A Bidirectional Invariant Representation of Motion for Gesture Recognition and Reproduction

Raffaele Soloperto*, Matteo Saveriano[†] and Dongheui Lee[†]

Abstract— **Human action representation, recognition and learning is of importance to guarantee a fruitful human-robot cooperation. In this paper, we propose a novel coordinate-free, scale invariant representation of 6D (position and orientation) motion trajectories. The advantages of the proposed invariant representation are twofold. First the performance of gesture recognition can be improved thanks to its invariance to different viewpoints and different body sizes of the actors. Secondly, the proposed representation is bi-directional. Not only the original Cartesian trajectory can be converted into the 6 invariant values, but also the motion in the original space can be retrieved back from the invariants. While the former aspect handles robust human gesture recognition, the latter allows the execution of robot motions without the need to store the Cartesian data. Experimental results illustrate the effectiveness of the proposed invariant representation for gesture recognition and accurate trajectory reconstruction.**

I. Introduction

In the foreseeable future a close daily collaboration between humans and robots will take place. To guarantee a smooth and efficient human-robot interaction, the robot needs to understand human intentions and to react in a proper and autonomous way.

We aim at making the robot able to recognize human actions and to execute new behaviors from the recognized motions [1], [2]. Gesture recognition in real scenarios is a complicated problem, since the same gesture can be performed by different people and from different view points. Thus, it is desirable to have action representations which are coordinate-free and scale invariant.

Numerous studies have addressed the problem of human gesture invariant description and recognition. Some authors proposed to calculate *affine* transformations (rotation, translation and scaling) invariant descriptors from the image coordinates. In [3], [4] invariants under affine and projective transformations are proposed. To compute these invariants, one has to track the five fixed points in the image plane during the whole gesture. The invariants under affine transformations proposed in [5] requires to track the same six points in all frames.

Other approaches are based instead on Euclidean group invariants. In [6], [7] the authors propose to represent the motion with the *spatio-temporal curvature* of the trajectory, which is invariant under roto-translations. In [8] the invariant signature of a motion is defined in terms of curvature, torsion and their first-order derivatives. A method is proposed to calculate these quantities without using high-order derivatives. The resulting representation is invariant to affine transformation and changes in the speed of the execution. In [9] a curvature-based 2D invariant representation is proposed that is invariant to roto-translations and linear scaling. In this representation the scale invariance is an intrinsic property and it is not obtained by introducing an artificial scale. The invariants in [9] are used in [10] to recognize and predict human tasks.

The aforementioned invariant representations for gesture recognition do not match our requirements for two reasons. Firstly, the previous approaches usually neglect the orientation part of the motion. Secondly, none of them can recover the Cartesian trajectory from the invariant signature. Hence, it is not possible to learn motions from the invariant signatures without storing also the Cartesian data.

In order to overcome these limitations, a 6D representation is proposed by using *Instantaneous Screw Axes* (ISA) in [11]. Two of the invariants are the linear velocity along the ISA and the rotational velocity around the ISA. The remaining four invariants approximate the motion of the ISA. The proposed approach is invariant to affine transformations, time scale and motion profile. Using these invariant representations the original motion can be recovered back, except for some special motions. However this representation requires high-order time derivatives, that are sensitive to noise. In practical cases, they cannot be estimated reliably due to the third time derivative of the position [12]. Despite its bidirectional property, the reconstruction error is not negligible.

We propose a unified approach that can be advantageous both for human motion recognition and robot motion generation. The proposed invariants are minimal, consisting of 6 values (3 for position and 3 for orientation), and invariant to affine transformations (rotation, translation and scaling). The proposed invariants are bi-directional, which allows the conversion from the Cartesian space to the *invariant space* and vice-versa, without any loss of information. In contrast to [11], our proposed representation lies at the velocity level and is less sensitive to noise. Moreover, we introduce a separation between the position related invariants and the orientation related ones. This separation reduces the singular cases, in which the original trajectory cannot be retrieved from the invariants.

The rest of the paper is organized as follows. The proposed

[†] Matteo Saveriano and Dongheui Lee are with Chair of Automatic Control Engineering, Technische Universität München, Munich, Germany matteo.saveriano@tum.de, dhlee@tum.de

* Raffaele Soloperto is with Department of Electrical, Electronic, and Information Engineering, Universitá di Bologna, Bologna, Italy soloperto.raffaele@gmail.com

invariant representation, the so-called *SoSaLe-invariants*, is described in Section II. Preliminary results on gesture recognition and motion execution are shown in Section III. Finally, we conclude with discussions of further work in Section IV. For the reader who is not familiar with the rotation vector representation, an Appendix provides formulas to compute a rotation vector from a rotation matrix.

