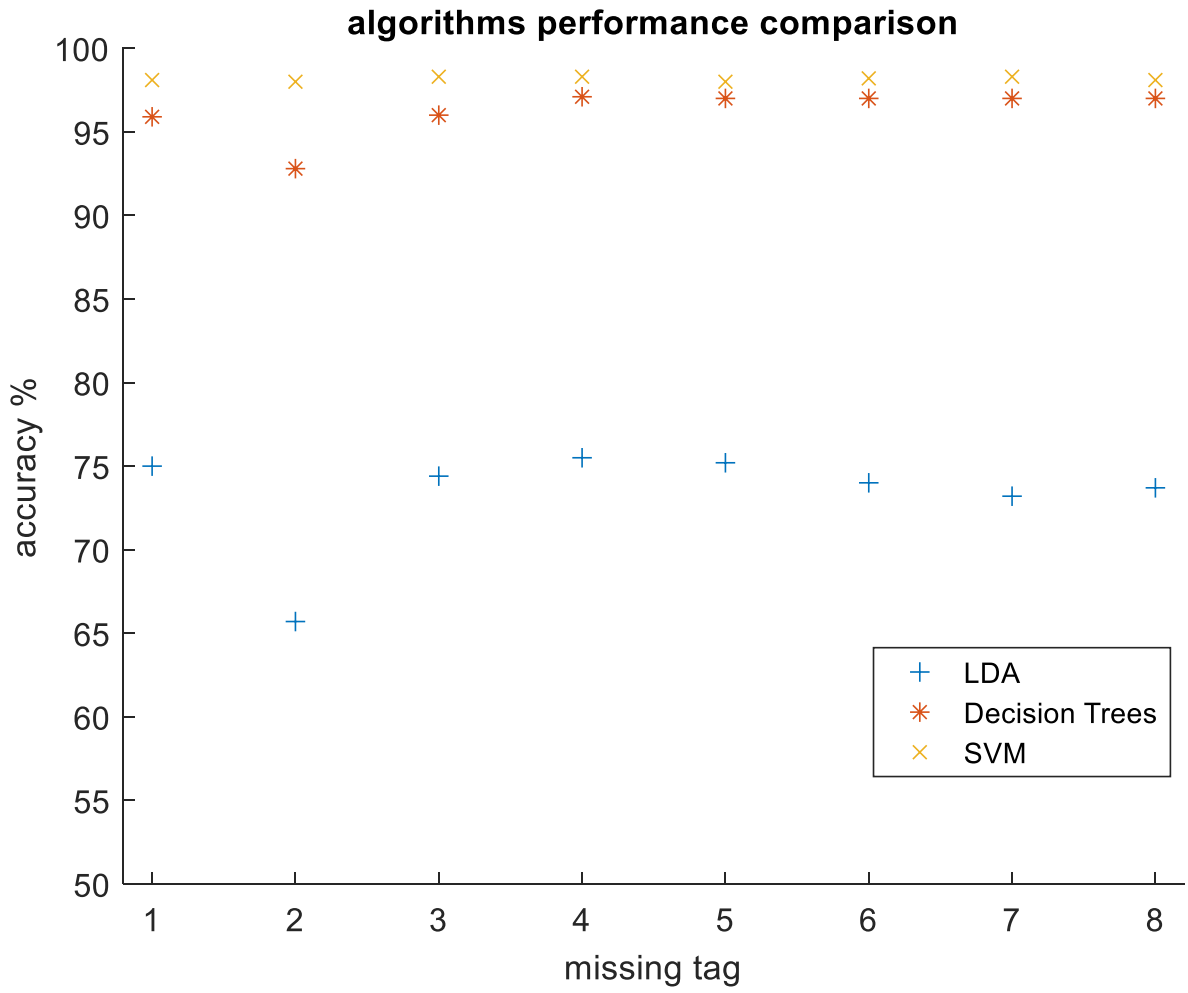
[31] "Apriltags csharp," [Online]. Available: https://github.com/BlackJocker1995/Apriltagcsharp.

