KU LEUVEN

FACULTEIT INGENIEURSWETENSCHAPPEN

Motion Recognition in Human-Robot Interaction using Invariant Descriptors

Konstantinos Gyftodimos

Thesis submitted for the degree of Master of Science in Mechanical Engineering

Thesis supervisors:

Joris De Schutter
Erwin Aertbeliën

Assessors:

Gianni Borghesan
Sander Dedoncker

Mentor:

Maxim Vochten

Academic year 2020 – 2021

Preface

I would like to thank my promoter and my professors for the guidance and the assistance they provided me. I would also like to thank the jury for reading the text. My gratitude also goes to my family and friends for the invaluable mental support.

Konstantinos Gyftodimos

Contents

Abstract

A successful and efficient collaboration between humans and robots can provide a framework, in which many sectors of the industry and household environments will improve. Nowadays, in order for robots to execute flexible tasks, offer services and make the production simpler, yet, more efficient, their proper interaction with humans is of crucial importance. In this thesis work, a research to ameliorate the human-robot interaction with further focus on motion recognition, will be conducted.

In the present work, the motion of a rigid object is expressed in a way, in which it will hold certain invariant properties with respect to several contextual dependencies of the recorded motion. The primary goal of the thesis is to investigate and validate the invariant properties of the motions through the recognition approaches that are proposed, followed by the juxtaposition and ranking of all the possible invariant representation schemes, that are investigated, in terms of recognition accuracy. At the same time, an equally important objective, is to prove that the higher the level of invariance a motion representation has, the less training motion demonstrations are required in order for this motion to get recognized.

The motion recognition algorithms that are proposed, consist of a modified machine learning, distance-based, and a deep learning method. The amount of motions required to train the corresponding methods is kept minimal and equal for every approach, so that a comparative basis can be established, and the invariance of the investigated motion representations can be discussed.

List of Figures and Tables

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

FS	Frenet-Serret
ISA	Instantaneous Screw Axis
DTW	Dynamic Time Warping
DTW-KNN	Dynamic Time Warp-based K-Nearest Neighbors Classification
LSTM	Long-Short Term Memory Network
LED	Light-Emitting Diode
RGB	Red Green Blue
CSS	Curvature Scale Space
CDF	Centroid Distance Function
TSF	Time Series Forest
RISE	Random Interval Spectral Ensemble Classification
IFT	Interval Feature Transformation
PIP	Perceptually Important Points
DCM	Direction Cosine Matrix
FCN	Fully Connected Network-layer

Symbols

x, y, z	Cartesian coordinates
x_i, y_i, z_i	coordinates of chosen reference frame
x_c, y_c, z_c	coordinates of LED Marker
$c[k]$	centroid distance funtion of a point k
$\mathbf{p}$	position vector
$w.r.f$	world reference frame
obj	frame attached to rigid body
$\mathbf{R}$	DCM - Rotation Matrix
e_x, e_y, e_z	normal vectors
$\mathbf{T}$	pose matrix
t_p	pose-twist vector
ω	rotational velocity vector
v	translational velocity vector
$\omega_1, \omega_2, \omega_3, v_1, v_2, v_3$	ISA-descriptors
$i_{r1}, i_{r2}, i_{r3}, i_{t1}, i_{t2}, i_{t3}$	FS-descriptors
v_0	translational velocity of rigid object, expressed in world reference frame
$\xi(t)$	degree of advancement along a motion trajectory
Θ	total rotation of a motion trajectory
L	total execution length of the motion
X	data-set of recorded motions
X_{train}	training data-set
X_{test}	testing data-set
k	number of neighbors
X_i	input sequence in LSTM-network
O_i	output vector of LSTM-network
H_i	hidden layer of LSTM-network
N	non-linearity function
W_{xh}	input to layer, weight matrix
W_{hh}	layer to layer, weight matrix
b	bias term
M_i	motion index
PM_i	mredicted motion index
c_{jj}	cell index in the diagonal of a confusion matrix

Chapter 1

Introduction

1.1 Background and Motivation

Nowadays, there is a doubtless need and an increasing demand for robots in many sectors of the industry. The successful interaction of humans and robots can act as a successful tool for efficient production, in many fields such as in factory manufacturing lines, as well as in medicine and health care environments.

There is a trend from single mass-produced products to custom products/services. Therefore, robots will have to go from repetitive, fixed tasks to flexible, varied tasks. Programming a robot to do these varied tasks autonomously is very time-consuming. Humans are far better with doing varied tasks, so therefore human-robot collaboration can be very helpful, where the robot supports a human either cognitively and/or physically. Also in other environments (medical, household) besides industry, the robot has to move freely and interact with humans. All of these things bring challenges and a need for motion recognition to help in the interaction. To achieve a harmonious co-operation, it is important for the robot not only to understand the human motion by a raw input of a recorded motion, for example, but also to recognize similar motions, that are executed by a human, and be able to classify them.

In order for a robot to be able to understand human movement, the reformulation of this motion to the robot programming language of a motion trajectory, as shown in Figure 1.1, is mandatory. The reformulated model, should suffice to describe a specific motion class, such as wiping a table, under variable executions and real world contextual dependencies, such as the viewpoint from which the human motion is recorded, or the point on the object that is selected to be tracked during the motion demonstration. Additionally, the correct and precise recognition of these motions, through classification algorithms and other recognition techniques, can offer a robot the ability to predict future actions, and be able to understand and adjust its motion for its current activity, for a smooth and efficient collaboration with humans.

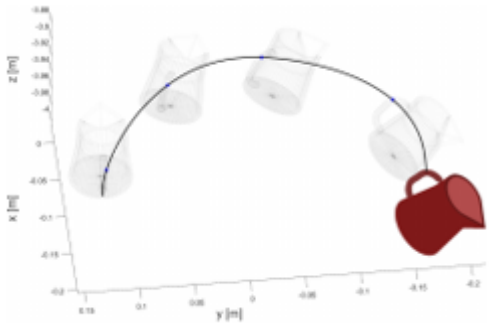


Figure 1.1: Motion of an object, illustrated from the view-point of programming language [18]

1.2 Objectives

The overall goal of the thesis is to achieve high recognition rates in various motion classes, through different classification techniques, using an alternative approach of demonstrating a motion in contrast with the traditional spatial trajectory approach that is mostly being used in the literature.

The representation of a motion of a rigid object, through descriptors that are invariant to contextual dependencies, such as the world reference frame from which the motion is recorded, the object's selected reference point, the velocity profile and the duration of a motion, through different calculation schemes will act as a basis for comparing their effect on recognition rate. Additionally, the construction of different classification algorithms and techniques will be used and compared for the same data-set of motions.

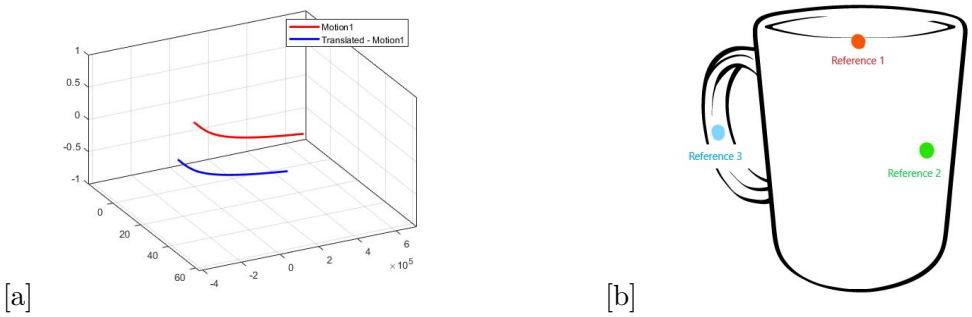
Another important objective of this thesis is to try to achieve the highest possible recognition rates, through variable algorithms, while reducing the need for large training data-sets.

1.3 Approach

In order to provide an invariant representation of a motion in terms of how the motion is being captured by the camera sensors, invariant motion descriptors of different types, schemes and ways of calculation will be used as a basis for the prediction of the corresponding motions.

1.3.1 Invariant Descriptors Approach

The most common approach to describe a motion of an object, is through representing its position at every time instant with Cartesian coordinates: x, y, z, and its orien-



[a] [b]

FIGURE 1.2: [a]. Spatial offset location of a motion. [b]. Different reference point choices on a rigid object.

tation by pitch, roll and yaw, both of which are expresses in a specific reference frame.

In the current thesis work, the representation of a trajectory of a motion will be referred to, as **invariants**, or **descriptors**. In mathematics, an invariant property of a variable or an object, corresponds to its independence on possible transformations, like for example: scaling, translation and rotation profile. The term invariant, in this study, refers to the fact that a rigid-body motion model does not depend to contextual dependencies, which include [18]:

- **World reference frame**: The choice of the frame, or the coordinate system in which the trajectory coordinates are recorded.
- **Spatial offset location**: The set of points in the defined coordinate system in which the motion takes place. For example, an offsetted motion can include a translation with respect to one or more axes (Figure 1.2.a).
- **References on the rigid body**: The choice of the points on the rigid object from which the motion will be recorded. There is an infinite number of points on a cup, for example, from which the position and the orientation of the cup can be found. (Figure 1.2.b).
- **Motion execution style**: Includes the various styles with which a motion is executed. To be more precise, the shaking of a cup, for example, can happen in a slow or a fast manner, with a small or a larger motion amplitude and different acceleration profiles.

To describe a certain motion, two types of invariant descriptors for a rigid body trajectory, that are constructed differently and have different invariant properties, are considered:

- **Frenet-Serret invariant descriptors.**
- **Instantaneous Screw Axis invariant descriptors.**

Also, different schemes of the previously mentioned invariant types will be considered. Each scheme corresponds to a different parameterization option for the motion trajectory, which include:

- **Time-based**: descriptors are a function of time.
- **Geometric**: descriptors are invariant to time.
- **Dimensionless Geometric**: descriptors are invariant to time, velocity profile and amplitude of motion.

To conclude, the invariant descriptors are constructed with three different calculation methods, which include analytical formulas, an optimized approach and discretized formulas, which will be explained in detail in the third chapter.

1.3.2 Motion Recognition Approaches

Following the the representation through invariant descriptors, comes the possible recognition of the corresponding motions by classification algorithms that were constructed, such as a small portion of the data is used as training data. The recognition approaches that are used to predict the classification accuracy of the motions are based on the philosophy of trying to recognize a motion-type in a more invariant way. The approaches are summarized below and will be furtherly elaborated in Chapter 4:

- **Dynamic Time Warp (DTW) - based, k-Nearest Neighbors Algorithm (DTW-KNN)**: KNN finds the distances between a sample of the data and all the examples in the data, selecting the specified number examples (k) closest to the sample, then votes for the most frequent label that has appeared. Normally, the data that is being used consists of two dimensional structures, containing information like the characteristics of a number of people, like height and age for example, while the corresponding labels consist of the people's names. In the current work, a three dimensional structure will be used to express a motion trajectory, while the labels will consist of the designated motion classes . Instead of using the traditional euclidean distance metric as an approach to the KNN classification algorithm, as it is common in literature, the DTW metric is used instead and is classified as a specific motion depending on the amount of nearest neighbors (motion classes) that have the closest DTW-distance over the length of each trial.
- **Long-Short Term Memory Network (LSTM)**: The third and final recognition approach that will be followed, is through an architecture of recurrent neural networks with feedback connections named LSTM-networks. An important property of LSTM, that makes is a suitable candidate for the current thesis work, is their ability to remember properties and store information of time sequences, while being trained.

1.4 Contributions

Through this thesis work, emerges the need to provide theoretical and applied contributions to the field of invariant trajectory representations of a rigid body object and enhance the knowledge between various motion recognition techniques and algorithms, while using a small amount of data to construct such algorithms, as well as maintaining invariance with respect to spatial dependencies. In following lines, the major contributions are demonstrated:

1. **The first contribution** includes the detailed comparison of both types of invariant descriptors, under all the parameterizations considered, while pointing out the characteristics of every invariant scheme and type of calculation. In parallel, information will be provided on how accurately a trajectory can be described depending on the type of the motion representation.

2. **The second contribution** is the juxtaposition of all the descriptor types and schemes over novel classification techniques for rigid body trajectories which include distance-based and deep learning approaches.

3. **The third contribution** relies on the invariance of the descriptors as far as motion classification is concerned. It is shown that for a descriptor that eliminates more dependencies than another less-invariant descriptor the recognition can be very high with a small amount of demonstrations in distance-based classification algorithms. Also, in a deep-learning approach, where usually the amount of demonstrations needed to train a classification model lies at about seventy percent **(75/100)** of the total dataset, only about eight percent **(8.3/100)** will be used, resulting in a palatable accuracy model.

1.5 Outline

The outline of the current work is presented succinctly as follows:

Chapter 2:
The second chapter provides information about related work, considering different invariant representations as well as different recognition approaches in the literature. Additionally, the pros and cons of these methods are being considered.

Chapter 3:
The third chapter describes the kinematics of rigid bodies and the derivation of the invariant descriptors under different parameterization types and calculation approaches.

Chapter 4:
The fourth chapter proposes a number of different classification techniques and algorithms that are being used for the recognition of motions described in an invariant way.

Chapter 5:
In the fifth chapter of the thesis, a comparison of different recognition approaches of invariant descriptors, as well as the recognition results will be presented and commented upon.

Chapter 6:
In the sixth and final chapter, general conclusions and a review of the contributions of the thesis are provided, as well as future work that can be conducted in order to continue and hopefully improve the results of the current study.

Chapter 2

Related Work and State-of-the-art

In the current chapter, the state of the art of the literature will be presented, regarding the representations of rigid body trajectory representation of motions and their corresponding recognition. The process of presenting the related work topic, will be executed in steps, starting from the traditional spatial trajectory description, followed by other invariant representations in academic research, while annotating their corresponding strengths and weaknesses. Then, other approaches of motion classification will be investigated and commented upon.

It is important to mention, before-hands, that an additional recognition approach that is already being used in the "invariant trajectory descriptor" - field of study, and inspired this thesis, is through the dynamic time warp algorithm, which will be represented in the first subsection of the fourth chapter, where the recognition approaches are discussed, and not in the present chapter.

2.1 Trajectory description of a rigid body object

In order to create a human-robot skill transfer framework, it is a challenge to capture the movement of a rigid object in space. There are a number of approaches that are used to achieve robust trajectory estimation and in the following sub-chapters the most influential will be discussed from the current work's point of view.

2.1.1 Trajectory estimation in real 3-D coordinates

A trajectory estimation perspective through recording and then reconstructing the motion of a rigid body, is elaborated in the current approach. The concept behind this method, begins with attaching multiple LED-markers on the rigid body that executes the trajectory. The LED-device consists of twenty (20) high intensity LED's on a recorder with multiple surfaces that provide position data from the time the motion starts until the time it ends. The high intensity of the LED markers provide

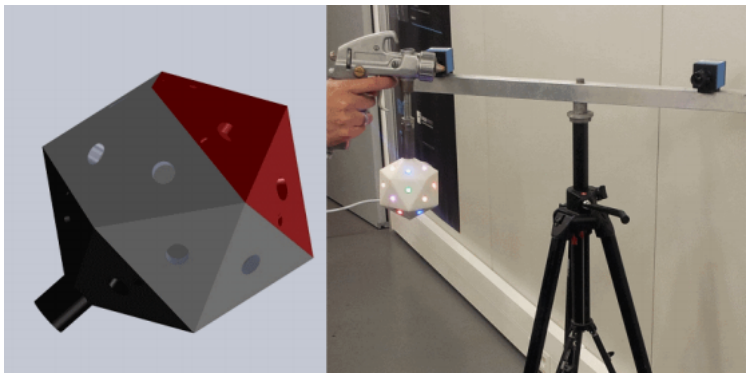


Figure 2.1: Illustration of the icosahedron LED-marker structure and actual mounting on a spray gun during the recording of the motion [10]

a very bright environment while the cameras record the motion which causes the back-ground noisy data to not have time to be acquired by the camera. In this way, the triangulation of the marker positions becomes more robust and the actual motion trajectory can be reconstructed without unwanted deviations and noise input[10].

As shown in Figure 2.1, the RGB-LED structure is mounted on a painting spray gun which executes the motion. At the exact moment that the camera setup starts recording the motion, the light emmiters on the icosahedron turn on and then off when the camera exposure time has expired to prevent the noise from being included in the recording of the motion.

In order to extract the position and the orientation of the rigid object (spray gun), a least squares estimation is used between the world reference which coincides with the camera center and the LED markers on the object, as shown in (**Equation 2.1**), where: x_c, y_c, z_c are the Cartesian coordiantes of the chosen world reference, x_i, y_i, z_i are the coordinates of the LED marker. The objective is to minimize the referred equation under several constraints.

$$(x_i - x_c)^2 + (y_i - y_c)^2 + (z_i - z_c)^2 - r^2 = 0 \tag{2.1}$$

The orientation estimation is calculated with the Kabsch algorithm.

This approach includes very useful assets for future work and has important advantages such as:

- Invariance of the motion recording to lightning conditions, due to the high intensity of the LEDs and the synchronized camera capture.
- Human to robot motion demonstration without the need of external software or any other tool.