II. Invariant Representation of Motion

A. *Position-Based Invariants*

Rigid body motions are usually described as a set of positions and orientations of a frame attached to the body (body frame) respectively to a reference (world) frame. In the Cartesian space, the position $\mathbf{p}\left(t\right)$ is, in each time instant t, the 3D vector connecting the center of the body frame with the center of the reference frame. The orientation is described by a 3×3 rotation matrix containing the components of each axis of the body frame in the world frame. Nevertheless, the minimum number of parameters needed to represent the orientation is three. In this paper we use a minimal representation of the orientation, i.e. the so-called *rotation vector* $\mathbf{r}(t)$ (see Appendix).

In our approach, we consider two frames attached to the rigid body. The first frame describes how the position changes in time, while the other describes how the orientation changes in time[1]. The position frame is shown in Fig. 1 and created as follows:

- The *x-axis* is the unit vector generated by the difference between two consecutive position vectors, hence it represents the linear velocity of the body in a unitary time:

$$\hat{\mathbf{x}}_p\left(t\right)=\frac{\mathbf{p}\left(t+\Delta t\right)-\mathbf{p}\left(t\right)}{\left\|\mathbf{p}\left(t+\Delta t\right)-\mathbf{p}\left(t\right)\right\|}=\frac{\Delta\mathbf{p}\left(t\right)}{\left\|\Delta\mathbf{p}\left(t\right)\right\|} \quad (1)$$

- The *y-axis* lies on the common normal between the *x-axis* at the current instant t and the *x-axis* at the next instant $t+\Delta t$:

$$\hat{\mathbf{y}}_p\left(t\right)=\frac{\hat{\mathbf{x}}_p\left(t\right)\times\hat{\mathbf{x}}_p\left(t+\Delta t\right)}{\left\|\hat{\mathbf{x}}_p\left(t\right)\times\hat{\mathbf{x}}_p\left(t+\Delta t\right)\right\|} \quad (2)$$

- The *z-axis* is the cross product between the *x-axis* and the *y-axis*:

$$\hat{\mathbf{z}}_p\left(t\right)=\hat{\mathbf{x}}_p\left(t\right)\times\hat{\mathbf{y}}_p\left(t\right) \quad (3)$$

The orientation frame in Fig. 2 is defined in a similar way. The *x-axis* is the normalized rotation vector[2] $\Delta\mathbf{r}(t)$:

$$\hat{\mathbf{x}}_r\left(t\right)=\frac{\Delta\mathbf{r}(t)}{\left\|\Delta\mathbf{r}(t)\right\|} \quad (4)$$

where $\Delta\mathbf{r}(t)$ represents the relative orientation between the instants $t+\Delta t$ and t. Hence, it represents the angular velocity needed to rotate the body from t to $t+\Delta t$ in a unitary time. The *y-axis* and the *z-axis* are then computed from (2), (3) by substituting $\hat{\mathbf{x}}_p$ and $\hat{\mathbf{y}}_p$ with $\hat{\mathbf{x}}_r$ and $\hat{\mathbf{y}}_r$ respectively. The direction of the axes of the position/orientation frames is chosen to avoid discontinuities (jumps of $\pm\pi$) between subsequent time instants[3].

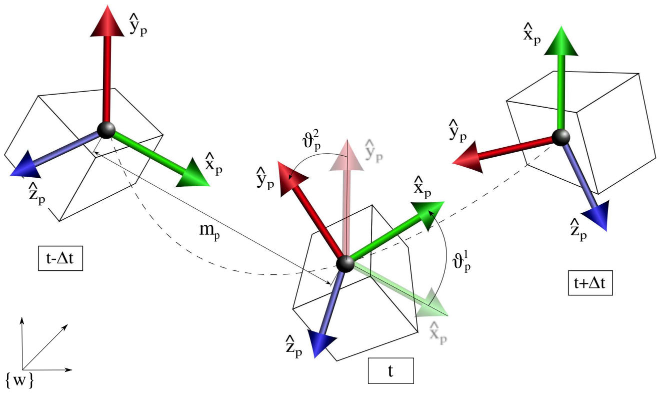


Fig. 1. Position frame in three time instants.

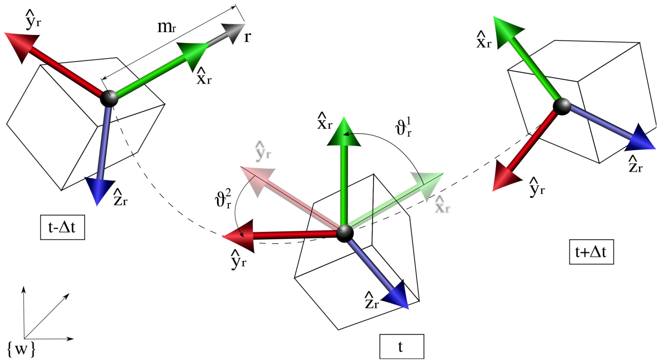


Fig. 2. Orientation frame in three time instants.