7 Appendices

7.1 Machine learning algorithms performance comparison

The performance of the three algorithms used for classification is compared here. The first section shows accuracy when one tag is excluded from the input data to machine learning. Second and third sections do the same when two and three tags are excluded, respectively.

7.1.1 One missing tag



7.1.2 Two missing tags

Data in this section and the next one is organized into tables such that rows and columns headers represent the tags numbers and colors correspond to different algorithms. For example, if tag 1 and tag 2 are excluded accuracy is 61.3%, 93.5 and 97.65 for LDA, decision trees and SVM respectively.

	2	3	4	5	6	7	8
1	61.3	74.6	75.5	75.6	74.4	73.9	74.2
	93.5	95.6	95.8	95.8	96	96.1	95.8
	97.6	98.1	98	97.9	98	97.7	98.1
2		64.6	68.1	69.2	65.6	64.2	65.2
		93.3	92.9	94	92.8	92.7	93.5
		98	98.4	98.3	98	97.9	97.9
3			76.2	76	74.1	73.3	73.7
			96	96.1	95.9	96.3	96
			98.7	98.3	98.2	98.4	98.3
4				76.2	75.3	74.6	75.3
				97	97	97.1	97
				98.1	98.4	98.3	98.3
5					75	73.9	74.7
					97	97	96.9
					97.9	98.2	98
6						73.1	73.3
						97	97
						98.4	98.3
7							73.1
							96.7
							98.3

LDA **Decision Trees** **SVM**

7.1.3 Three missing tags

	3	4	5	6	7	8
1,2	59.1	65.1	64.7	60	59.1	59.8
	89.7	93.6	93	93.3	91.7	91
	97.2	97.6	97.5	97.4	97.4	97
1,3		76.5	76.5	73.6	72.6	73.5
		95.5	95.7	95.6	95.4	96
		97.9	97.8	97.8	97.7	98.2
1,4			76.6	75.4	74.7	75.2
			95.8	96	95.8	95.6
			97.9	97.8	97.8	98
1,5				75.4	74.6	75.2
				95.7	95.7	95.7
				97.9	97.6	97.9
1,6					73.8	73.9
					96.1	96
					97.8	97.8
1,7						73.5
						96.1
						97.7

	4	5	6	7	8
2,3	68.3	69.9	64	60.6	62.7
	93.1	94.4	93.3	93.1	92.2
	98.4	98.2	98.1	98	98
2,4		73.7	67.5	66.5	66.8
		94.2	93	93.1	93.3
		98.3	98.3	98.4	97.9
2,5			70	69.6	68.6
			94.1	94	94.5
			97.9	97.8	98.1
2,6				64.6	65
				92.7	93.5
				98	97.8
2,7					63.6
					93.6
					97.9

	5	6	7	8
3,4	77.7	75.5	74.6	75.5
	96.1	96	96.3	96
	98.7	98.4	98.7	98.4
3,5		75.6	74.4	75.4
		96	96.4	96.1
		98.1	98.4	98.2
3,6			72.9	73.5
			96.2	96
			98.2	98.4
3,7				72.9
				96
				98.3

	6	7	8
4,5	75.8	75.5	75.9
	97	97	97
	98.1	98.2	98
4,6		74.3	75.1
		97	97.2
		98.4	98.3
4,7			74.1
			96.7
			98.3

	7	8
5,6	73.6	74
	96.9	97
	98.1	97.8
5,7		73.3
		96.7
		98.2

6,7,8	73.2
	96.7
	97.9

LICENSE

Non-exclusive license to reproduce thesis and make thesis public

I, Hassan Mahmoud Shehawy Elhanash,

1. herewith grant the University of Tartu a free permit (non-exclusive license) to:

1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright of my thesis, Optical Tracking of the forearm for Classifying Fingers Poses, supervised by Karl Kruusamäe

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive license does not infringe the intellectual property

rights or rights arising from the Personal Data Protection Act.

Tartu, **20.05.2018**