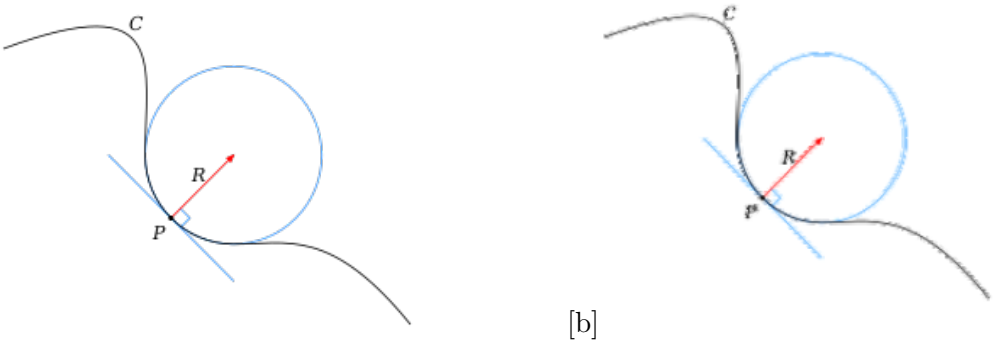
On the other hand, compared to the current approach that is used, it has the following deficiencies:

- The trajectory generation depends on the position of the camera setup (world reference frame), so as cameras placement should be very precise.
- Any possible variations in the speed and acceleration of the recorded motion may not be captured in a robust way.

2.1.2 View invariant motion trajectory

Another interesting and novel representation of a motion trajectory took place in the University of Illinois. According to [2], a view-invariant representation of a trajectory is achieved by two separate affine independent methods.

In the first viewpoint of this research, the representation of curvature scale space (CSS) is used. In the CSS representation, the shape of a parameterized trajectory is examined under different levels of smoothness, ranging from a fine to a coarse and noisy signal. Additionally, the curvature of data the parametric trajectory curve is extracted and then represented as an evolution of curvature with respect to increasing levels of smoothness. It is worth to mention that the curvature that is calculated at each point of the trajectory is a local descriptor of the point, thus with this method, invariance is achieved with respect to translation and rotation of the trajectory. An example of the curvature that is estimated under different levels of smoothness is illustrated in Figure 2.2 in a two dimensional case.



[a] [b]

FIGURE 2.2: [a]. Curvature of a point in a one dimensional case with a high smoothness level. [b]. Same configuration under noisy smoothness level.

In the second viewpoint, the representation of centroid distance function (CDF) is used. The motion trajectory is then represented by a discrete function (c[k]) of the distance of every point (k) in the curve from the centroid of the curve. The centroid of the curve can be defined as the mean position of all the points in the trajectory, or else, as an imaginary center of gravity of the curve:

$$c[k] = \sqrt{(x[k] - x_c)^2 + (y[k] - y_c)^2 + (z[k] - z_c)^2} \tag{2.2}$$

While both these methods offer a vast contribution to the trajectory representation field, it is important to mention that there are some limitations one should take into account. In the CSS method, where curvature is calculated in different smoothness schemes at each discrete point of the trajectory, a lot of information is lost in-between the points, despite the robust calculation of the curvature at the points themselves and a possible under-sampling of the points can play a crucial role in the final represented trajectory, as well as in the CDF approach due to its discrete nature. Another limitation in the CDF approach despite its invariance characteristics, is that the centroid of every motion that is demonstrated has to be calculated each time, in order for the trajectory to be reconstructed.

2.2 Motion Classification Approaches in Trajectories Representations

In the present content, some of the recognition algorithms and techniques that are used to classify a motion, the representation of which is similar to the presently used, will be demonstrated. After a brief overview of the methods of the ways that are used to recognize motions in a similar manner to this thesis, some will be highlighted in two sub-chapters, due to their methodological resemblance with the approaches that will be discussed in the fourth chapter.

The present problem of classifying a certain trajectory to its corresponding class, can be modelled as a time-series recognition problem and as a result, inspiration was drawn from the following recognition approaches in the literature:

1. **Shapelet-Based Classification:** In this approach, shapelets, or in other words, segments of the input trajectory, provide localized information about every segment. Such information, can be make the classification process easier, since trajectories that belong to the same class, have more or less same local characteristics in some of their corresponding segments. Each shapelet (Figure 2.3) is denoted as an interval of the trajectory. The shapelet-based local features, are used into a shapelet-based classifier [1]. Apart from the capability of this method for multi-class classification, the current approach provides invariance with respect to the length of the motion, since for example a motion class like: human walking, has similar local characteristics over its intervals, for varying, walking, periods of time.

2. **Shapelet Transform Classification:** Although similar with the previous method, the algorithm in this case identifies the top k-shapelets in the data-set. Then, k-features for the new data-set are calculated. Every single k-feature is found as the distance of the series to each one of the k-shapelets. An important

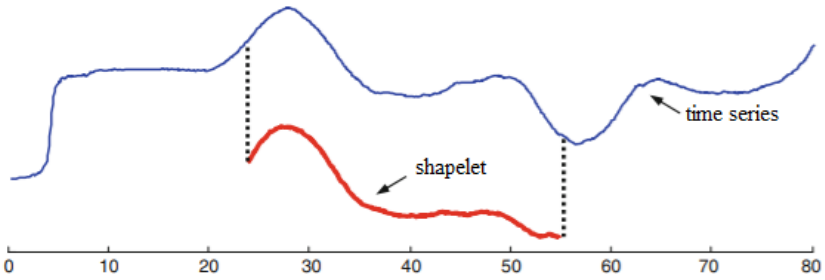


FIGURE 2.3: Shapelet representation (red) in an interval of the time series (blue).[1]

advantage of this extended shapelet approach, is that since every k-feature is stored in a part of a vector structure, any vector-based classification algorithm can be applied to the transformed data-set[4].

3. **Time Series Forest Classification (TSF):** The process behind this approach starts by splitting a time-series trajectory into random segments, with random lengths and starting positions. Then information like mean, standard deviation and slope are extracted from each segment into a vector of features, and then the training of a decision tree takes place based on the extracted features. For a new data-set of time-series, the determination of the corresponding class will be achieved through the number of trees in the forest[7].

4. **Random Interval Spectral Ensemble Classification (RISE):** This recognition approach can be described as an alternative adaptation of the **TSF** algorithm. Sometimes due to the large lengths of the input time-series like in a voice sample or in a large, in terms of spatial length, rigid body motion, can make **TSF** a very slow and unreliable recognition technique. This problem is dealt with the transformation of features into the Fourier domain[11].

The first approach that is explained in more detail, is through interval feature transformation for time series classification, using important trajectory points of a time-sequence and then a deep learning approach, where raw spatial trajectory data are converted into an image structure. The reason that these approaches were studied and chosen to be elaborated further from the literature, is the similar context of the recognition methods that are used in the present thesis work.

2.2.1 Interval feature classification of time series

A very important and novel recognition method which applies to time series, and as an extension to a rigid body trajectory is through interval feature transformation (IFT)[20]. In this approach, instead of trying to recognize a motion by comparing points from a time series sample to the next one, features of the trajectories in specific time intervals are compared between the two samples.

The interesting features in a trajectory are represented by critical points called perceptually important points (PIPs). By identifying PIPs through PIPs algorithm [5] it is easier to segment a time-series trajectory in smaller sub-trajectories that contain important information for their recognition. An example of these points in a time series generated by PIPs algorithm is illustrated in Figure 2.4.

In the next step, the best k-distinguishable feature vectors, where k is a scalar chosen from the user, are extracted and compared between all samples via various algorithms for time-series classification with robust results but at the cost of very big computational efficiency due to the amount of features.

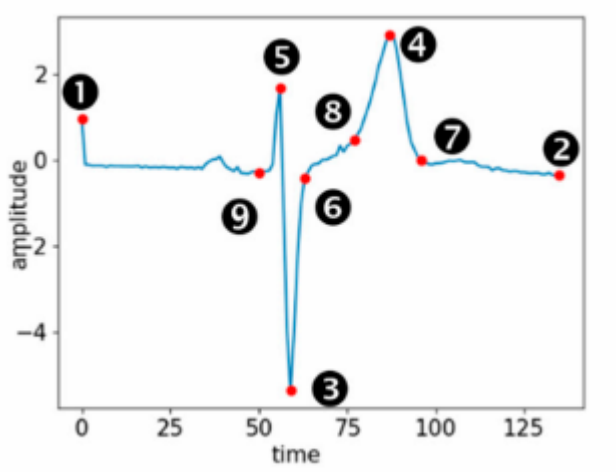


Figure 2.4: Time series segments seperated by PIPs after a few iterations of the algorithm. [20]

2.2.2 Spatial trajectory recognition via representation learning

The general concept behind this classification approach, lies in the transformation of a spatial trajectory that is dependent on time, to an image, which is segmented into weighted sub-image parts which contain information about characteristics and features of the corresponding part of the trajectory. The recognition process, then takes place through a pre-trained feed-forward neural network. The procedure in which the data set of trajectories is manipulated in order to be transformed to an image representation is illustrated in Figure 2.5 and is explained as follows[9]:

1. All input trajectories are scaled and fixed in a constant time interval which is represented by an image box.

2. The trajectories are segmented in time steps, illustrated by the dashed lines in (Figure 2.4).

3. Then the image box is divided into smaller sub-images, each of which contains a number of dashed lines. The orientation and the number of dashed lines in a sub-image, contains useful information about the class that the primary motion corresponds to, such as spatial translation, velocity and acceleration profile.

4. Finally, a constant weight is applied to all the sub-images, that corresponds to the number of dashed lines in the local region.

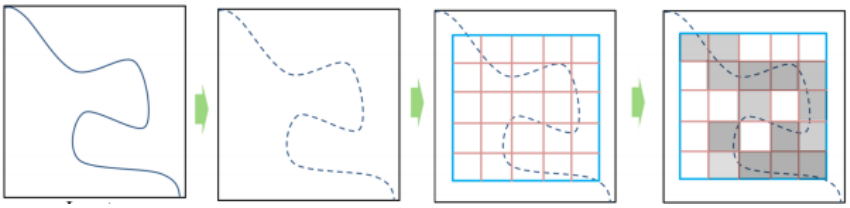


FIGURE 2.5: Procedure of converting a spatial trajectory to an image representation, as input for a learning based classification approach. [9]

The pre-processing of the classification which was described, can offer robustness in the recognition procedure. To begin with, by scaling and fixing all the trajectories from all the possible motion classes, in a certain time interval offers some invariance itself, since the duration of the movement becomes of low importance. Secondly, through the division of the trajectory into sub-images, the execution style information (velocity and acceleration profile) of every motion is included into each sub-image, which can, for example, include information about how fast, the motion is executed. On the other hand, although this method can apply for a rigid body trajectory in two dimensions, it may be much harder to implement in three directions, in which case the classification cannot happen through a pre-trained neural network but instead a convolutional network could be constructed, which then has to be trained with a very big amount of the dataset of motions as a training set.

2.3 Conclusion

The methods that are used for motion recognition in the current chapter were fundamental for the understanding of recognition in general. A study of the literature is a standard practice for an academic work. In the fourth chapter, two approaches for motion recognition will be followed and elaborated, a distance based and a deep learning approach respectively, which come into coalition with the methods discussed in this chapter.

Chapter 3

Invariant Representation of Rigid Body Motion

In the present chapter, the main concept behind how a trajectory of a rigid body can be modelled through invariant descriptors of different types, will be explained. Firstly, an overview of basic kinematics of rigid bodies will be provided, followed by the detailed elaboration of the two types of invariant representations that are used in this thesis. Then, a sub-chapter will be devoted to explain and compare the different parameterizations of the pre-referred types of invariant representations. Finally, different calculation approaches will be discussed and compared in a similar manner. It is noteworthy that the explanation of the invariant motion representations of a rigid body object, is the starting point, based on which the contributions in the next chapters are established.

3.1 Position and Orientation of Rigid Bodies

The term "**rigid body**" is used to depict a solid body whose deformation can be zero or very limited so as it can be neglected. Through attaching a three-dimensional frame to a rigid body object, its position and orientation can be determined with respect to the world reference frame.

The position of an object at a specific time moment, can be expressed through a vector in three-dimensional space: ${}_{w.r.f}\mathbf{p}^{w.r.f,obj}$. First, the $(w.r.f)$ sub-script shows the world reference frame in which the coordinates of the object are expressed. Additionally, the $(w.r.f, obj)$ super-script, corresponds to the axis that is on the object itself[17]. For example, in Figure 3.1, the position of a designated point on a rigid body, on which a reference frame (obj) is attached, can be described with the vector (p) with respect to the world reference frame $(w.r.f)$.

In three dimensions, the orientation of an object is not as simple as translation to describe. There are a lot of ways in which a rigid body can rotate as it can pitch,

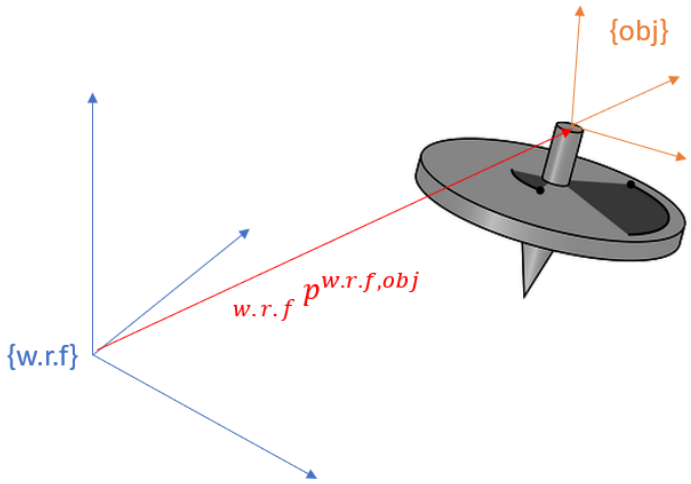


Figure 3.1: Position vector of a rigid body object with respect to world reference.

roll and yaw in two directions. There are various representations of the orientation of an object among which, quaternions, Euler angles, axis-angles representations and direction cosine matrices **(DCM)**, are the most dominant. The latest approach, which is used in this thesis, is basically a transformation matrix that transforms one coordinate reference frame to another. The DCM is a 3x3 matrix that includes the unit vectors of the reference frame that is attached on the rigid body object (obj) and is denoted by: ${}^{obj}_{w.r.f}\mathbf{R}$, where:

$$ {}^{obj}_{w.r.f}\mathbf{R} = [e_x \quad e_y \quad e_z] \tag{3.1} $$

Eventually, the position and the orientation of a designated point on a rigid body object can be described by combining the translational vector and the DCM into the pose matrix: ${}^{obj}_{w.r.f}\mathbf{T}$.

$$ {}^{obj}_{w.r.f}\mathbf{T} = \begin{bmatrix} {}^{obj}_{w.r.f}\mathbf{R} & {}_{w.r.f}\mathbf{p}^{w.r.f,obj} \\ 0_{1x3} & 1 \end{bmatrix} \tag{3.2} $$

To conclude, the pose matrix, which represents the combination of position and orientation of a reference frame attached to a rigid body, is a 4x4 matrix that includes the DCM which is a 3x3 matrix, the position vector, which is 3x1-dimensional vector, and a 1x3 zero vector.

3.2 Velocity of Rigid Bodies

The spatial velocity of a rigid body object, or else **twist**, can be derived from its **pose** and represents the translational velocity along an axis and the rotational velocity around an axis of an object into a common six-dimensional vector. Although the translational velocity is the same for every point of an object along the axis, the rotational velocity varies, depending on the chosen reference on the object itself. There are various type of twists, although the ones that will be discussed due to their connection with the thesis context is the pose-twist and the screw-twist.

3.2.1 Pose - Twist

Half the components of the **pose-twist** vector (t_p), consist of the translational velocity that is the rate of change of the "x", "y" and "z" components of a point in space and it is defined with respect to the world reference frame ($w.r.f$). The time derivative of the position vector (${}_{w.r.f}\dot{p}^{w.r.f,obj}$), is sufficient to describe it.

The expression for the rotational velocity (${}^{obj}_{w.r.f}\dot{R}$), in contrast, is not the direct time-derivative of the rotation matrix (${}^{obj}_{w.r.f}R$) that was discussed earlier. Instead the expression for the rotational velocity, requires the multiplication of the rotation matrix with a symmetric matrix, as shown in the following expression:

$$ {}^{obj}_{w.r.f}\dot{R} = \begin{bmatrix} 0 & \omega_z & -\omega_y \\ -\omega_z & 0 & \omega_x \\ \omega_y & -\omega_x & 0 \end{bmatrix} {}^{obj}_{w.r.f}R \tag{3.3} $$

In equation (3.3), the ω_x, ω_y, ω_z components represent the rotational velocity around the corresponding axis and form the form the rigid body rotational velocity vector: ${}_{w.r.f}\omega$. The six-dimensional pose-twist vector is expressed as follows:

$$ t_p = \begin{pmatrix} {}_{w.r.f}\omega \\ \\ {}_{w.r.f}\dot{p}^{w.r.f,obj} \end{pmatrix} \tag{3.4} $$

3.2.2 Screw - Twist

A second way in which the rotational and translational velocity of a rigid body object can be described is through the **screw-twist** vector. In this case, the following assumptions should hold. The reference chosen point on the object, coincides with the origin of the world reference frame ($w.r.f$). If this point is assumed to to be attached to the body, it genuinely obtains a certain velocity: v_0. In other words, when comparing the screw-twist with the pose-twist only the translational velocity

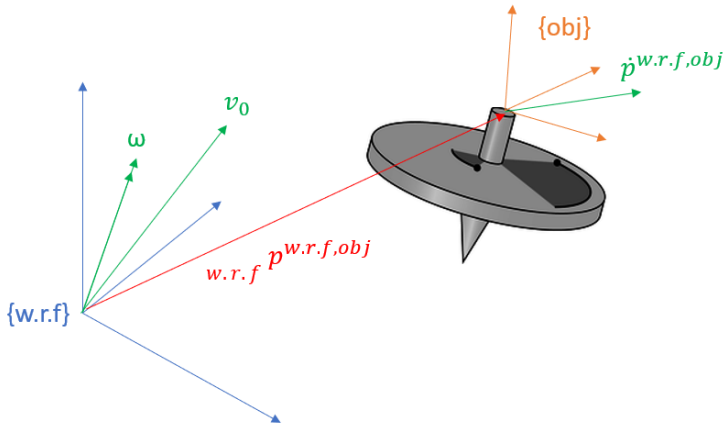


Figure 3.2: Translational and rotational velocity of a rigid body object according to the screw-twist, with respect to world reference frame.

components change inside the vector, while the rotational velocity components remain the same:

$$t_s = \begin{pmatrix} {}^{w.r.f}\omega \\ \\ v_0 \end{pmatrix} \tag{3.5}$$

To be more coherent, an illustration of the representation of both the translational and rotational velocity components of a rigid body can be observed in Figure 3.3, in which the green color represents the velocities.