Once the frames are both defined, the six invariant values can be defined. Two invariants correspond to the norm of the relative positions and orientations between consecutive frames:

$$\begin{aligned} m_p\left(t\right) &= \left\|\Delta\mathbf{p}(t)\right\|=\Delta\mathbf{p}(t)\cdot\hat{\mathbf{x}}_p\left(t\right) && (5)\\ m_r(t) &= \left\|\Delta\mathbf{r}\left(t\right)\right\|=\Delta\mathbf{r}\left(t\right)\cdot\hat{\mathbf{x}}_r(t) && (6)\end{aligned}$$

where $\hat{\mathbf{x}}_p(t)$ and $\hat{\mathbf{x}}_r(t)$ are computed using (1) and (4) respectively. The m_p and m_r invariants in (5) and (6) describe the motion of the body. Four more values are used to describe the rotation of the position frame and orientation frame.

Let us consider the position frame. The *y-axis* lies on the common normal between two consecutive *x-axes*. According to the Denavit-Hartenberg notation [13], the frames at t and $t+\Delta t$ can be aligned considering only the rotations about the x and y axes. As illustrated in Fig. 1, the frame needs to rotate of an angle θ_p^1 about $\hat{\mathbf{y}}_p(t+\Delta t)$ in order to align $\hat{\mathbf{x}}_p(t)$ to $\hat{\mathbf{x}}_p(t+\Delta t)$, and then the frame needs to rotate of θ_p^2 about $\hat{\mathbf{x}}_p(t+\Delta t)$ in order to align $\hat{\mathbf{y}}_p(t)$ to $\hat{\mathbf{y}}_p(t+\Delta t)$.

[1] In the formulas, the following notation is used: the subscript p refers to the position, while r refers to the orientation.

[2] As detailed in Appendix the rotation vector represents the orientation as an axis and an angle (amount of rotation around the axis).

[3] We simply compute the angle α between the two axis an invert the second axis if $\alpha\geq\pi/2$.

Hence, two more invariants for the position are defined as:

$$\theta_p^1(t) = \arctan\left(\frac{\hat{\mathbf{x}}_p(t) \times \hat{\mathbf{x}}_p(t+\Delta t)}{\hat{\mathbf{x}}_p(t) \cdot \hat{\mathbf{x}}_p(t+\Delta t)} \cdot \hat{\mathbf{y}}_p(t)\right) \quad (7)$$

$$\theta_p^2(t) = \arctan\left(\frac{\hat{\mathbf{y}}_p(t) \times \hat{\mathbf{y}}_p(t+\Delta t)}{\hat{\mathbf{y}}_p(t) \cdot \hat{\mathbf{y}}_p(t+\Delta t)} \cdot \hat{\mathbf{x}}_p(t+\Delta t)\right) \quad (8)$$

Following a similar reasoning, two more invariants for the orientation are defined as:

$$\theta_r^1(t) = \arctan\left(\frac{\hat{\mathbf{x}}_r(t) \times \hat{\mathbf{x}}_r(t+\Delta t)}{\hat{\mathbf{x}}_r(t) \cdot \hat{\mathbf{x}}_r(t+\Delta t)} \cdot \hat{\mathbf{y}}_r(t)\right) \quad (9)$$

$$\theta_r^2(t) = \arctan\left(\frac{\hat{\mathbf{y}}_r(t) \times \hat{\mathbf{y}}_r(t+\Delta t)}{\hat{\mathbf{y}}_r(t) \cdot \hat{\mathbf{y}}_r(t+\Delta t)} \cdot \hat{\mathbf{x}}_r(t+\Delta t)\right) \quad (10)$$

Note that (7) and (9), as well as (8) and (10), are formally the same.

B. Velocity-Based Invariants

The described procedure can be also used to compute invariants starting from the linear $\mathbf{v}$ and angular $\boldsymbol{\omega}$ velocities. The *x-axis* of the linear and angular velocity frames are computed as:

$$\hat{\mathbf{x}}_v(t) = \frac{\mathbf{v}(t)}{\|\mathbf{v}(t)\|} \qquad \hat{\mathbf{x}}_\omega(t) = \frac{\boldsymbol{\omega}(t)}{\|\boldsymbol{\omega}(t)\|} \quad (11)$$

The related invariants m_v and m_ω are computed as:

$$m_v(t) = \|\mathbf{v}(t)\| = \mathbf{v}(t) \cdot \hat{\mathbf{x}}_v(t) \quad (12)$$

$$m_\omega(t) = \|\boldsymbol{\omega}(t)\| = \boldsymbol{\omega}(t) \cdot \hat{\mathbf{x}}_\omega(t) \quad (13)$$

The other invariants θ_v^1, θ_v^2, θ_ω^1 and θ_ω^2 are simply calculated by substituting $\hat{\mathbf{x}}_p(t)$ with $\hat{\mathbf{x}}_v(t)$ in (2), and $\hat{\mathbf{x}}_r(t)$ with $\hat{\mathbf{x}}_\omega(t)$ in (3).

In the discrete time a simple relation exists between the position and the velocity-based invariants. Recalling that $\Delta\mathbf{p}$ and $\Delta\mathbf{r}$ represent respectively the linear and angular velocities between two consecutive frames in a unitary time, we have

$$m_v(t) = \|\mathbf{v}(t)\| = \frac{\|\Delta\mathbf{p}(t)\|}{\Delta t} = \frac{m_p(t)}{\Delta t} \quad (14)$$

$$m_\omega(t) = \|\boldsymbol{\omega}(t)\| = \frac{\|\Delta\mathbf{r}(t)\|}{\Delta t} = \frac{m_r(t)}{\Delta t} \quad (15)$$

From (1), (4) and (11) it is easy to verify that $\hat{\mathbf{x}}_v(t) = \hat{\mathbf{x}}_p(t)$ and $\hat{\mathbf{x}}_\omega(t) = \hat{\mathbf{x}}_r(t)$. Then, from (2), (3), it is possible to prove that $[\theta_v^1,\ \theta_v^2,\ \theta_\omega^1,\ \theta_\omega^2] = [\theta_p^1,\ \theta_p^2,\ \theta_r^1,\ \theta_r^2]$.

The velocity-based invariants have the same properties of the position-based ones, being simply obtained by dividing m_p and m_r by the sample time Δt. Nevertheless, in robot control, the inverse kinematic problem is usually solved at the velocity level using the relation [14]:

$$\dot{\mathbf{q}}(t) = \mathbf{J}^\dagger(\boldsymbol{q}) \cdot \begin{bmatrix}\mathbf{v} \\ \boldsymbol{\omega}\end{bmatrix} \quad (16)$$

where $\mathbf{J}^\dagger(\boldsymbol{q})$ is the pseudo-inverse of the Jacobian matrix. In our experiments, we adopted the velocity-based invariants, since they can be effectively used to recognize motions and to reproduce these motions on real robots. The velocity-based SoSaLe-invariants for the synthetic data in Fig. 3, are shown in Fig. 4. Linear and angular velocities are computed by numerical differentiation with a sample time of 0.1s.

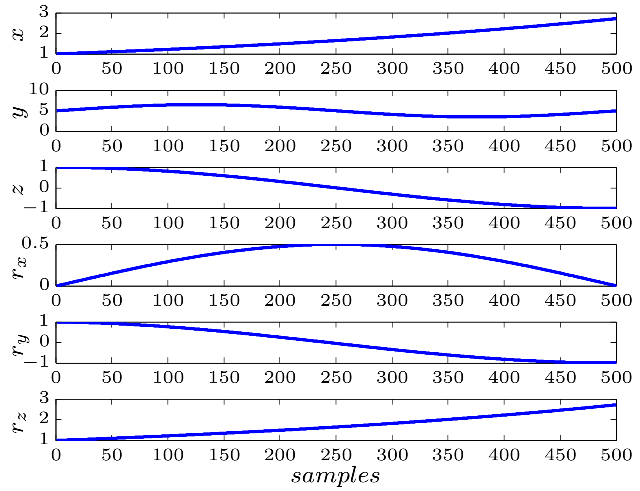


Fig. 3. Synthetic Cartesian data. Positions are in meter, orientations in radiant.

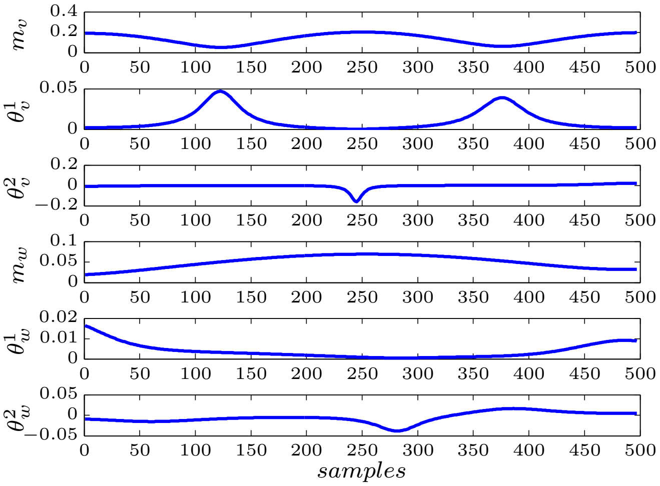


Fig. 4. Velocity-based SoSaLe-invariant representation. m_v is in m/s, m_ω in rad/s and the θ_i^j are in rad.