3.3 Invariant Trajectory Descriptions of a Rigid Body

In the present sub-chapter, two different interpretations of trajectory description will be elaborated, the derivation of which is closely related to the body kinematics that were explained in the previous sub-chapter. First, the **Instantaneous Screw-Axis invariant descriptors** for rigid body motion will be introduced. Then, **Frenet-Serret invariants** for point motion will be explained, and their extended form which concerns rigid body motion. These kind of representations provide invariance of contextual dependencies, depending on how they are parameterized, but this will be furtherly explained in the next sub-section.

3.3.1 Instantaneous Screw Axis Invariant Representation for Rigid Body Motion

Based on Chasles' Theorem, the position and the orientation of a rigid body object can be completely described by the motion of an imaginary axis in space, the **instantaneous screw axis (ISA)**. In order for the motion of ISA to be able to describe a rigid body motion, the body itself should also rotate, otherwise for an object moving in a straight line the ISA will not be defined in a unique way and the motion description, will then become ill-posed.

The rotation and the translation of the instantaneous screw axis (ISA), can be entirely represented by six invariant descriptors [6], which consist of three (3) translational velocity and three (3) rotational velocity descriptors, respectively: $v_1, v_2, v_3, \omega_1, \omega_2, \omega_3$, as presented in Figure 3.3, where the trajectory motion of a spinning-top is uniquely defined at each time instant by the six (6) ISA - invariant descriptors on the instantaneous screw axis.

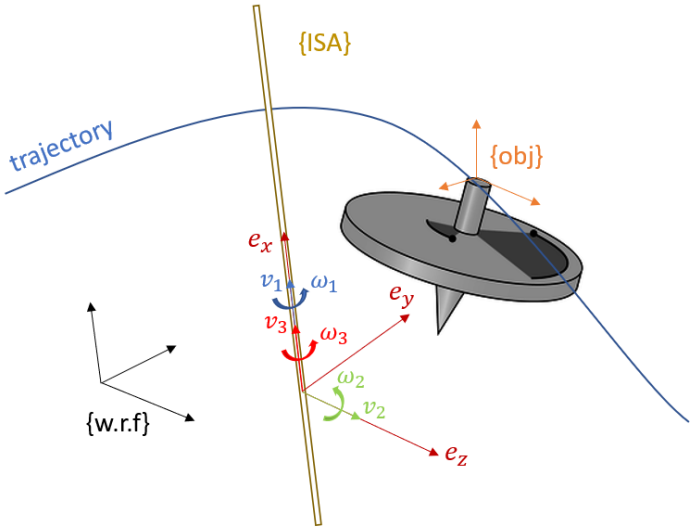


FIGURE 3.3: Description of rigid body motions, with six invariant descriptors on an instantaneous screw axis (ISA).

Each one of the six (6) descriptors, are obtained from the screw-twist representation, in (**Equation 3.5**), where the rotational part: ${}_{w.r.f}\omega$, which from now will be denoted as: ω, corresponds to the three (3) rotational invariant descriptors: $\omega_1, \omega_2, \omega_3$ and the translational part of the vector: v_0, which from now will be denoted as: v, corresponds to the three (3) translational invariant descriptors: v_1, v_2, v_3. Every descriptor's value is different during the evolution of the trajectory of the spinning-top, so at each time instance the invariants from now on, can be also referred to as: $v(t)$

and $\omega(t)$, with units: "m/s" and "rad/s" respectively. Due to to their dependence on time, these invariant descriptors will be also referred as: **time-based invariant descriptors**. Finally their time derivatives will be annotated with a number of dots over each velocity, depending on the order of the time derivative.

Before getting into detail into the physical meaning and the derivation of the descriptors, it is mandatory to express how the instantaneous screw axis (ISA) is defined. Three (3) normal vectors: e_x, e_y, e_z, are calculated and together define the orientation of the ISA.

The first normal vector: e_x, is derived directly from the rotational velocity as follows:

$$e_x = \frac{\omega}{||\omega||} \tag{3.6}$$

The second normal vector: e_y, is orthogonal to the first, and is calculated as:

$$e_y = \frac{\omega \times \dot{\omega}}{||\omega \times \dot{\omega}||} \tag{3.7}$$

The third normal vector: e_z, can be obtained directly from the two previous axes:

$$e_z = e_x \times e_y \tag{3.8}$$

The first two invariant descriptors v_1 and ω_1 represent the translational velocity of the rigid body along the ISA-axis and its rotational velocity around it, respectivelly as the rigid object moves in space, as shown in Figure 3.4 and are calculated in the equations below:

$$\omega_1 = ||\omega|| \tag{3.9}$$

$$v_1 = \frac{v\,\omega}{||\omega||} \tag{3.10}$$

The last four invariant descriptors $v_2, v_3, \omega_2, \omega_3$, which are illustrated in Figure 3.5, represent the position and orientation of the ISA itself and not of the rigid body directly. To be more precise, the v_2 and ω_2 invariants represent the motion between the common normal of two successive ISA-axes in two consecutive moments.

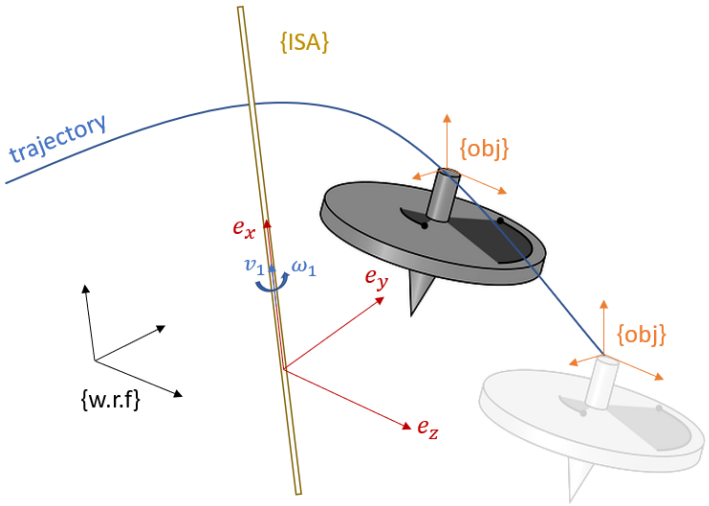


FIGURE 3.4: Representation of the rigid body motion by the first two invariants: v_1 and ω_1

Additionally, the v_3 and ω_3 invariants constitute the rotation and translation along the ISA-axis, which is necessary for moving the common normal. In this way at each time moment, through an axis with six (6) descriptors, the derivation of the position and orientation of a rigid body object is achieved. Below, these formulas of the four invariant descriptors are presented:

$$\omega_2 = \frac{\omega \times \dot{\omega}}{||\omega||^2} \tag{3.11}$$

$$v_2 = \frac{\omega \times \dot{\omega}}{||\omega \times \dot{\omega}||} \frac{(\dot{\omega} \times v + \omega \times \dot{v}) \cdot ||\omega||^2 - 2(\omega \times v)(\omega \cdot \dot{\omega})}{||\omega||^4} \tag{3.12}$$

$$\omega_3 = \frac{((\omega \times \dot{\omega}) \times (\omega \times \ddot{\omega})) \cdot \omega}{||\omega \times \dot{\omega}||^2 \cdot ||\omega||} \tag{3.13}$$

$$v_3 = -\frac{[\dot{\omega} \times (\omega \times \dot{\omega}) + \omega \times (\omega \times \ddot{\omega})] \cdot [||\omega||^2 \cdot (\dot{\omega} \times v + \omega \times \dot{v}) - 2\omega\dot{\omega}(\omega \times v)]}{||\omega||^2 \cdot ||\omega \times \dot{\omega}||^2} \tag{3.14}$$

$$-\frac{[\dot{\omega} \times (\omega \times \dot{\omega})] \cdot [||\omega||^2 \cdot (\ddot{\omega} \times v + 2\dot{\omega} \times \dot{v} + \omega \times \ddot{v}) - 2(||\dot{\omega}||^2 + \omega \cdot \dot{\omega})(\omega \times v)]}{||\omega||^3 \cdot ||\omega \times \dot{\omega}||^2}$$

$$+[\frac{3}{2} \cdot \frac{\omega \cdot \dot{\omega}}{||\omega||^2} + \frac{(\omega \times \dot{\omega}) \cdot (\omega \times \ddot{\omega})}{||\omega \times \dot{\omega}||^2}] \cdot \frac{(\omega \times (\omega \times \dot{\omega})) \cdot ||\omega||^2 \cdot (\dot{\omega} \times v - \omega \times \dot{v}) - 2(\omega \cdot \dot{\omega})(\omega \times v)}{||\omega||^3 \cdot ||\omega \times \dot{\omega}||^2}$$

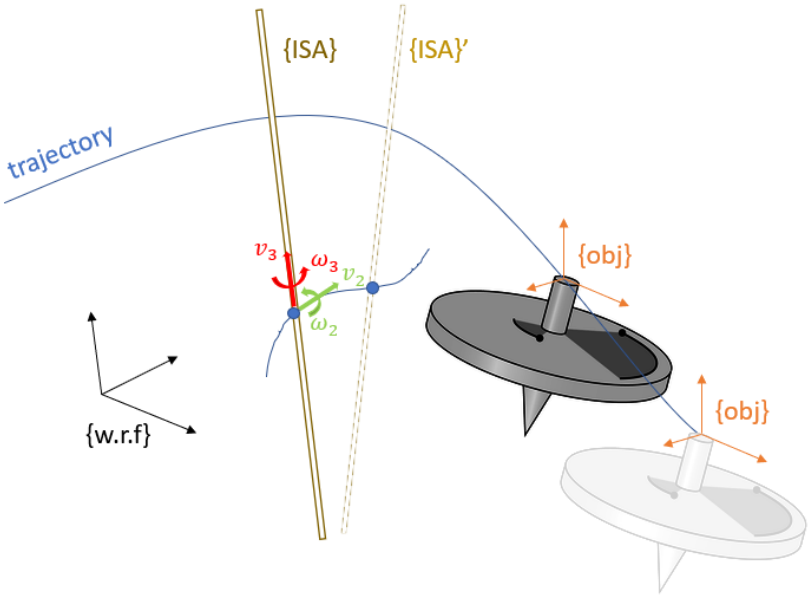


Figure 3.5: Representation of the motion of the instantaneous screw axis (ISA), between two successive time instants.

3.3.2 Frenet-Serret Invariant Representation for Point Motion

Before expressing the trajectory representation of a rigid body with **Frenet-Serret invariant descriptors (FS)**, it is necessary to explain the how the FS-descriptors for a point's translational motion, with no dimensions, work.

In order to describe the trajectory of a point via Frenet-Serret descriptors, the first step is to extract its position in the three-dimensional space, with respect to the world reference frame, through a "3x1" vector: $p(t)$, which includes the corresponding spatial information. At the same time a frame is attached to the moving point in space, called the **Frenet-Serret moving frame** as explained in [3]. Then on this moving vector in space, three FS-descriptors: "$i_1(t), i_2(t), i_3(t)$" are defined, which are a function of time, and change in every time instance of the point motion, as shown in Figure 3.6.

In order to derive the normals of the FS-frame: e_x, e_y, e_z, it is important to mention that the velocity of the described point at each time instance is denoted as: $v(t) = \dot{p}(t)$. Also the first and second order derivatives are used for the calculation of the normals and the descriptors: $\dot{v}(t), \ddot{v}(t)$.

The first descriptor that is shown in Figure 3.6: i_1, expresses the velocity of the point when it moves along the path, so along: e_x and therefore its derivation is rather straightforward:

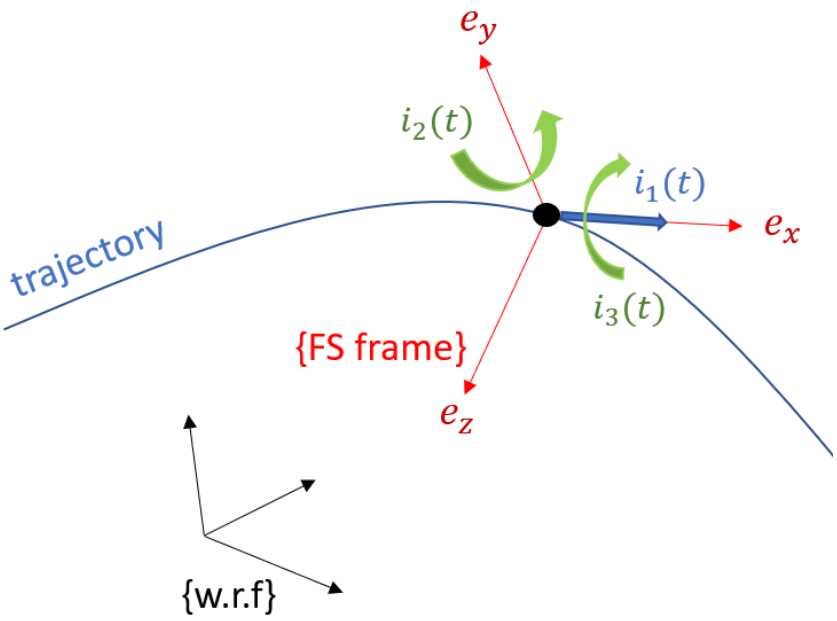


FIGURE 3.6: FS-Descriptors for the translation of a point in space, with respect to world reference frame.

$$e_x = \frac{v(t)}{||v(t)||} \tag{3.15}$$

$$i_1(t) = ||v(t)|| \tag{3.16}$$

The second descriptor: i_2, expresses the rotational velocity of the point around the axis: e_y. The physical meaning of e_y, is the orthogonal to e_x axis, around which the latest rotates. The i_2-descriptor can be seen as the rotational velocity that is required to turn the axis: e_x or in other words the direction of the point's velocity. From another point of view, i_2 could be seen as a way to measure the deviation from the trajectory line a concept similar to "curvature". They are both calculated as:

$$e_y = \frac{v(t) \times \dot{v(t)}}{||v(t) \times \dot{v(t)}||} \tag{3.17}$$

$$i_2(t) = \frac{||v(t) \times \dot{v(t)}||}{||v(t)||^2} \tag{3.18}$$

The last axis of the FS-moving frame is, of course, orthogonal to the axes that were

previously explained and can be calculated as:

$$e_z = e_x \times e_y \tag{3.19}$$

The third descriptor: i_3, as shown in Figure 3.6 is defined on e_x. The reason is that the position of the first axis would be affected by the third descriptor: i_3 and not only the second i_2. In other words, the third descriptor shows how much the point deviates from the plane and it calculated as:

$$i_3(t) = \frac{((v(t) \times \dot{v}(t)) \cdot \ddot{v}(t) \cdot ||v(t)||}{||v(t) \times \dot{v}(t)||^2} \tag{3.20}$$

3.3.3 Frenet-Serret Invariant Representation for Rigid Body Motion

In [17], an extended approach of the pre-referred method is considered, in which an extended form of FS-descriptors, the **extended Frenet-Serret descriptors** are used to describe the motion of a rigid body instead of a point. In a similar manner like the instantaneous screw axis descriptors, the representation took place through six(6) descriptors. Now, the rotational velocity of the rigid body: ω, with respect to the world reference frame (wrf) needs to be taken into account for the formulation of the motion. The overall representation consists of three translational: i_{t1}, i_{t2}, i_{t3}, and three rotational descriptors: i_{r1}, i_{r2}, i_{r3}, denoted with a "t" and an "r", respectivelly as follows:

$$i_{t1} = ||v|| \tag{3.21}$$

$$i_{t2} = \frac{||v \times \dot{v}||}{||v||^2} \tag{3.22}$$

$$i_{t3} = \frac{((v \times \dot{v}) \times (v \times \ddot{v})) \cdot v}{||v \times \dot{v}||^2 \cdot ||v||} \tag{3.23}$$

$$i_{r1} = ||\omega|| \tag{3.24}$$

$$i_{r2} = \frac{||\omega \times \dot{\omega}||}{||\omega||^2} \tag{3.25}$$

$$i_{r3} = \frac{((\omega \times \dot{\omega}) \times (\omega \times \ddot{\omega})) \cdot \omega}{||\omega \times \dot{\omega}||^2 \cdot ||\omega||} \tag{3.26}$$

3.4 Parameterization Approaches in ISA and FS Invariant Descriptors

Until now, the motion representation of a rigid body by the ISA and the FS descriptors was always a function of time. Under different parameterizations [6], they could be transformed into more invariant descriptions of trajectories. The two parameterizations that are taken into account into the current thesis not only for the representation but also for the recognition of such motions in the next chapters, apart from time-based descriptors, are:

- **Geometric Descriptors:** Where the rigid body representation is not dependent on time anymore.
- **Dimensionless Geometric Descriptors:** Where the rigid body representation, except from time, is invariant to amplitude of motion, and its velocity profile.