C. Trajectory Reconstruction

Pose Reconstruction: The rigid body pose (position and orientation) reconstruction proceeds in three steps. Firstly, the pose of the position (rotation) frame in each time instant is calculated as:

$$\mathbf{H}_p(t) = \begin{bmatrix}\mathbf{R}_y(\theta_p^1)\mathbf{R}_x(\theta_p^2) & \mathbf{m}_p \\ \mathbf{0}^T & 1\end{bmatrix} \quad (17)$$

$$\mathbf{H}_r(t) = \begin{bmatrix}\mathbf{R}_y(\theta_r^1)\mathbf{R}_x(\theta_r^2) & \mathbf{m}_r \\ \mathbf{0}^T & 0\end{bmatrix} \quad (18)$$

where $\mathbf{m}_p = [m_p\ 0\ 0]^T$, $\mathbf{m}_r = [m_r\ 0\ 0]^T$, $\mathbf{R}_y(\alpha)$ and $\mathbf{R}_x(\alpha)$ are the elementary rotations of an angle α about the y and x axis [14].

Secondly, knowing the initial pose of the position/rotation frame respectively to the world frame, namely $\mathbf{H}_p(0)$ and $\mathbf{H}_r(0)$, it is possible to compute:

$$\mathbf{H}_p^w(t) = \mathbf{H}_p(0) \cdot \mathbf{H}_p(\Delta t) \cdot \dots \cdot \mathbf{H}_p(t) \quad (19)$$

$$\mathbf{H}_r^w(t) = \mathbf{H}_r(0) \cdot \mathbf{H}_r(\Delta t) \cdot \dots \cdot \mathbf{H}_r(t) \quad (20)$$

$\mathbf{H}_p^w(t)$ and $\mathbf{H}_r^w(t)$ represent the pose of the position frame and the rotation frame respectively to the world frame in a generic time instant t.

Finally, the original position in a generic time instant t is computed as:

$$\mathbf{p}(t) = \mathbf{H}_p^w(t)_{[1:3,4]} \tag{21}$$

where $\mathbf{H}_p^w(t)_{[1:3,4]}$ are the first three elements of the fourth column of $\mathbf{H}_p^w(t)$. Following a similar approach we can also compute $\Delta\mathbf{r}(t) = \mathbf{H}_r^w(t)_{[1:3,4]}$. The original rotation matrix respectively to the world frame in a generic time instant t is then computed as:

$$\mathbf{R}^w(t) = \exp(\Delta\mathbf{r}(0)) \cdot \dots \cdot \exp(\Delta\mathbf{r}(t)) \tag{22}$$

where $\exp(\mathbf{r})$ transforms a rotation vector into a rotation matrix (see Appendix).

Velocity Reconstruction: The process to reconstruct the velocity of the rigid body motion is analogous to that used to reconstruct the position, but it requires only two steps. We give the formulas:

$$\begin{aligned}
\mathbf{H}_v(t) &= \begin{bmatrix} \mathbf{R}_y(\theta_p^1)\mathbf{R}_x(\theta_p^2) & \mathbf{m}_v \\ \mathbf{0}^T & 0 \end{bmatrix} && (23)\\
\mathbf{H}_\omega(t) &= \begin{bmatrix} \mathbf{R}_y(\theta_r^1)\mathbf{R}_x(\theta_r^2) & \mathbf{m}_\omega \\ \mathbf{0}^T & 0 \end{bmatrix} && (24)\\
\mathbf{H}_v^w(t) &= \mathbf{H}_v(0) \cdot \mathbf{H}_v(\Delta t) \cdot \dots \cdot \mathbf{H}_v(t) && (25)\\
\mathbf{H}_\omega^w(t) &= \mathbf{H}_\omega(0) \cdot \mathbf{H}_\omega(\Delta t) \cdot \dots \cdot \mathbf{H}_\omega(t) && (26)\\
\mathbf{v}(t) &= \mathbf{H}_v^w(t)_{[1:3,4]}(t) && (27)\\
\boldsymbol{\omega}(t) &= \mathbf{H}_\omega^w(t)_{[1:3,4]}(t) && (28)
\end{aligned}$$

The reconstruction error, i.e. the norm of the difference between the original linear and angular velocity (time derivative of the data Fig. 3) and the retrieved velocity from the proposed invariants (Fig. 4), is shown in Fig. 5. For comparison, the error with the reconstruction approach in [11] is about 10^{-3} m/s for the linear and about 10^{-3} rad/s for the angular velocity.

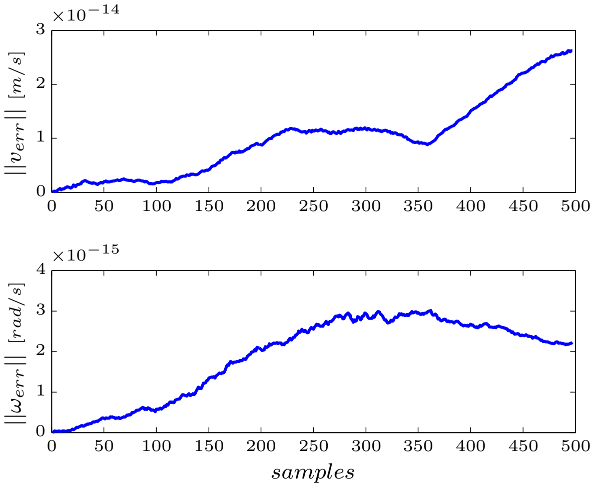


Fig. 5. Velocity reconstruction error of the proposed invariant representation.

D. Invariance Properties

In this subsection the invariance properties of the proposed representation are investigated. Due to the limited space, we focus on the velocity-based invariants in the experiments and the analysis. Note that the following properties are also valid for the position-based invariants.

Roto-translations: Translations of the reference frame do not affect the linear and angular velocity. The invariance to rotations can be easily proved through the proprieties of the norm, inner product and vector product. Lets us consider a generic rotation $\mathbf{R}$, applied to the linear velocity[4]. It is known that rotations does not affect the norm of a vector, neither the angle between vectors. Indeed, we have

$$m_v = \|\mathbf{R}\boldsymbol{\Delta}\mathbf{v}\| = \|\mathbf{R}\| \, \|\Delta\mathbf{v}\| = \|\Delta\mathbf{v}\| \tag{29}$$

where the property $\|\mathbf{R}\|$ is used, and

$$\begin{aligned}
\theta_{Rv}^1 &= \arctan\left(\frac{(\mathbf{R}\hat{\mathbf{x}}_v(t) \times \mathbf{R}\hat{\mathbf{x}}_v(t+\Delta t))}{\mathbf{R}\hat{\mathbf{x}}_v(t) \cdot \mathbf{R}\hat{\mathbf{x}}_v(t+\Delta t)} \cdot \mathbf{R}\hat{\mathbf{x}}_v(t)\right)\\
&= \arctan\left(\frac{(\hat{\mathbf{x}}_v(t) \times \hat{\mathbf{x}}_v(t+\Delta t))^T}{\hat{\mathbf{x}}_v(t)^T\mathbf{R}^T \cdot \mathbf{R}\hat{\mathbf{x}}_v(t+\Delta t)}\mathbf{R}^T \cdot \mathbf{R}\hat{\mathbf{y}}_v(t)\right) = \theta_v^1
\end{aligned} \tag{30}$$

Following the same reasoning it is easy to prove the invariance of θ_v^2.

Time, linear and angular scale: The invariance with respect to the time scale is useful to compare motions executed at different speeds. Following the approach in [11] we define a dimensionless time:

$$t' = \frac{t}{t_f} \tag{31}$$

where t_f is the duration of the motion. Invariants independent on the time scale are then obtained by multiplying the six m_n^i and θ_n^i by t_f and by substituting t with t'.

The invariance with respect to the scaling factors is of importance in gesture recognition with different users [9]. The four θ_n^i, representing angles between unit vectors, are independent on linear and angular scales. To make the two m_n^i values invariant to scaling factors, we can divide the invariants by the linear and angular scale of motion:

$$m_v'(t) = \frac{m_v(t)}{\int_{t=0}^{t_f} |m_v(t)|} \quad m_\omega'(t) = \frac{m_\omega(t)}{\int_{t=0}^{t_f} |m_\omega(t)|} \tag{32}$$

where t_f is the duration of the motion. In the discrete time case, the integral in (32) becomes the sum over all samples.

Reverse motion: In some cases it can be useful to have the same representation for motions executed in a direction or in the reverse one. To get this property m_v and m_ω should be considered in the reverse way:

$$m_v^{rev}(t) = m_v(t_f - t) \tag{33}$$

$$m_\omega^{rev}(t) = m_\omega(t_f - t) \tag{34}$$

where t_f is the duration of the motion. The remaining four invariants need not only to be considered in the reverse way,

[4] For angular velocity is analogous.

but also to be shifted by a certain value (Δt or $2\Delta t$), as follows:

$$\theta_n^{1,rev}(t) = \theta_n^1(t_f - t + \Delta t) \qquad n = v, \omega \qquad (35)$$

$$\theta_n^{2,rev}(t) = \theta_n^2(t_f - t + 2\Delta t) \qquad n = v, \omega \qquad (36)$$

Speed invariance: The invariance to the motion profile (velocity of execution) can be achieved only in theory. In real situations, due to the discrete sample time of real sensor, changes in the speed can strongly affect the measured motion and the resulting invariants [8].