In the geometric approach, the trajectory, that until now consisted of consecutive time points in space, is parametrized through a scalar number that is dependent on time, the degree of advancement: $\xi(t)$. The rate in which this scalar changes over-time, corresponds to the velocity of a point in the motion trajectory, and is referred to, as rate of advancement: $\dot{\xi}(t)$. With this simplistic, thus, genuine approach, the trajectory can be fully parametrized in every time point as:

$$\dot{\xi}(t) = w_{\xi} \cdot \frac{||\omega(t)||}{\Theta_s} + (1 - w_{\xi}) \cdot \frac{||v(t)||}{L_s} \tag{3.27}$$

The corresponding weights of rate of advancement: w_{ξ}, are scalars with range from "0" to "1", and the parameters: Θ_s, L_s, are again input scalars that are used to scale correctly the translational and the rotational speed.

In order to reparametrize the descriptors, regarding of their type (**FS or ISA**), into their geometric form, the first step is the inversion of the degree of advancement: $t(\xi)$, and then every descriptor seperatelly should be divided with: $\dot{\xi}(t(\xi))$. For example, the fourth Frenet-Serret descriptor: i_{r1}, is reparametrized to $i_{r1}(\xi)$ in (**Equation 3.28**). After the conversion of all six (6) invariants, depending on their type, is finished, invariance is achieved with respect to the time of execution of the motion.

$$i_{r1}(\xi) = \frac{i_{r1}(t)}{\dot{\xi}(t(\xi))} \tag{3.28}$$

In the dimensionless geometric approach, a similar procedure is followed as in (**Equation 3.27**). The concept behind this method, is based on turning the units of the translational: $v(t)$ and rotational speed: $\omega(t)$ to unit-less quantities. In order for that to be achieved, the degree of advancement: $\xi(t)$, is scaled to a unit (1), and

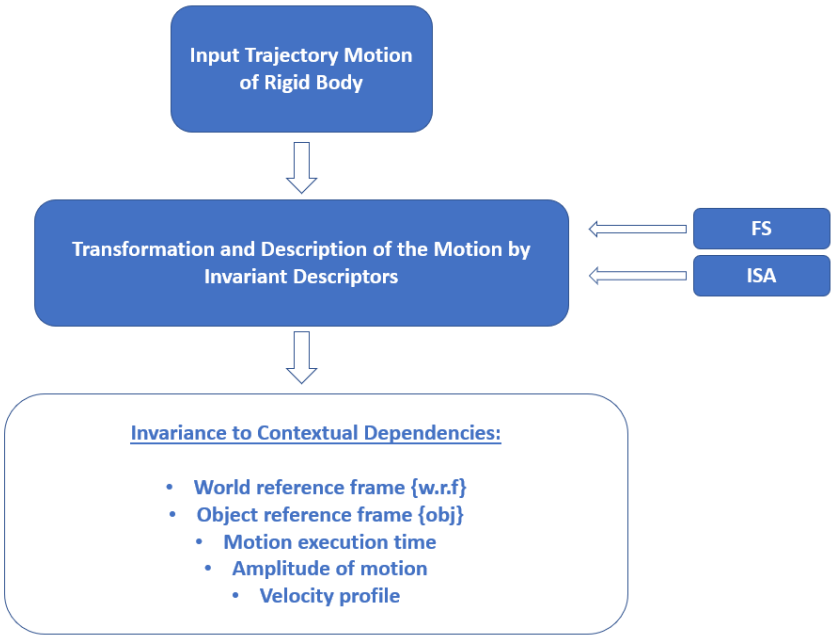


Figure 3.7: Pipeline overview of representing a rigid body motion in a way that is invariant to contextual dependencies.

the dimensionless parameters: $\Theta_{s,dim}$, $L_{s,dim}$, are now defined as the integral of the $\omega(t)$ and $v(t)$, respectively, over the execution time of the motion, from: t_0 to t_f, as shown in (**Equations 3.29, 3.30**). In this way, when divided by the corresponding velocities, the velocities become unit-less. Under this type of parameterization of the descriptors, the representation of a rigid body motion becomes invariant in many ways that are mentioned in the present sub-chapter. In Chapter 5, where the results of the motion recognition approaches will be presented, it will be clear, that the more invariant properties a trajectory representation has, the best classification accuracy it can achieve. In Figure 3.7, a brief overview of the process of making the motion invariant to contextual dependencies will be summarized and illustrated.

$$\Theta_{s,dim} = \int_{t_0}^{t_f} ||\omega(t)||\, dt \tag{3.29}$$

$$L_{s,dim} = \int_{t_0}^{t_f} ||v(t)||\, dt \tag{3.30}$$

Data-Set		
Motion	Execution Style	Camera View-point
Shaker	Gewoon	1
Tafel Afvegen	Groter	2
Tas Scheppen en Uitgieten	Sneller Trager	3
Tas Uitgieten	Trager	-
Tas Wegzetten	-	-
OnderKwart	-	-
ScheppenEten	-	-
Schilderen	-	-
Sinus	-	-
Snijden	-	-

TABLE 3.1: Data-set of motions consisting of the motion classes, the execution styles and the camera view-points.

3.5 Data-Set and Calculation Approaches of ISA and FS Invariant Descriptors

In the present sub-chapter two (2) calculation approaches of the Frenet-Serret and the Instantaneous Screw Axis invariant descriptors will be discussed. First, the data-set that was provided and acquired in [18] will be explained, which includes ten classes, executed in different ways and recorded in various angles. Secondly, the methodology behind calculating the descriptors via analytical formulas will be provided, and then, an optimized approach for the invariants calculation will be provided. In both cases, the strengths and weaknesses will be discussed, and in the next chapters, it will be clear how they affect in an immediate way the recognition of the rigid body motions.

The data-set [18], that was provided, consists of ten (10) different motions executed by a human, like shaking, pouring, putting away a cup, wiping a table and other motions, that are mentioned (in Dutch) in the first column of Table 3.1. Every motion, is executed under four (4) different execution styles which are mentioned in the second column of the same table and consist of the:

- Execution of a motion in a slow manner.
- Execution of a motion with a normal speed.
- Execution of a motion with a fast speed.
- Execution of a motion in a slow then a fast speed, in order for variations to exist, in the velocity profile.

The motion is also recorded from three (3) different view-points of the camera (third column of Table 3.1, in order to check if the view under three different world reference

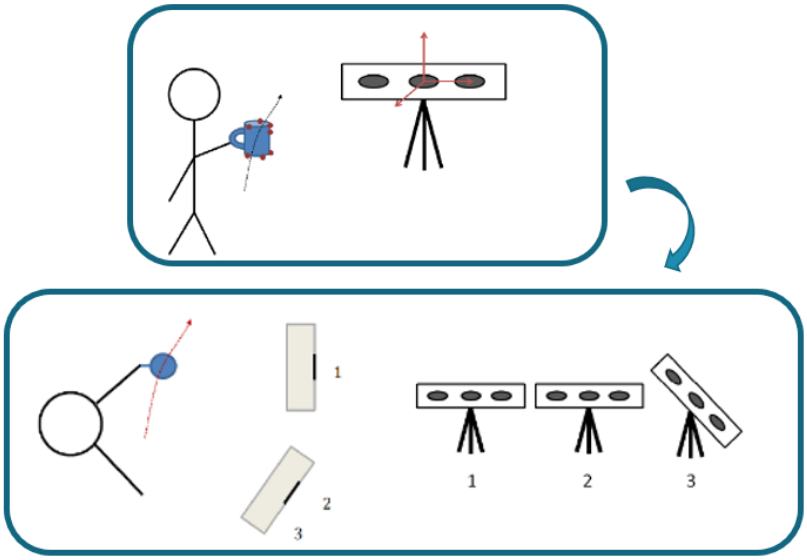


Figure 3.8: Illustration of a human executing the motion, like shaking a cup in various styles, which is recorded from the three camera view-points.

frames, as shown in Figure 3.8, is capable of affecting the results in recognition of the motions, which will take place in the next chapters. Finally, it is important to mention, that the total number of motions in the data-set that was provided is $10 \times 10 \times 12 = 1200$, consisting of 10 motion classes, repeated 10 times each, under $3 \times 4 = 12$ variations (3 in camera view-point and 4 in execution styles).

In order to calculate the invariant descriptors for the two (2) types, **FS** and **ISA**, considered in this thesis, the marker positions on the object that executes the motion were smoothed to reduce the measurement noise, via a Kalman Smoother, and then the smoothed marker positions and their corresponding derivatives were calculated. Next, through (**Eq: 3.9-3.14**) and (**Eq: 3.21-3.26**), the instantaneous screw axis and Frenet-Serret descriptors were calculated, respectively, as shown in (**Figure 3.9**).

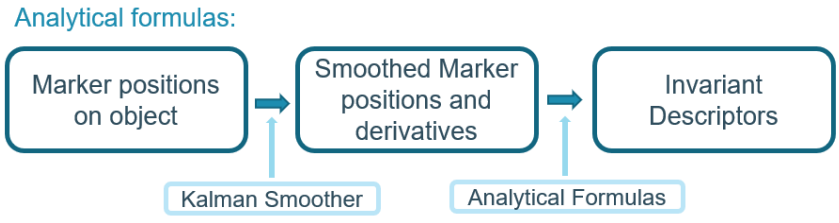


Figure 3.9: Illustration of the procedure of the invariant descriptors calculation via analytical formulas.

To provide a sufficient illustration of how a trajectory, of a rigid body, calculated with analytical formulas is represented by the invariant descriptors, in Figure 3.10, the six time-based ISA-descriptors: $\omega_1, \omega_2, \omega_3, v_1, v_2, v_3$ of the "Shaker" motion, repeated ten (10) times is provided. In the same way, in Figure 3.11, the six time-based FS-descriptors: $i_{r1}, i_{r2}, i_{r3}, i_{t1}, i_{t2}, i_{t3}$, for the "Tas scheppen en uitgieten" motion, are demonstrated in a similar manner, over the 10 descriptor trials.

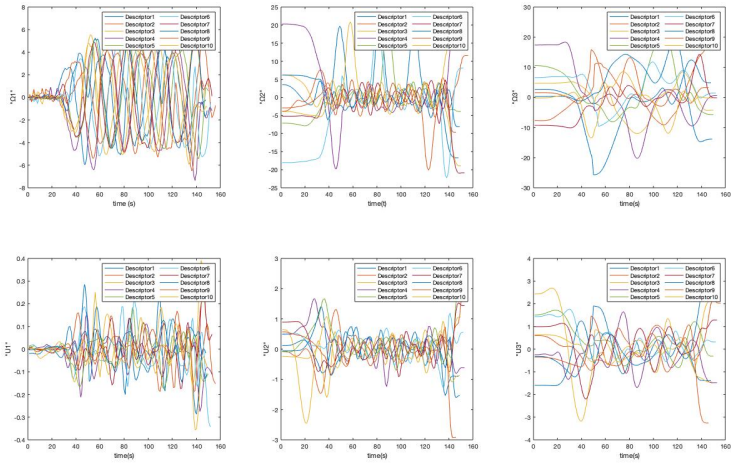


FIGURE 3.10: ISA-time-based-descriptors in the order: $\omega_1, \omega_2, \omega_3, v_1, v_2, v_3$ for "Shaker" motion, for 10 trials each.

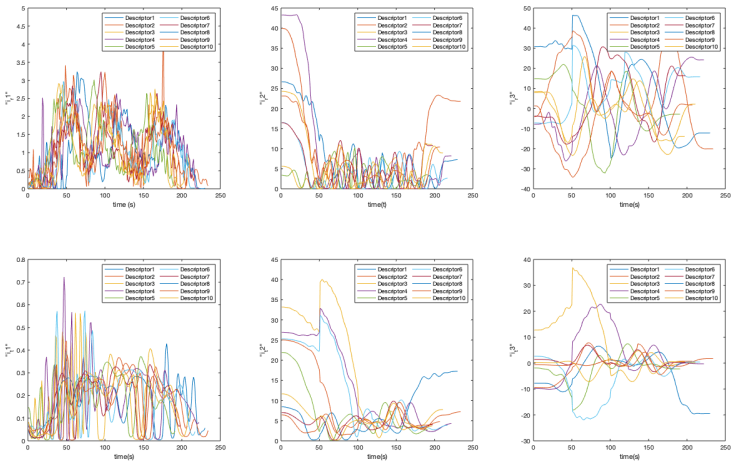


FIGURE 3.11: FS-timebased-descriptors in the order: $i_{r1}, i_{r2}, i_{r3}, i_{t1}, i_{t2}, i_{t3}$ for "Tas scheppen en uitgieten" motion, for 10 trials each.

In appendix A, also the geometric and dimensionless geometric ISA-descriptors of the "Shaker" motion and FS-descriptors of the "Tas scheppen en uitgieten" motion will be presented as well, over ten demonstration trials.

Due to the nature of rigid body motion in space, sometimes the rigid body may be forced to stay still for a moment, or move in a straight line for some seconds. In this case, it is evident that the translational or the rotational speed, or both, will tend to, or become zero, resulting in some undefined descriptors or some unwanted peaks, both for the ISA and the FS descriptor-types. These anomalies in the calculation of the invariant descriptors will be referred to as: "singularities", and is a common phenomenon during the calculation via analytical formulas. Instead, a second calculation approach, considering an optimized way of describing a rigid body trajectory, was introduced by [17], and was also researched by the current thesis work so as it can be applied to all parameterization types of the descriptors (time-based, geometric, dimensionless geometric), tends to eliminate most of the "singularities" in the trajectory. In this approach, the goal is to minimize the difference between the measured motion trajectory and the reconstructed trajectory by the invariant descriptors, with added regularizations presented in [17], as shown in Figure 3.12:

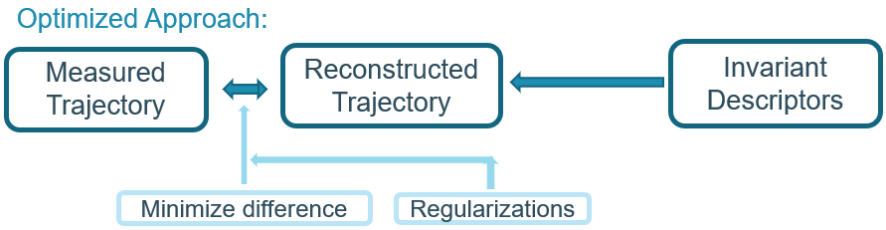


Figure 3.12: Optimized approach for the calculation of the invariant descriptors.

By using this approach, the calculation computational efficiency is drastically lower, than the calculation of the descriptors with analytical formulas (about 1000 times slower for the whole data-set). On the other hand, most of the singularities are eliminated, as shown in Figure 3.13, where for a certain motion, the ISA-descriptors are illustrated, when calculated with analytical formulas and with the optimized approach.

3.6 Conclusion

In this chapter, the concept of invariant descriptors was explained, under different parameterizations. Additionally, an overview as well as an illustration of what the descriptors look like was provided, under different types of calculations. An important aspect of the invariants is that each descriptor, resembles a time-series sequence, which will be the base-idea for the recognition techniques that will be elaborated in the next chapter. Additionally, it is expected, as explained, that the more invariant

a descriptor of a trajectory is, the highest recognition rates it can achieve. Finally, it is also expected that descriptors that are calculated with the optimized approach will be recognized better, than those calculated through analytical formulas, due to the limited "singularities" in the time-series.

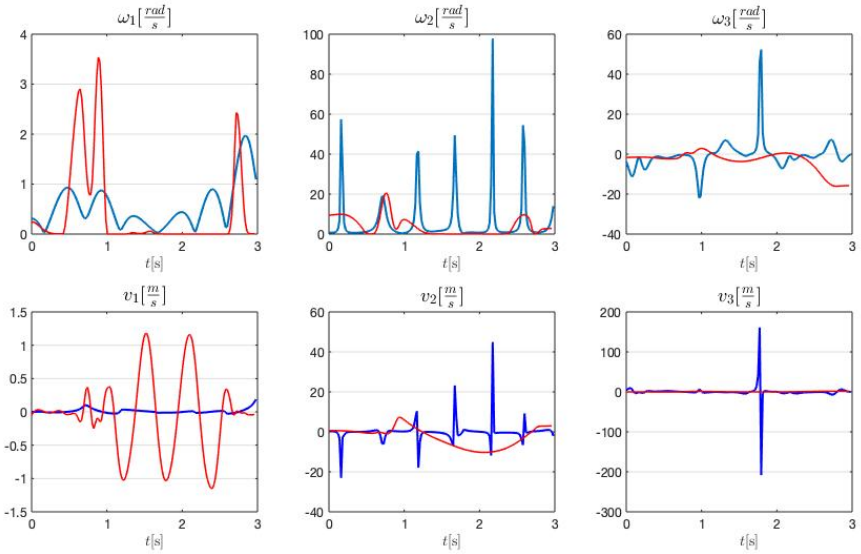


FIGURE 3.13: Comparison of ISA-descriptors calculated with analytical formulas (blue), with the optimized descriptors (red)

Chapter 4

Motion Recognition Approaches

In the present chapter, the approaches that were followed, in the present thesis, in order to achieve robust recognition will be presented. After an overview of the already established related recognition work on the invariant trajectory representation field concerning classification via dynamic time warping (**DTW**), a distance based (**DTW-KNN**) and a deep learning based (**LSTM**) method ,will be elaborated in detail. Both recognition approaches, were executed in **MATLAB**-environment.