Noise sensitivity: The proposed representations lies at velocity level and do not require high order derivatives of the Cartesian trajectory. Hence, in real application, a simple filtering technique (such as first-order or moving-average filters) can be adopted.

Samples delay: To compute the invariants in a time instant t one has to know the Cartesian trajectory in t and in the two next time instants. Hence, a delay of three samples is introduced. The representations in [11], instead, depend only on the current time instant, but they require the third-order derivative of the position (orientation). In real applications, time derivatives have to be computed numerically from the position level, introducing the same delay of three samples.

E. Special Motions

Singularity occurs when an axis cannot be defined. It happens when the denominator of a function results equal to zero or it cannot be calculated. We discuss these cases and propose effective solutions. Note that, in contrast to [11], the original motion can be reconstructed also in the singular cases.

Pure Rotation: $\hat{\mathbf{x}}_p(t)$ is set as the previous one. Hence, from (5), m_p is equal to zero. The remaining values θ_p^1 and θ_p^2 are normally computed from (7) and (8).

Pure Translation: $\hat{\mathbf{x}}_r$ is set as the previous one. Hence, from (6), m_r is equal to zero. The remaining values θ_r^1 and θ_r^2 are normally computed from (9) and (10).

Translation along a straight line: If $\hat{\mathbf{x}}_p(t)$ and $\hat{\mathbf{x}}_p(t+\Delta t)$ are coincident, $\hat{\mathbf{y}}_p(t)$ is set as the previous one, in order to be normal to $\hat{\mathbf{x}}_p(t)$ and, therefore, also normal to $\hat{\mathbf{x}}_p(t+\Delta t)$.

Rotation about parallel axes: If $\hat{\mathbf{x}}_r(t)$ and $\hat{\mathbf{x}}_r(t + \Delta t)$ are parallel $\hat{\mathbf{y}}_r(t)$ is set as the previous one, in order to be normal to $\hat{\mathbf{x}}_r(t)$ and, therefore, also normal to $\hat{\mathbf{x}}_r(t + \Delta t)$.

III. Experimental Results

In this section we compare the performances of the proposed invariants (SoSaLe-invariants) with two other invariants: SaLe-invariants in [7] and DS-invariants [8][5].

A. Synthetic data - noise sensitivity

This experiment aims at showing the robustness to noise of the SoSaLe-invariants. We use the synthetic data in Fig. 3. Linear and angular velocities and accelerations are computed by numerical differentiation with a sample time of 0.1 s.

A new series of samples is generated by adding a Gaussian noise with increasing power to the original data (decreasing *signal noise ratio* (*snr*) from 100 to 5). The noisy sequence is compared with the original one by computing their *Dynamic Time Warping (DTW)* distance. The results in Fig. 6 clearly show that our velocity-level SoSaLe-invariant representation exhibits a reduced noise sensitivity, compared to the SaLe-invariants (acceleration level) and the DS-invariants (jerk level).

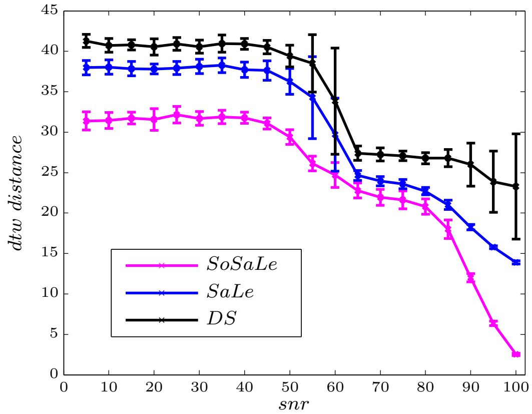
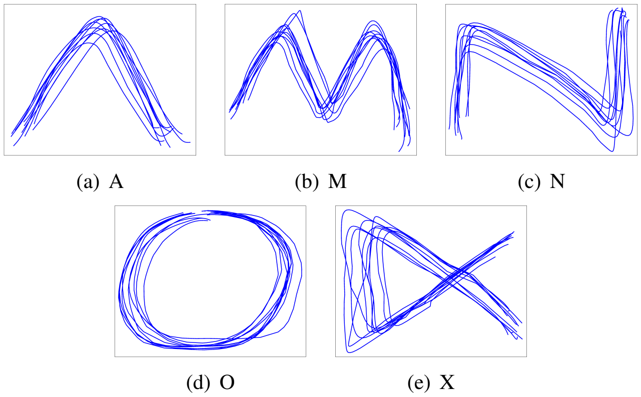


Fig. 6. DTW distances for the SaLe, SoSaLe and DS invariants when increasing Gaussian noises are applied to the original trajectory.