As previously mentioned in the second chapter, the reason behind presenting this part of the related work in this chapter, lies in its close correlation to the theoretical framework of the novel approaches that will be explained in the next sub-sections, as well as the better understanding from the reader's point of view.

4.1 Related Work

4.1.1 Dynamic Time Warping

The dynamic time warp distance between two time series is a distance metric for comparing two time series, where parts of the time series can be non-linearly warped for a better matching.

A very important feature of dynamic time warping is that it is possible to compare two time series (or sequences of points) of different length, creating a type of invariance with respect to the execution time of the motion. The later is feasible due to the nature of **DTW-algorithm**, which instead of comparing the raw Euclidean distance between two points of two different time-series (or sequences), it compares a point from the first time-series to the point that is most likely similar to it, from the second time-series.

An example to make the procedure more coherent, is illustrated in Figure 4.1, where one can observe two time series (a blue and a red). In the top figure, the Euclidean distance between every point in the time-series is calculated, providing a low amount of similarity between the points where the distance is calculated. On the

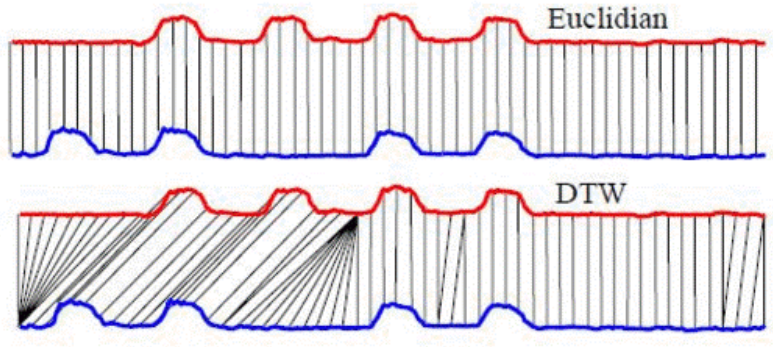


Figure 4.1: Similarity measure of two time-series (red and blue), by calculating the Euclidean (top) and the DTW (bottom) distance of points between them.

other hand, the "warping" in the bottom figure can be observed, resulting in a better similarity measure between the two time-series. To conclude, the resulting distance in the bottom figure, is robust against small variations in the execution speed of both time series compared to e.g. the Euclidean distance.

4.1.2 Dynamic Time Warping Algorithm for Motion Classification (DTW-exponential)

For the DTW-exponential classification, the recognition of a set of **descriptor trials** will take place. Each trial is represented by a structure that consists of six invariant descriptors, depending on the type:

- $i_{t1}, i_{t2}, i_{t3}, i_{r1}, i_{r2}, i_{r3}$: for the Frennet-serret trajectory representation.
- $\omega_1, \omega_2, \omega_3, v_1, v_2, v_3$: for the Instantaneous Screw Axis trajectory representation.

A model trajectory is then build for every one out of the ten motion classes in [18], using the trials recorded from the first camera view-point, with a normal execution speed. The model is calculated by averaging all trials in the training data-set such that the DTW-distance of the models to the trials is minimal. The variance of the model is captured using an exponential likelihood function: $p(DTW_{ij}|c_j)$. When a new trial is compared using DTW distance, between every trial i and the model j, to the different models, the exponential likelihood tells us how likely the distance is given the model. Likelihood and prior are combined in a Bayesian way to get a posterior probability on which the classification decision is made and a class c_j is recognized.

4.2 Dynamic Time Warp (DTW) - based, k-Nearest Neighbors Algorithm (DTW-KNN)

In the present section, a novel distance based approach for the recognition of invariant trajectories will be presented, based on the k-Nearest Neighbor Algorithm. After briefly explaining how the algorithm is generally used for classification in the literature, i will go through my alternative approach for recognition (DTW-KNN), using the DTW-distance as a similarity distance metric.

4.2.1 k-Nearest Neighbors Algorithm

The traditional k-Nearest Neighbor is a simple supervised machine learning algorithm, used for regression and classification applications. A simplified approach is, that given a number of training vectors that consist of features and assigned with labels (classes) to each training vector in a two-dimensional space, the k-NN algorithm [19] attaches a new label to a new input testing feature vector that enters the two-dimensional space. The classification of the new test-vector is dependent on k, which is a chosen scalar and describes the number of input train-vectors that are closer to the new test-vector. The term closer, lies in the minimum values of a chosen distance metric between the features of the new test-vectors over all the train-vectors.

The most usual distance metrics that are applied between the features of the vectors in order for the algorithm to recognize a new vector and label it are the **Euclidean** and the **Manhattan** distance metrics and are presented in **Equation 4.1** and **Equation 4.2**, respectively [15]:

$$\sqrt{\sum_{n=1}^{N}(X_i^{test} - X_i^{train})^2} \tag{4.1}$$

$$\sum_{n=1}^{N}|X_i^{test} - X_i^{train}| \tag{4.2}$$

where N is the number of features the vectors contain and X_i^{train}, X_i^{test}, correspond the to the i^{th} train and test-vector respectively.

In a simple illustration of how the k-NN algorithm works in a two-dimensional space, in Figure 4.2. A new test-vector (green) will be classified into two possible classes (red and blue), based on $k = 3$ train-vectors, the features of which are closer to the input test-vector. It is obvious that the blue neighbors appear the most around the green vector, while there is only one red neighbor. In other words, the green test vector most likely includes blue-class features and therefore is classified as blue.

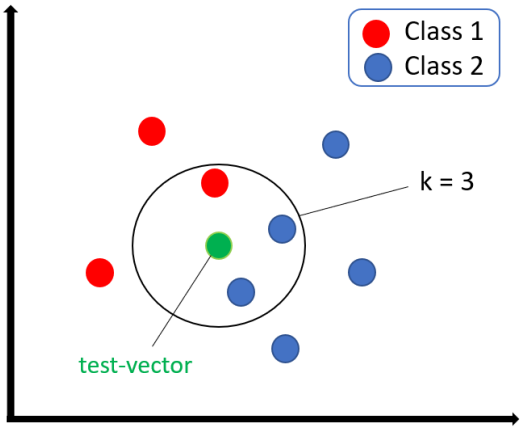


Figure 4.2: Illustration of KNN classification for a test-vector within k-range of labeled train-vectors.

4.2.2 DTW-KNN for Invariant Trajectory Representations

In this sub-chapter, the recognition pipeline will be elaborated, based on **time-based instantaneous screw axis descriptors**: $\omega_1(t), \omega_2(t), \omega_3(t), v_1(t), v_2(t), v_3(t)$, so as the procedure is clear and understandable for the readers of this thesis work. The procedure followed for the rest of the representations (ISA, FS), under various parameterizations (time-based, geometric, dimensionless geometric) is very similar and their recognition results will be explained and compared in the next chapter.

In order to create a classification pipeline for the invariant trajectory representations that were explained in detail in the third chapter, the first step is the reformulation of the data-set so as KNN algorithm can be implemented.

In this case, the evolution of the six descriptors over the motion execution time, completely describe the trajectory of a rigid body. Each set of six descriptors that corresponds to a specific motion will be referred to as **trial**. The execution time of each trial is divided into a number of **samples**, which depends on the initial recording of each trial. As a result, the X_i^{train} and X_i^{test}, are three dimensional matrices, as illustrated in Figure 4.3 that consist of three directions:

1. Trials Direction
2. Invariants Direction
3. Samples Direction

The amount of trial motions that are used for training the data is chosen as one over twelve (1/12) of the whole data-set and includes all the trials executed in a normal style from the first camera viewpoint (Gewoon1). As a consequence, the test-trials cover the rest of the data-set (11/12). In order for every trial to have the

same samples length, each trial is interpolated to a 200-samples length. This act, provides a type of invariance itself, as far as the execution time of the motion is considered. The dimensions of the test and train three dimensional matrices, as a result, are shown below, where $i = 1, .., 6$ and corresponds to the invariant in the order presented in Figure 4.3:

- X_i^{train} = 100 x 6 x 200
- X_i^{test} = 983 x 6 x 200

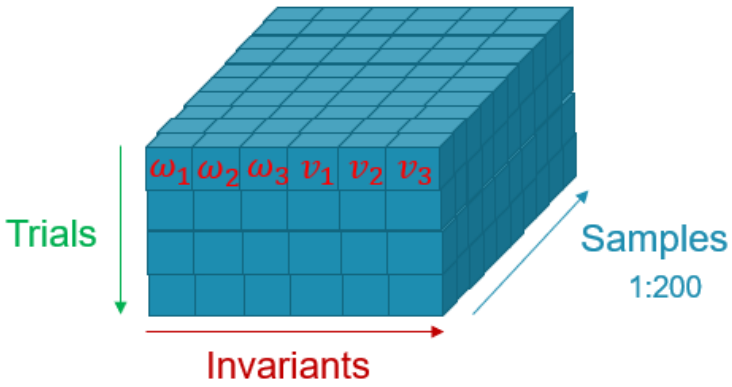


FIGURE 4.3: Representation of a 3D data matrix, with the corresponding directions: Trials (green), Invariants (red), Samples (blue).

Once the X_i^{train} and X_i^{test}, are formulated, the corresponding labels of the training trials are also provided and stored in a 100×1 -matrix, named Y_{train}.

As explained, every trial consists of six different time-series of 200-length each. A crucial point in the classification pipeline is the normalization of the six time-series since half of them that represent the: $\omega(t)$ descriptors are expressed in rad/s and the other half that represent the: $v(t)$ descriptors are expressed in m/s. In order to transform the descriptors to unit-less and comparable, the highest scalar value of each descriptor i was chosen over the samples direction. In the next step, every i^{th} descriptor is multiplied by a weight: w_i which corresponds to one over the maximum absolute value of the i^{th} descriptor. The normalization of weights, is also presented in a pseudo-code in Figure 4.4, which represents the weights normalization for the X_i^{test} matrix. The same procedure is also followed for X_i^{train}.

For the next step, the DTW-distances of i-descriptors of every test-sample over all the train-samples are calculated, via the DTW-algorithm as explained in [13] and then are summed over the samples-direction and saved in a new matrix structure, called **distance matrix** and denoted as D. The rows of D, represent the number of test-trials and the columns represent the number of train-trials, so the dimensions of the D-matrix are: $983x100$, and the value of a random nxm-cell represent the

```
$Calculate normalization weights for X_test
for i=1:6
  wi=1/max(max(abs(X_test(:,i,:))));
end

$Place weights in a structure w
w=[w1;w2;w3;w4;w5;w6]

$Normalize the test data
for weight_index=1:size(w,1)
X_test(:,weight_index,:)=w(weight_index)*X_test(:,weight_index,:);
end
```

Figure 4.4: Pseudo-code of normalization of weighs on the test-set: X_i^{test}.

DTW-distance between the n^{th} test-sample and the m^{th} train-sample.

The final step in the recognition pipeline, begins with re-arranging the cells in the D-matrix, so as the DTW-distance between every test and train-trial, is stored in increasing order.

Then a scalar parameter k is chosen. This parameter represents the number of neighbor train-trials that have the smallest DTW-distance the new test-trial that is going to get labeled. Since, in the previous step, the distance were sorted in increasing order, when choosing k=3, for example, the three closest train trials (neighbors) to the new test-trial will be the most attractive candidates for the test-trial's classification. It is important to remember that each training trial is labeled a priori and all the train trials' classes are stored inside: Y_{train}, as mentioned previously. An example of this case is illustrated in Figure 4.5, for purposes of clarity of the final classification step.

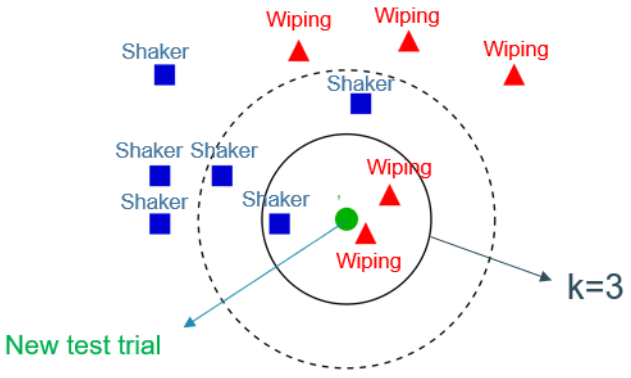


Figure 4.5: New test-trial labeled as "Wiping" motion, through the DTW-KNN algorithm.

4.3 Long short-term memory network (LSTM) for Invariant Trajectory Representations

As an alternative approach for recognition of motions described in an invariant way, a deep learning approach was chosen. Long short-term memory networks (LSTM's), [14], are a specific architecture of recurrent neural networks (RNN's) with feedback connections.

The reason behind the choice of constructing and using such a network, instead of using a feed-forward artificial network or a convolutional neural network, as mentioned in the literature review chapter, is because it can handle:

- **Sequential Data**: With the right reformulation of the data-set, a rigid body trajectory can be divided into a number of sequence structures so as they can be used as input to train the LSTM network.
- **Variable length of motion trials**: In the DTW-KNN approach, all trials were interpolated to a common 200-length size. By doing that although an invariance, with respect to the execution time of the motion, was achieved, some information of the trajectory was lost during the interpolation, a problem, which is eliminated with the current approach.

In the remaining content of this chapter, the procedure behind the construction of an LSTM network for the recognition of motions described in an invariant way through **FS** and **ISA** descriptors will be elaborated in steps.

The **first step** consists of the reformulation of the data-set. It is important to mention, that also in this approach, the splitting between the train and the testing data took place under the same ratio (1/12 of data-set and 11/12 of data-set, respectively) as in DTW-KNN, such as a comparative basis can be established between the different approaches used in this thesis.

Instead of using a three dimensional matrix that consists of the number of trials, the number of invariant descriptors and the sample-length for each trial, in this case the X^{train} and X^{test}, are reformulated into a sequence-type structure, where the sample-length for each trial is not normalized and kept as it is, without losing information. It is clear that now that every trial in the data-set, consists of a cell in which information is stored about the six descriptors that describe the motion and the sample-length of the trial, as illustrated in Figure 4.6. After all trials of motion are reformulated this way, the splitting of the test and train trials takes place, under the pre-determined ratio.

The **second step**, consists of the construction of the LSTM network, using the training sequences as inputs to the network, so as it can be trained efficiently in the third step.

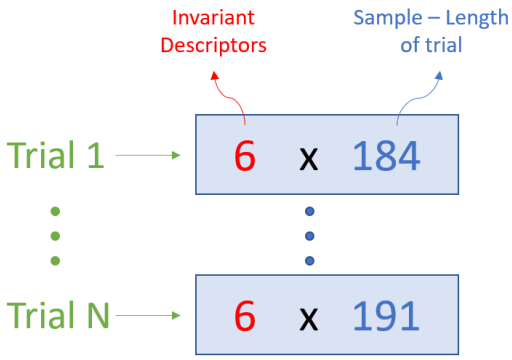


Figure 4.6: Illustration a the motion trial (green) sequence, containing information about the descriptors (red) and the sample-length (blue) of this trial.

There are a hundred (100) input training sequences (1/12 of data-set), and every sequence from now on, will be referred to as X_i, where X denotes the input sequence and $i = 1, .., T$, the sample length of this motion trial from the time moment $t = 1$ to $t = T$, where T is the total time-length of the motion. The dimension of each input X_i, as a consequence, is: 6×1, where six (6) corresponds to the number of descriptors.

Each motion trial sequence X_i, is a unique input to a hidden layer H_i and gives an output O_i. In this case each output O_i is a 10×1-vector, due to the ten (10) motion classes that are available in the data-set, and the hidden layers H_i are interconnected. Before explaining how the network operates, it is might be useful to present the inputs that go through the interconnected sequence of hidden layers, in order for each input: X_i to provide a O_i output (Figure 4.7).

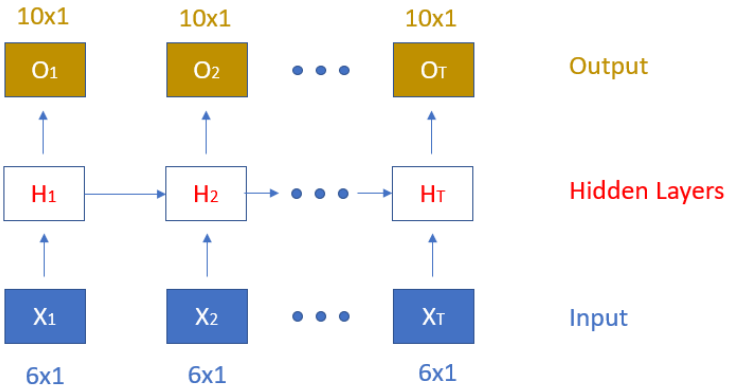


Figure 4.7: Sequences of motion trial inputs: X_i that go through interconnected hidden layers: H_i and give output: O_i.

As mentioned, in this case, a signal consists of six (6) features for every time point

and ten (10) classes in total. Inside every interconnected hidden layer, there is a number of neurons (hidden units), a parameter which is chosen by the user Figure 4.8. The choice of the number of hidden units is elaborated in the next chapter, where the results will me commented upon, including the possible limitations that were encountered.

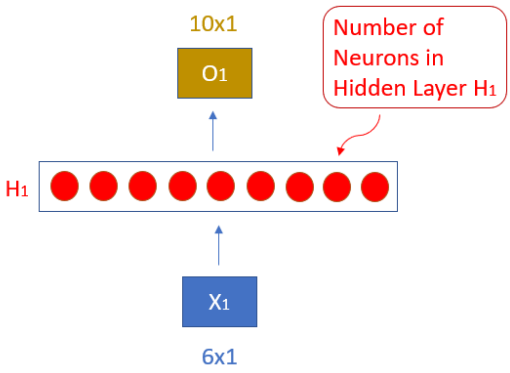


FIGURE 4.8: Illustration of the neurons (hidden units) inside a hidden layer: H_i.

The importance and the reason of choice of constructing an LSTM-network for motion recognition, lies on the **interconnection** of the hidden layers. The information stored in the neurons of the first hidden layer: H_1, is not wasted, but instead is re-used, due to the interconnections between them. In this way, the information of the features (invariant descriptors) at a random k-point of the trajectory is also re-used in the next time step (next point): $k+1$, so as in simple words, as the trajectory evolves, it is possible to know what happened through the whole motion trajectory and not just in discrete points of it.

An important question that might emerge, could be about how the interconnections between the hidden layers work and how are they chosen in the current thesis work. Every hidden layer: H_t is connected, and can be calculated from the previous hidden layer: H_{t-1}, with an operation that includes two (2) constant weight matrices: W_{xh}, W_{hh}. These matrices, remain the same during the whole length of the network and the procedure of calculating H_t is shown by:

$$H_t = N(W_{xh} \times X_t + W_{hh} \times H_{t-1} + b) \tag{4.3}$$

where:

- $N()$: is a non-linear function, which in this thesis work is chosen as $tan(H_t)$.
- X_t: is the motion trial input X_i at time t, with dimensions: (6×1).
- H_t, H_{t-1}: are the hidden layer matrices at time t and $t-1$ respectively, with dimensions: $(Number of Neurons \times 1)$.

- W_{xh}: is the input x to layer h weight matrix, with dimensions: $(Number of Neurons \times 6)$.
- W_{hh}: is the layer h to layer h weight matrix, with dimensions: $(Number of Neurons \times Number of Neurons)$.
- b: is a bias matrix, with dimensions: $Number of Neurons \times 1$

The initial calculation of: W_{xh} and W_{hh}, is done through (Equation 4.3) at $t = 0$, and they remain constant through the LSTM-network.

After the hidden layer sequence, comes a fully connected layer consisting of ten (10) neurons, equal to the number of motion classes available to be recognized. All the hidden units of each hidden layer are connected to all ten (10) neurons of the fully connected layer. Then due to the multi-class nature of this recognition approach, a SOFTMAX prediction formula was chosen, as suggested by [8], in order to finally determine the corresponding class of the input sequence trial that is fed into the network. The final structure of the LSTM-network that was described is illustrated in Figure 4.9.

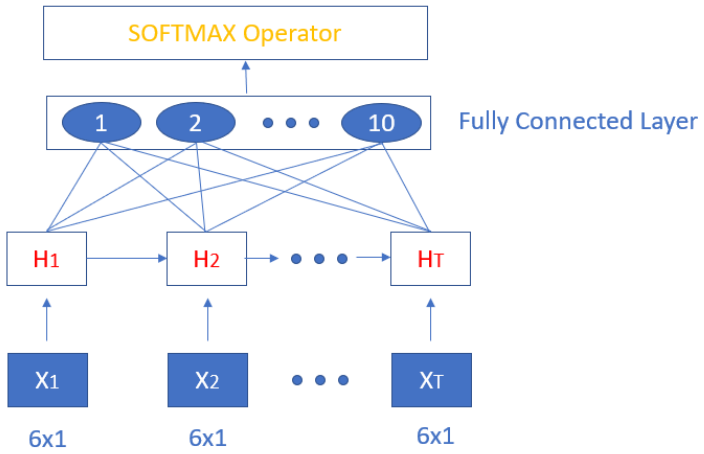


Figure 4.9: Final (simplified) structure of the LSTM-network, with emphasis on the fully connected layer, followed by the SOFTMAX operator.

The **third step** lies in the training of the constructed LSTM network, with the training data as input, so as when one feeds the test-data inside the network, the recognition can take place. Normally, these network architectures require about 75% of the whole data-set to be used for training purposes and the rest of 25% for testing, while in this thesis, only the 8.33% (1/12) of the data-set will be used for the network-training and the rest (11/12) will be used for testing the classification accuracy of the trained network.

Before training the network, each train-trial is split and sorted in increasing order of sample-length resulting in a sorted data format. This procedure is very simple and is illustrated in Figure 4.10

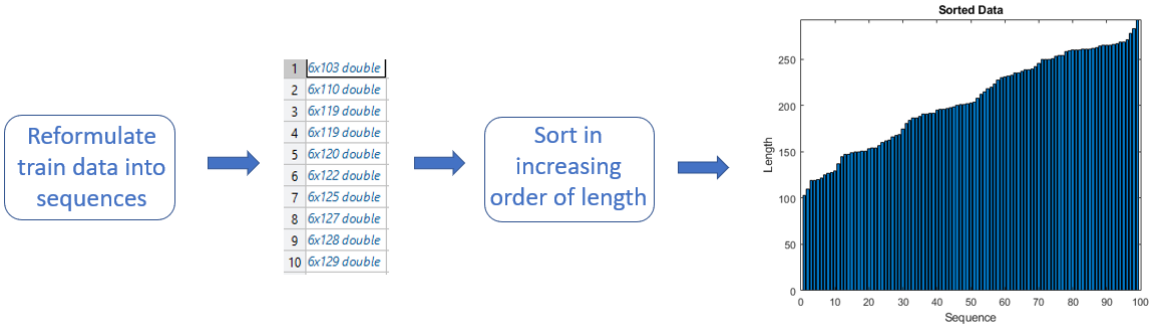


FIGURE 4.10: Illustration of the pipeline from the reformulation of the training data-set to the sorted data format.

In the figure above, one can observe the (100) train-trials, sorted in increasing order. In addition, the sorted data are divided in trial-batches of equal number. This number is called the mini-batch size and is a user-based parameter. Mini-batch size is a scalar number that can equally divide the input trials into a number of batches. Due to the number of the train trials (100) in this case, the mini-batch size is chosen to be: ten (10), so as the data can be divided into $100/10 = 10$ equal batches. Finally, in each mini-batch, all trial samples are chosen to become normalized so as all become equal to the smallest sample-length in each mini-batch, as presented in Figure 4.11. Dividing the sorted data into a number of mini-batches is very useful for the computational efficiency of the network training, since at each training iteration of every epoch (total training cycle over the data-set), the number of observations that are included is equal to the mini-batch, and thus, less memory is required for the network to train.

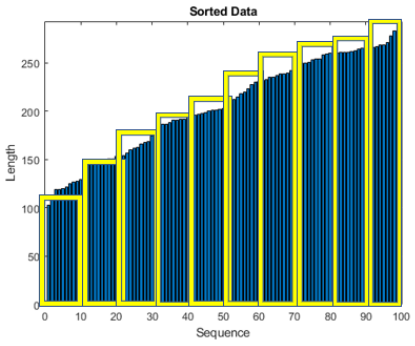


FIGURE 4.11: Division of the sorted-data into ten (10) mini-batches, each of which includes ten (10) motion trials.

In addition, the Adam optimization algorithm, was chosen for the training of the network, so as the error between the predicted and the real motions is minimized, since it can handle gradients on noisy problems, due to the nature of the invariant descriptors' data, as presented in [12], although other optimization algorithms were taken into account initially but were outperformed by Adam. During the training of the LSTM network with the input training-data, it is important to follow its accuracy in order to have a rough estimate of how well the network is being trained. This was done by plotting the **training accuracy** of the network as well as its **loss**. The training accuracy is how accurately the network classifies the training data to the correct class that they belong to, without inputting the test-data at all. In contrast, the loss is, in simple words, how far the predicted class is from the real motion class. In other words, as the network trains, the accuracy should increase to almost 100% and the loss should decrease to almost 0%. The accuracy and the loss are shown in a combined way, so as they are comparable, in Figure 4.12. The present figure represents the training process of a network that corresponds to the dimensionless geometric Frenet-Serret invariant descriptors.

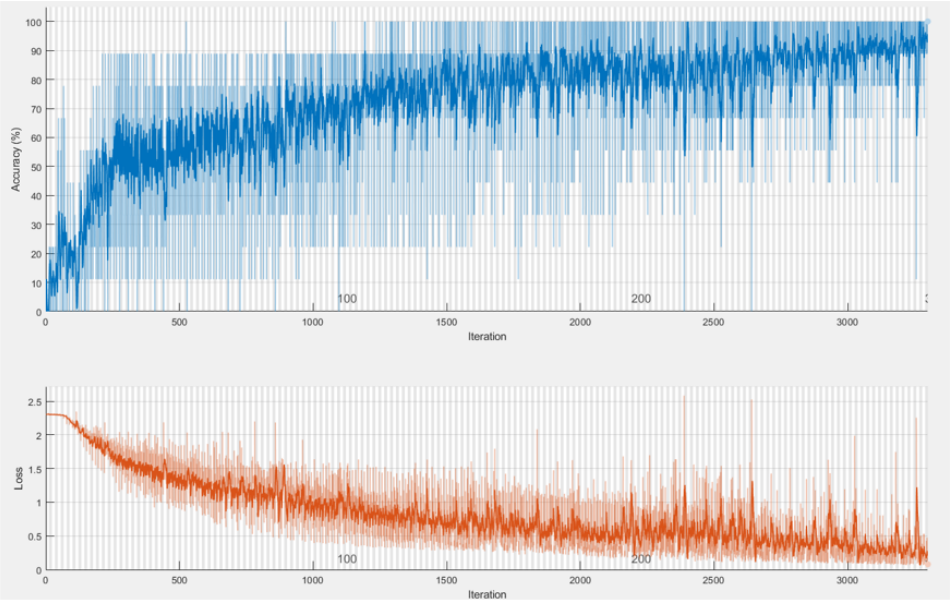


Figure 4.12: Training accuracy (blue) and loss (orange), during the training iterations of the constructed LSTM-network for dimensionless geometric FS invariants.

After the training of the constructed LSTM-network, the **final step** is to check its testing accuracy. By feeding X^{test} into the trained-LSTM, after dividing it into mini-batches in the same way as it happened for X^{train}, the recognition part takes place, and the motion trials in X^{test} are labelled accordingly.

4.4 Conclusion

In this chapter, two different recognition approaches (DTW-KNN, LSTM-network) were proposed for classifying rigid body motions, described in an invariant way, while being inspired from the state of the art recognition approaches (Interval feature classification of time series, Spatial trajectory recognition via representation learning) that were explained in the first chapter. The first approach is through a distance-based machine learning algorithm and the second, through a deep-network architecture. In addition, due to the similar recognition context of the state of the art with the current methods, the close coalition and the inspiration of the approaches' methodologies, should be evident.

The classification accuracy of the test data for the different types of invariant descriptors (FS, ISA), under different parameterizations (time-based, geometric, dimensionless geometric) and calculation approaches (analytical formulas, optimized approach), for the two novel methods for trajectory classification that were introduced in this chapter, will be presented and analyzed in the next chapter, while also comparing them with the DTW-exponential approach, that was mentioned in the start of the current chapter.

Chapter 5

Comparison between Motion Recognition Approaches and Discussion of Results

After the elaboration of the recognition approaches that were followed in the previous chapter, first the recognition results of the DTW-KNN approach will be presented and compared with the DTW-exponential approach. Then, in the second sub-chapter the classification results of using the LSTM-network will be illustrated, followed by annotating the strengths and the weaknesses of this method, while also comparing them to the previous approaches. Finally, in the third sub-chapter, the discussion of the results will take place, while focusing on how the invariance of the descriptors that describe a motion trajectory affects the classification accuracy of the motion.

5.1 DTW-KNN Classification Accuracy and Comparison with DTW-exponential

Before the demonstration of the recognition results and the comparison of approaches, it is important to mention that the classification accuracy for each invariant motion type, parameterization and calculation approach will be presented through a **confusion matrix**, as shown in [16]. The confusion matrix is a structure that represents the recognition accuracy of the motions. The rows represent the actual motion classes M_j and the columns represent the predicted motion classes PM_j, where: j is the number of class labels, that are presented also in Table 3.1 and $j = 1, 2, ..., 10$. As a result, the cells in the **diagonal of the confusion matrix**: c_{jj}, represent how well or poorly a motion is classified.

All cells c of the confusion matrix, are expressed in a probability range from 0 to 1. In the worse situation where: $c_{jj} = 0$ for example, the j motion has totally failed to get recognized and was probably confused with an other motion class.

As explained, the **dimensionless geometric invariant descriptors** have the highest invariance with respect to contextual dependencies, and thus they are expected to perform better, under the different recognition approaches, compared to the **geometric**, followed by the **time-based** invariant descriptors. The classification accuracy results for the DTW-KNN approach are presented via the confusion matrix representation, in an order from highest to lowest invariant description of the motion. Also it is important to mention that the **overall classification accuracy** for every invariant description type, parameterization and calculation approach is also presented, so as an overall view of the results is provided.

For efficiency purposes, only the Frenet-Serret invariants' confusion matrices will be presented in this chapter, under the dimensionless geometric, geometric and time-based parameterizations, calculated with analytical formulas. The rest of the recognition results can be found on the appendix chapter.

It was mentioned in the previous chapter the the number of neighbors: k is a user-based scalar parameter. Through testing, the best values of k, depending on the type of the invariant descriptors are determined as:

- k=4 : for FS-invariants
- k=3 : for ISA-invariants

For **dimensionless geometric FS-invariants**, the overall classification accuracy of ten motion classes is: **81.18%** and the confusion matrix for the recognition of the ten classes is shown in Figure: 5.1.

	PM1	PM2	PM3	PM4	PM5	PM6	PM7	PM8	PM9	PM10
M1	**0,5872**	0	0	0	0	0,0642	0	0,3486	0	0
M2	0	**0,9694**	0,0204	0	0	0	0	0	0,0102	0
M3	0	0	**0,7079**	0	0	0	0,2022	0,0787	0,0112	0
M4	0	0	0	**0,9903**	0	0	0	0,0097	0	0
M5	0	0	0	0,0575	**0,8046**	0,0805	0	0	0,0575	0
M6	0	0	0	0,0273	0,0909	**0,8364**	0,0091	0	0,0364	0
M7	0,0106	0	0,1809	0	0	0	**0,7553**	0,0213	0,0319	0
M8	0,0108	0	0,0215	0,0430	0	0,0430	0,2258	**0,6344**	0,0215	0
M9	0	0	0,0101	0,0800	0	0,0200	0	0,0300	**0,86**	0
M10	0	0	0,0200	0	0	0	0	0,0200	0	**0,9600**

Figure 5.1: Confusion Matrix for ten (10) motion classes with dimensionless geometric FS invariant descriptors.

For **geometric FS-invariants**, the overall classification accuracy of ten motion classes is: **57.7%** and the confusion matrix for the recognition of the ten classes is shown in Figure: 5.2.

	PM1	PM2	PM3	PM4	PM5	PM6	PM7	PM8	PM9	PM10
M1	**0,1193**	0,0275	0,0092	0	0	0	0	0,3761	0	0,4679
M2	0	**1**	0	0	0	0	0	0	0	0
M3	0	0,0225	**0,4831**	0	0	0	0,4157	0,0674	0	0,0112
M4	0	0	0,0097	**0,8932**	**0**	0,0097	0,0097	0,0291	0,0485	0
M5	0	0,0230	0,0230	0,0805	**0,6322**	0,0345	0	0	0,2069	0
M6	0	0,0273	0,1279	0,0727	0,3636	**0,2455**	0,0727	0,0091	0,0818	0
M7	0	0,0638	0,2872	0,0106	0	0,0106	**0,4681**	0,1170	0,0426	0
M8	0	0	0,0538	0,0215	0	0	0,1183	**0,7634**	0,0323	0,0108
M9	0	0	0,01	0,2	0	0,0100	0,0100	0	**0,77**	0
M10	0	0,33	0,01	0	0	0	0,0300	0,22	0	**0,41**

FIGURE 5.2: Confusion Matrix for ten (10) motion classes with geometric FS invariant descriptors.

For **time-based FS-invariants**, the overall classification accuracy of ten motion classes is: **53.1%** and the confusion matrix for the recognition of the ten classes is shown in Figure: 5.3.

	PM1	PM2	PM3	PM4	PM5	PM6	PM7	PM8	PM9	PM10
M1	**0,5119**	0,0183	0,018	0,0917	0	0,0367	0,0183	0,156	0	0,1486
M2	0	**0,5204**	0,1327	0,0408	0,0204	0	0,1224	0,0816	0,0408	0,0408
M3	0	0	**0,3034**	0,1798	0	0,0449	0,3820	0,0787	0,0112	0
M4	0	0	0	**0,8447**	**0**	0,0971	0,0388	0,0097	0,0097	0
M5	0	0	0	0,0230	**0,6437**	0,2759	0,0115	0	0,0460	0
M6	0	0	0	0,0091	0	**0,9909**	0	0	0	0
M7	0	0,0106	0,1915	0,2660	0,0745	0,0532	**0,3511**	0,0426	0,0106	0
M8	0	0	0	0,2796	0	0,1720	0,0215	**0,4839**	0,043	0
M9	0	0	0,01	0,18	0,16	0,09	0,12	0	**0,440**	0
M10	0	0,17	0,03	0,19	0	0,08	0,02	0,22	0,05	**0,24**

FIGURE 5.3: Confusion Matrix for ten (10) motion classes with time-based FS invariant descriptors.

The three remaining confusion matrices for the instantaneous screw axis descriptors under the different parameterizations will be included in Appendix B.

The comparison of the recognition results, for the invariants that were calculated through **analytical formulas**, between the two approaches (DTW-KNN and DTW-exponential) are shown in 5.4. It is clear, that DTW-KNN managed to outperform the pre-existing DTW-exponential approach in three (3) out of (6) cases.

Overall Classification Accuracy Comparison – Analytical Formulas			
Type of Descriptor	DTW-based KNN	DTW-exponential	Highest!
Dimensionless FS	81.18%	82.1%	DTW-exponential
Dimensionless ISA	74.67%	72.12%	DTW-based KNN
Geometric FS	57.7%	50.14%	DTW-based KNN
Geometric ISA	64.6%	70.1%	DTW-exponential
Time-based FS	53.1%	47.71%	DTW-based KNN
Time-based ISA	50.15%	56.31%	DTW-exponential

Figure 5.4: Comparison of overall results between DTW-KNN and DTW-exponential approaches with analytical formulas.

In addition, with the **optimized approach** of calculating the invariant descriptors, results are expected to be higher, due to the elimination of the singularities and as a consequence, a better recognition for the invariant descriptor motion signals, but in the cost of high computational efficiency. The later, can be observed in: 5.5, where the two recognition approaches are, once more, compared. Also, in the case of optimized approach, although the results where higher than with the analytical formulas - approach, the classification accuracy increased by about 3-6% in each type. As it is expected, the DTW-KNN outperformed the DTW-exponential in three (3) out of (6) cases again.

Overall Classification Accuracy Comparison – Optimized Approach			
Type of Descriptor	DTW-based KNN	DTW-exponential	Highest!
Dimensionless FS	83.13%	85.55%	DTW-exponential
Dimensionless ISA	80.02%	77.10%	DTW-based KNN
Geometric FS	59.09%	53.00%	DTW-based KNN
Geometric ISA	66.98%	74.17%	DTW-exponential
Time-based FS	58.88%	50.05%	DTW-based KNN
Time-based ISA	53.33%	59.31%	DTW-exponential

Figure 5.5: Comparison of overall results between DTW-KNN and DTW-exponential approaches with optimized approach.

5.2 LSTM-network Classification Accuracy and Comparison with DTW-exponential, DTW-KNN

After constructing the LSTM-network as explained in the previous chapter, the training data-set and the corresponding labels were inserted as input to the network. During the first try, the network failed to train, due to the absence of normalization in the six (6) invariant descriptor signals (**features**). The normalization of the input feature-signals is a crucial step, since the magnitude of the values between signals can be very high. The normalization takes place through centering the scalar values of each feature-signal so as they have zero (0) mean value.

To summarize, the hyperparameters that were used for the training of the network are summarized as:

- **Sequence Input**: Sequence input with 6 dimensions.
- **Hidden Layer**: LSTM-hidden layer with 16 hidden units.
- **Fully Connected Layer**: Includes 10 hidden units.
- **Classification Output**: Single class output.

For comparison purposes, also in the present sub-chapter, the confusion matrices of the ten (10) motions will be presented for **dimensionless geometric FS**, which along with the highest invariance, it achieved the highest recognition rate between all the parameterizations considered.

For **dimensionless geometric FS-invariants**, the overall classification accuracy is: **50.71%** and the corresponding confusion matrix is presented in Figure 5.6.

	PM1	PM2	PM3	PM4	PM5	PM6	PM7	PM8	PM9	PM10
M1	**0,3727**	0	0,0455	0,0182	0,0727	0,0091	0,1182	0,3545	0,0091	0
M2	0	**0,9691**	0	0	0	0	0	0	0	0,0309
M3	0	0	**0,6180**	0,0112	0,0112	0	0,0449	0,0337	0,1011	0,1798
M4	0	0	0,068	**0,5728**	**0,0097**	0	0,0291	0,0874	0,2136	0,0194
M5	0	0	0,0115	0,2184	**0,4943**	0,0345	0,0115	0,0690	0,1609	0
M6	0,1273	0	0,0364	0,0273	0,0273	**0,6636**	0,0182	0,0727	0,0273	0
M7	0	0,2283	0,9804	0,0543	0	0	**0,1739**	0,1304	0,0109	0,0217
M8	0,0220	0	0,1099	0,4286	0,0220	0,1868	0,022	**0,0989**	0,1099	0
M9	0	0	0,0606	0,2828	0,0101	0	0	0,0404	**0,6061**	0
M10	0	0,0306	0,2041	0,1327	0,0102	0	0,0204	0,0204	0,1020	**0,4796**

FIGURE 5.6: LSTM-network, confusion matrix with dimensionless geometric FS invariant descriptors.

The rest of the recognition results are presented in: 5.7, where one can observe the classification accuracy between all approaches and also that the more invariant a descriptor is, the best classification accuracy it achieves, although the recognition results with this method are **lower compared to the other approaches that were followed but still not negligible**.

The reason is that in order to train an LSTM-network, a rule of thumb is that the training data-set should be at least three times bigger than the testing-set, where a set of motions will be recognized. The pre-mentioned ratio lies in about:

- Training data-set (X_{train}) = 75% x Data-Set (X)
- Testing data-set (X_{test}) = 25% x Data-set (X)

In contrast, in the current thesis, as mentioned before, only 8.33% of X was used as: X_{train} and the rest 91.667% of X was used as: X_{test}, in all approaches, including the LSTM-approach. This alone, justifies the recognition results and the contributions of this thesis work that will be also elaborated in the concluding chapter.

Overall Classification Accuracy Comparison – Analytical Formulas			
Type of Descriptor	DTW-based KNN	DTW-exponential	LSTM-Network
Dimensionless FS	81.18	82.1	50.71
Dimensionless ISA	74.67	72.12	44.31
Geometric FS	57.7	50.14	30.01
Geometric ISA	64.6	70.1	24.44
Time-based FS	53.1	47.71	18.12
Time-based ISA	50.15	56.31	16.97

Figure 5.7: Comparison of overall results between DTW-KNN, DTW-exponential and LSTM approaches.

To conclude, a very interesting result will be presented and took place when trying to increase the training data-set. If the ten (10) classes to be recognized are split in two parts, and then the recognition takes place for the first (5) and then for the remaining five (5) motions, the overall classification accuracy increases significantly. The later, is equivalent with doubling the training data-set. As an example, for the **geometric instantaneous screw axis invariants**, when following the previous procedure, the corresponding confusion matrices are presented in Figures: 5.8, 5.9, and the overall classification accuracy for the split motion classes is:

- **Classes 1 to 5**: Overall recognition accuracy = 63.78%
- **Classes 6 to 10**: Overall recognition accuracy = 75.31%

	PM1	PM2	PM3	PM4	PM5
M1	**0,6889**	0,0111	0,0778	0,2	0,0222
M2	0	**0,8701**	0,0260	0,0779	0,0260
M3	0,1	0,2	**0,4857**	0,1714	0,0429
M4	0,0723	0,2530	0	**0,5783**	0,0964
M5	0,1029	0,1912	0	0,1324	**0,5735**

FIGURE 5.8: Confusion Matrix, LSTM-Network, for Classes 1 to 5, using the geometric instantaneous screw axis invariant descriptors.

	PM6	PM7	PM8	PM9	PM10
M6	**1**	0	0	0	0
M7	0,0526	**0,4474**	0,2632	0,0132	0,2237
M8	0,0685	0,0685	**0,5616**	0,0411	0,2603
M9	0	0,0375	0,0250	**0,901**	0,0375
M10	0	0,0253	0,1519	0	**0,8228**

FIGURE 5.9: Confusion Matrix, LSTM-Network, for Classes 6 to 10, using the geometric instantaneous screw axis invariant descriptors.

From the confusion matrices, above, it can be observed that although the training data-set is very low compared to the state of the art methods, some motions: M_2, M_6, M_9, M_{10}, achieve excellent recognition rates, while the rest are still moderately recognized.

5.3 Discussion

In the present sub-section, the discussion of the results will take place. Starting from the general comparison of the overall classification accuracy results, it is presented that the distance-based recognition approaches (**DTW-exponential, DTW-KNN**) outperform the deep learning approach (**LSTM-network**), in all cases.

In **all approaches**, regardless of higher or lower classification results, a common attribute lies on the invariance of the descriptors. As mentioned in chapter: 1 and furtherly elaborated in chapter: 3, the Frenet-Serret and instantaneous screw axis time-based descriptors, offer invariance with respect to the camera viewpoint and the reference frame on the rigid body, while being dependent of time. By transforming the time-based into geometric invariant descriptors, also invariance with respect to the execution time of the motion is achieved. Finally, the dimensionless geometric

invariant descriptors provide additional independence with respect to the velocity profile and the amplitude of the motion. Taking the above into consideration, it is shown by the recognition results that the more invariant properties a descriptor has, the better classification ratios it achieves, regardless the approach.

In the **DTW-KNN** recognition approach, all the classes were classified in a very good ratio. The results that are presented in the confusion matrices, take into account the six (6) invariant signals depending on the descriptor type (FS or ISA). Sometimes, due to singularities, the six (6) feature signals, do not represent a motion in a full extent, since some invariants, for example, become zero if a motion is executed on a straight line. As a result, when removing some invariants completely, the classification results improve even more, depending on the motion class. Nevertheless, the results that were presented, consider all six (6) invariant signals, so as robustness, with respect to future motion that are going to be recognized, is achieved.

Additionally, the overall classification accuracy can be also increased by alternating the k-neighbors scalar parameter, in each case. For consistency reasons, once more, it was decided that for all Frenet-Serret invariant descriptors will be classified under $k = 4$, and instantaneous screw axis descriptors under $k = 3$.

Finally, the **DTW-KNN** approach, evidently, outperforms the **KNN-exponential** approach in the following invariants:

- Dimensionless Instantaneous Screw Axis invariant descriptors.
- Geometric Frenet-Serret invariant descriptors.
- Time-based Frenet Serret invariant descriptors.

while similar results are achieved in the remaining invariants.

In the **LSTM**-approach, although motions achieved an average recognition rate, some motions were still recognized in an very good ratio, like motion-2 (M_2) that has achieved a 96.91% accuracy and also other classes, as presented in: 5.6. Again in this approach, the most invariant descriptors achieve higher recognition rates than their less invariant counterparts.

At this point, it should be taken into consideration that the network was trained with a very limited percentage of training data, resulting in a very moderate classification ratio with respect to the amount of the data provided to train the LSTM-network.

In addition, a first step to the future extension content of this thesis, lies in the recognition results that were achieved when splitting the ten (10) classes to be recognized in half, and then recognize each set separately through the LSTM-network. Through this action, the results that were achieved, a comparable basis was achieved, with the two-previous methods (DTW-exponential, DTW-KNN). Taking the later

into consideration, a first step is achieved for recognizing the motion of a rigid body through deep-learning techniques with very limited training data.

5.4 Conclusion

In the present chapter the recognition results of the distance-based (DTW-exponential, DTW-KNN) and the deep-learning (LSTM) recognition approaches were presented and discussed. While in all cases, the amount of training data that was used to train the corresponding algorithm was the same so the results between the approaches could be comparable, the distance-based approaches outperformed the deep-learning LSTM network. Followed by the presentation and the comparison of the results, comes the concluding chapter, in which the contributions of this thesis will be discussed, as well as the future work that is taken into consideration.

Chapter 6

Conclusion and Future Work

The overall objective of the thesis is to achieve robust classification rates, through different types of recognition approaches, for an alternative representation of rigid-body motions, instead of using a set of three-dimensional set of spatial points to represent a trajectory, as it is commonly used in the literature. The motion trajectories, in the current work, are represented through two description types, the Frenet-Serret and the instantaneous screw axis descriptors. Depending on the parameterization of the descriptors (time-based, geometric, dimensionless geometric), different levels of invariance is achieved, which has an immediate effect on the recognition of the motions. For all the recognition approaches that are considered, the data-set that was used to train the corresponding algorithms consists of the same amount of data, which is all the motions under a normal execution speed and recorded from the first camera view-point. Taking the later into consideration, a secondary objective of the thesis is to achieve high overall classification accuracy of motions while using a very limited amount of training data (about 8.33% of the whole data-set).

In the last, concluding, chapter of this thesis, first the contributions of this thesis will be recalled and discussed. Additionally, the second part of this chapter is dedicated to the future work propositions to extend the content and the research of the work.

6.1 Contributions

6.1.1 Comparison of Invariant Descriptor Types under Different Parameterizations

The two invariant ways of describing a rigid-body motion trajectory consist of the Frenet-Serret and the Instantaneous Screw Axis invariant descriptors. Each description offers different invariant properties, under different parameterizations, for a recorded motion as mentioned in Chapter 1 and elaborated in Chapter 3. It is evident that for the time-based descriptors, invariance is achieved with respect to the recording view-point of the camera and the choice of reference on the rigid body that

executes the motion but still dependence with respect to time is present. In addition, while eliminating the time from the description of the motion, by transforming the time-based to geometric descriptors, further invariance is achieved due to the elimination of time, so the execution time of the motion is no longer considered as an influencing factor to the description of the motion. While further parameterizing the descriptors to dimensionless geometric, additional dependencies are eliminated such as velocity profile and the motion execution length, without losing the invariance that was achieved during the previous parameterizations.

6.1.2 Juxtaposition of Descriptors over Novel Classification Techniques for Rigid-Body Trajectories

While being inspired by the state-of-the-art distance-based and deep learning classification approaches that were explained in Chapter 2, it was decided to use an alternative K-nearest neighbor approach, while replacing the traditional euclidean distance between the features of the trials, with the dynamic time-warp distance of the points in each trial. Through this approach, an additional invariance is achieved with respect to the execution time of the motion, since through DTW similarity between points is detected, before the trials are classified through DTW-KNN. With the application of the current method, it was determined that for both types of descriptors (FS, ISA) the motions were robustly recognized. Depending on the parameterization of each invariant descriptor type, and as a result the amount of invariant properties that characterize it, it was shown by Tables: 5.4, 5.5, that the more invariant properties a descriptor has, the highest classification accuracy it can achieve. Intuitively, the dimensionless geometric invariants (FS/ISA) achieved the highest results, followed by geometric (FS/ISA) and the time-based (FS/ISA). In addition, also with the second approach that was followed (LSTM-network), inspired by the properties of LSTM networks, that were encountered in the past, that except from being able to handle variable length of motion trials, it can accept as input sequential data, which, in this case, with the right manipulation of the data-set the data were reformulated into sequences as explained in Chapter 4. The recognition results of this approach, validate the results of the DTW-KNN approach. In other words, also with this deep-learning approach, the invariance of the descriptors played a significant role to the results. The later, can be observed in Table 5.7, where again the most invariant descriptors achieve the highest results.

6.1.3 Elimination of Large Training Data Required using Invariant Descriptors

In every recognition method that is followed in the present thesis work, the same amount of input data is used in order to train the corresponding classification model (1/12 or 8.33% of the whole data-set). The first reason behind this action lies in the robust presentation and comparability of the results behind the methods. Additionally, the second, and most important, reason, lies in the invariant properties of each descriptor. It was proven through the confusion matrices 5.1 and 5.6 ,that

were presented for dimensionless geometric FS invariants, for example, that in both cases (DTW-KNN, LSTM), the motions were recognized properly and with high classification results, while a minimal and similar training data-set was provided. An additional, but important comment, is that when constructing the programming code of the classification algorithms, some additional tests were executed (to double/ triple the training data-set), which resulted in higher recognition results also for the other parameterizations of the invariant descriptors. To conclude, it is crucial to comment on the fact that although a regular recurrent neural network or an LSTM would require a huge amount of training data to train and then classify the motions, the invariant properties of the descriptors eliminate that need, which is also proven by the confusion matrix 5.6, where with very limited-training data, most of the dimensionless FS invariant motions were classified in a robust way.

6.2 Future Work

6.2.1 Computational Efficiency of the Optimized Approach for Better Recognition Rates

In the present developed recognition algorithms the most popular calculation scheme for the invariant descriptors is through analytical formulas. Although the optimized approach, in which the construction of the invariants is done by minimizing the difference between the measured and the descriptor-represented trajectory, offers more robust invariant descriptors and eliminates most of the singularities in the trajectory, the calculation efficiency is very low.

A future proposition is to reconstruct a minimization problem with different regularizations and weights manipulation, such as a quicker calculation of the optimized invariants can be possible. In this way, except from robust descriptors, also more robust classification results will emerge, through the proposed recognition approaches, as explained in the contributions section of this chapter, with lower computational cost.

6.2.2 Segmentation of Invariant Signals and Recognition through a Convolutional Neural Network

A possible future work extension of the current thesis, lies in the segmentation of the invariant signals and through the local information in each segmented point in the trajectory, a construction of a convolutional neural network for motion classification.

As explained, the six (6) invariant descriptor signals, regardless of the descriptor type (FS, ISA), can be seen as time-series signals. A future work proposition is based on the division of each time-signal into an equal number of smaller n-sub-signals. Each sub-signal contains local information like curvature at each discrete time-point (t_p) of the motion. As a result, each time-point in the motion trajectory, corresponds to a specific curvature value, for each descriptor signal.

The reformulation of the curvature values of each point t_p into an image-based structure, is possible if for an image of size: $t_p \times t_p$, where at each cell: $C_{tp,tp}$, the corresponding curvature value is assigned. If the same procedure executed for each one of the six (6) invariant descriptor signals, a six-dimensional image structure will be constructed as the first layer of a convolutional neural network, which can be trained and tested with a recorded motion data-set.

Appendices

Appendix A

Invariant Descriptors for every type and parameterization, over ten demonstration trials

A.1 ISA and FS Invariant Descriptors

Geometric A.1 and A.2 dimensionless geometric ISA-descriptors, normally executed (Gewoon), from the first camera view-point, over ten demonstration trials, for the "Shaker" motion:

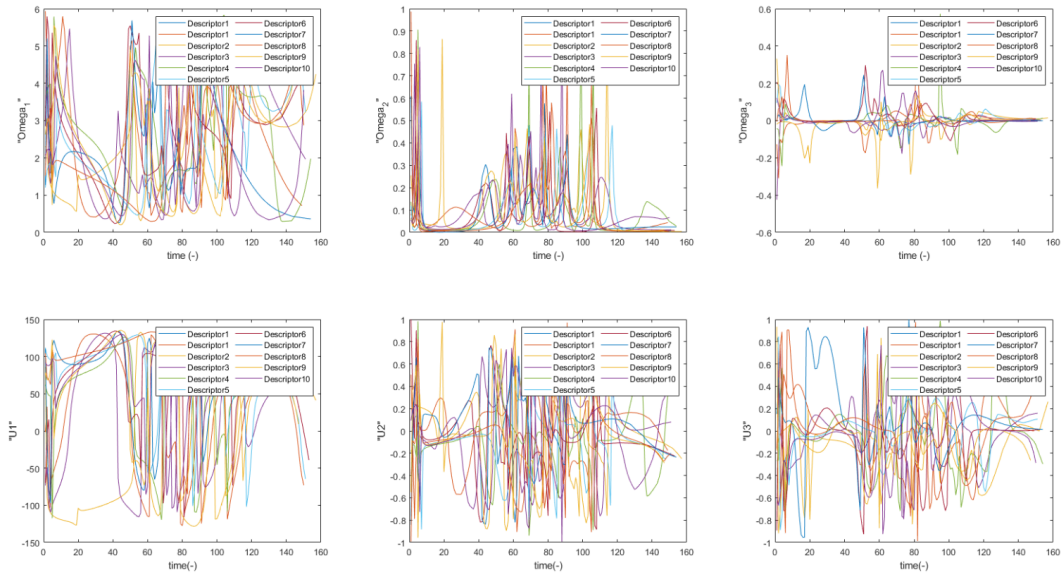


FIGURE A.1: ISA-geometric-descriptors in the order: $\omega_1, \omega_2, \omega_3, v_1, v_2, v_3$ for the "Shaker" motion, over 10 trials.

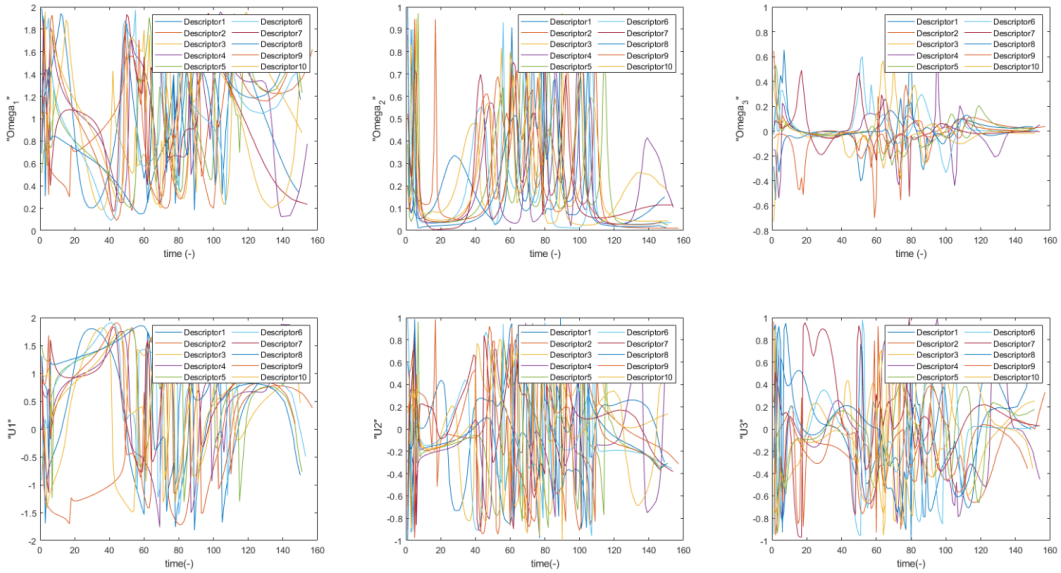


Figure A.2: ISA dimensionless geometric descriptors in the order: $\omega_1, \omega_2, \omega_3, v_1, v_2, v_3$ for the "Shaker" motion, over 10 trials.

Geometric A.3 and A.4 dimensionless geometric FS-descriptors, normally executed (Gewoon), from the first camera view-point, over ten demonstration trials, for the "Tas scheppen en uitgieten" motion:

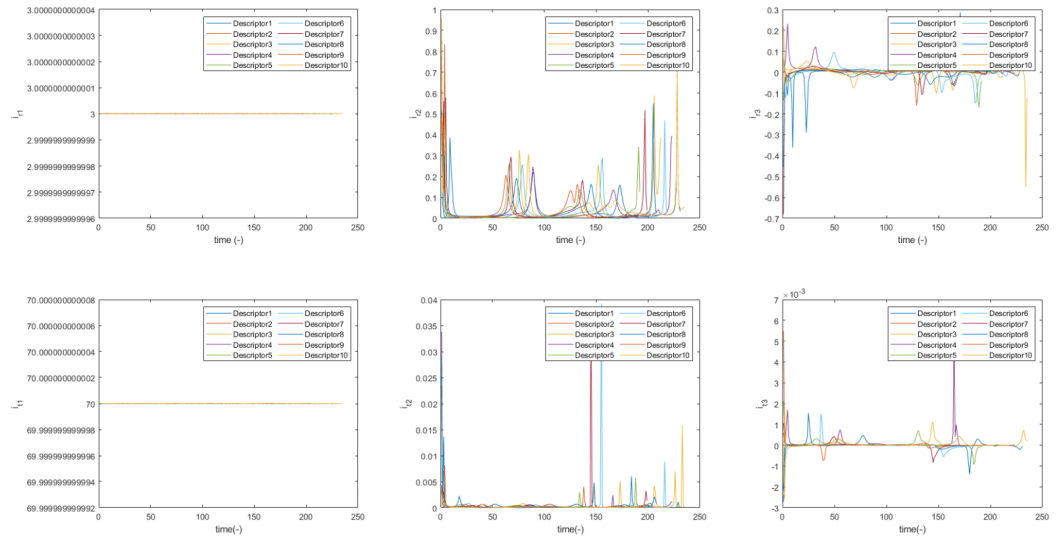


Figure A.3: FS geometric descriptors in the order: $i_{r1}, i_{r2}, i_{r3}, i_{t1}, i_{t2}, i_{t3}$ for the "Tas scheppen en uitgieten" motion, over 10 trials.

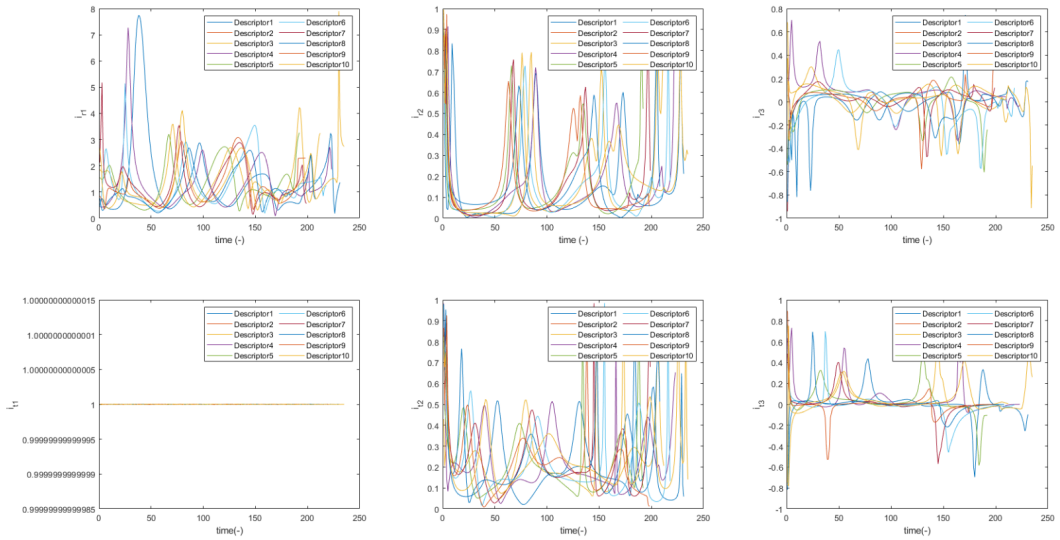


FIGURE A.4: FS dimensionless geometric descriptors in the order: $i_{r1}, i_{r2}, i_{r3}, i_{t1}, i_{t2}, i_{t3}$ for the "Tas scheppen en uitgieten" motion, over 10 trials.

Appendix B

DTW-KNN Confusion Matrices for ISA-Descriptors

The confusion matrices for the dimensionless geometric B.1, geometric B.2 and time-based B.3 ISA-descriptors with k=3 and overall accuracy: $74.67\%, 64, 6\%, 50.15\%$, respectively.

	PM1	PM2	PM3	PM4	PM5	PM6	PM7	PM8	PM9	PM10
M1	**0,432**	0,0734	0,1101	0,0367	0	0	0,0826	0,1379	0,0092	0,1193
M2	0	**1**	0	0	0	0	0	0	0	0
M3	0	0,0112	**0,7640**	0	0	0	0,1011	0,0787	0,0449	0
M4	0	0	0,0291	**0,8738**	0	0	0,0097	0	0,0874	0
M5	0	0	0	0,046	**0,6667**	0,0920	0	0	0,1954	0
M6	0	0	0	0	0	**1**	0	0	0	0
M7	0	0	0,2128	0,0426	0	0,0106	**0,6809**	0	0,0532	0
M8	0	0,0215	0,2688	0	0	0	0,0753	**0,6237**	0	0,0108
M9	0	0	0,01	0,09	0	0	0	0	**0,90**	0
M10	0,08	0,25	0,1300	0	0	0	0,04	0,05	0	**0,51**

FIGURE B.1: Confusion Matrix for ten (10) motion classes with dimensionless geometric ISA invariant descriptors.

	PM1	PM2	PM3	PM4	PM5	PM6	PM7	PM8	PM9	PM10
M1	**0,1193**	0,4679	0,0917	0,0642	0	0	0,0642	0,1376	0,0367	0,0183
M2	0	**0,949**	0,0102	0	0	0,0204	0	0	0,0204	0
M3	0	0,0449	**0,6742**	0,0449	0	0	0,0674	0,1236	0,0449	0
M4	0	0,0097	0,0097	**0,5825**	**0**	0,0097	0,0291	0,0777	0,2816	0
M5	0	0	0	0,092	0,6437	0,046	0,0115	0,0115	0,1954	0
M6	0	0	0	0	0	**1**	0	0	0	0
M7	0	0,0532	0,1489	0,0532	0	0	**0,6064**	0,0319	0,1064	0
M8	0	0,0323	0,1720	0	0	0	0,086	**0,6559**	0,043	0,0108
M9	0	0	0,01	0,03	0,01	0,01	0	0	**0,94**	0
M10	0	0,44	0,16	0	0	0	0,04	0,04	0,01	**0,31**

Figure B.2: Confusion Matrix for ten (10) motion classes with geometric ISA invariant descriptors.

	PM1	PM2	PM3	PM4	PM5	PM6	PM7	PM8	PM9	PM10
M1	**0,1831**	0,0459	0	0,0459	0	0,0459	0,0183	0,1101	0	0,5505
M2	0	**0,4898**	0,2653	0,051	0,0102	0	0,1429	0	0	0,0408
M3	0	0	**0,2697**	0,1348	0	0,0787	0,4045	0,0899	0,0225	0
M4	0	0	0	**0,7864**	0	0,1359	0,0485	0,0097	0,0194	0
M5	0,023	0	0,0115	0,0115	**0,5862**	0,2989	0,0115	0	0,0575	0
M6	0	0	0	0,0182	0,0091	**0,9727**	0	0	0	0
M7	0	0,0106	0,1489	0,2766	0,0213	0,0426	**0,4574**	0,0106	0,0319	0
M8	0	0	0,0538	0,1613	0	0,2043	0,0215	**0,5161**	0,0430	0
M9	0	0	0,01	0,19	0,2	0,09	0,07	0	**0,44**	0
M10	0	0,17	0,09	0,13	0	0,11	0,09	0,12	0,02	**0,27**

Figure B.3: Confusion Matrix for ten (10) motion classes with time-based ISA invariant descriptors.

Bibliography

[1] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[2] F. I. Bashir, A. A. Khokhar, and D. Schonfeld. View-invariant motion trajectory-based activity classification and recognition. *Multimedia Systems*, 12(1):45–54, 2006.

[3] S. Bellens, R. Smits, E. Aertbelien, H. Bruyninckx, J. De Schutter, et al. Geometric relations between rigid bodies (part 1): Semantics for standardization. *IEEE Robotics & Automation Magazine*, 20(1):84–93, 2013.

[4] A. Bostrom and A. Bagnall. A shapelet transform for multivariate time series classification. *arXiv preprint arXiv:1712.06428*, 2017.

[5] F.-L. Chung, T. C. Fu, R. Luk, and V. Ng. Flexible time series pattern matching based on perceptually important points. 2001.

[6] J. De Schutter. Invariant description of rigid body motion trajectories. *Journal of Mechanisms and Robotics*, 2(1), 2010.

[7] H. Deng, G. Runger, E. Tuv, and M. Vladimir. A time series forest for classification and feature extraction. *Information Sciences*, 239:142–153, 2013.

[8] R. A. Dunne and N. A. Campbell. On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function. In *Proc. 8th Aust. Conf. on the Neural Networks, Melbourne*, volume 181, page 185. Citeseer, 1997.

[9] Y. Endo, H. Toda, K. Nishida, and J. Ikedo. Classifying spatial trajectories using representation learning. *International Journal of Data Science and Analytics*, 2(3-4):107–117, 2016.

[10] M. Ferreira, L. Rocha, P. Costa, and A. P. Moreira. Stereoscopic vision system for human gesture tracking and robot programming by demonstration. In *International Workshop on Robotics in Smart Manufacturing*, pages 82–90. Springer, 2013.

[11] M. Flynn, J. Large, and T. Bagnall. The contract random interval spectral ensemble (c-rise): the effect of contracting a classifier on accuracy. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 381–392. Springer, 2019.

[12] I. K. M. Jais, A. R. Ismail, and S. Q. Nisa. Adam optimization algorithm for wide and deep neural network. *Knowl. Eng. Data Sci.*, 2(1):41–46, 2019.

[13] P. Senin. Dynamic time warping algorithm review. *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*, 855(1-23):40, 2008.

[14] A. Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020.

[15] D. Sinwar and R. Kaushik. Study of euclidean and manhattan distance metrics using simple k-means clustering. *Int. J. Res. Appl. Sci. Eng. Technol*, 2(5):270–274, 2014.

[16] S. Visa, B. Ramsay, A. L. Ralescu, and E. Van Der Knaap. Confusion matrix-based feature selection. *MAICS*, 710:120–127, 2011.

[17] M. Vochten. Invariant representations of rigid-body motion trajectories with application to motion recognition and robot learning by demonstration. 2018.

[18] M. Vochten, T. De Laet, and J. De Schutter. Comparison of rigid body motion trajectory descriptors for motion representation and recognition. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3010–3017. IEEE, 2015.

[19] Y. Wu, K. Ianakiev, and V. Govindaraju. Improved k-nearest neighbor classification. *Pattern recognition*, 35(10):2311–2318, 2002.

[20] L. Yan, Y. Liu, and Y. Liu. Interval feature transformation for time series classification using perceptually important points. *Applied Sciences*, 10(16):5428, 2020.