B. English letter dataset

In this experiment we show the recognition and reconstruction performance of the proposed representation on real data. The five capital letters A, M, N, O, X are used. As shown in Fig. 7, ten repetitions for each letter are provided. Data are collected from one user by tracking his right hand position at 30 Hz with an RGB-D camera[6].



(a) A

(b) M

(c) N

(d) O

(e) X

Fig. 7. The 10 repetition of the English letters collected from a human demonstrator.

1) Recognition: To show the benefits of the invariance to affine transformations, we extend the dataset by adding forty repetitions for each letter. These series are obtained

[5] For simplicity, we name these invariants after the names of authors: SoSaLe (SOloperto SAveriano Lee), SaLe (SAveriano LEe) and DS (De Schutter).

[6] The hand tracking is performed using the OpenNI (openni.org) library. The used version (OpenNI v1.5.4) does not provide the hand orientation.

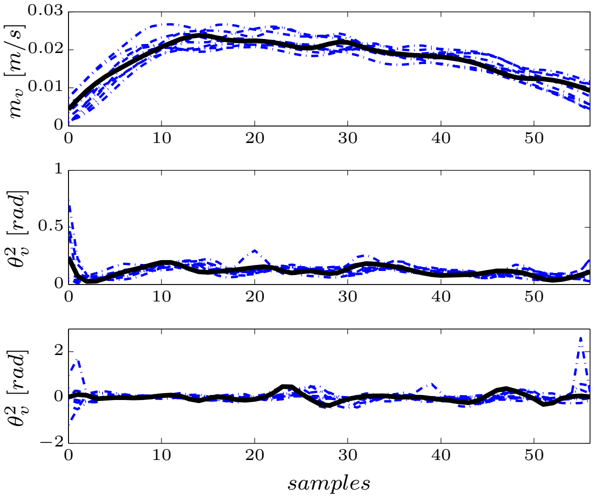


Fig. 8. SoSaLe-invariant representation for the letter O. The blue dot dashed lines represent the invariants for each demonstration. Only 10 lines are visible due to the invariance to affine transformations. The black solid line is the model obtained with the *dynamic time warping* approach in [9].

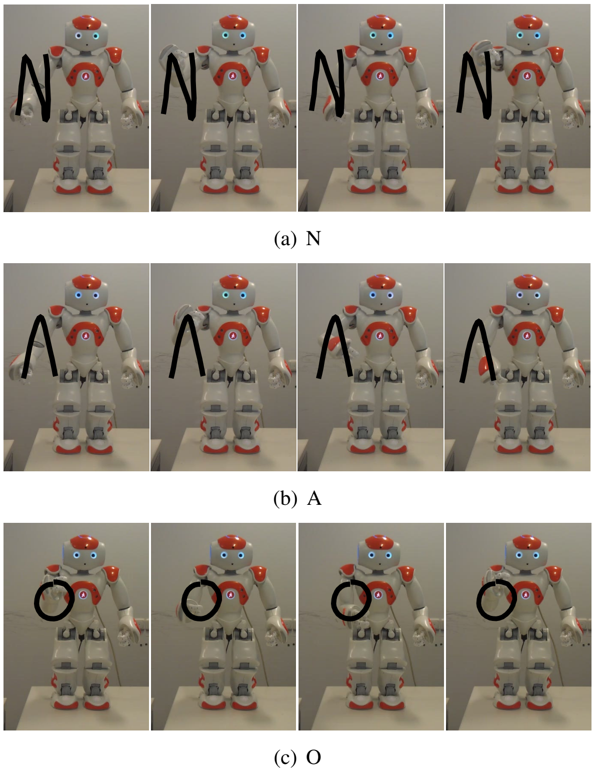
by applying a random affine transformation (rotations, translations and scaling are in the interval $[-1,\ 1]$) to the original data. After this procedure, the dataset consists of 50 demonstrations for each letter.

We tested the proposed SoSaLe-invariants against the SaLe-invariants and the DS-invariants. For the recognition we used the DTW-based averaging approach in [12], that shows high recognition rates also for complex datasets [9]. The results of the training procedure for the letter O are shown in Fig. 8. The results, obtained using half of the samples for training and the rest for testing, are shown in Fig. 10. The proposed approach outperform both the SaLe-invariants and the DS-invariants. This is probably due to the higher noise sensitivity that the SaLe-invariants and the DS-invariants have with respect to the SoSaLe-invariants.

2) Reproduction: In this experiment, we make use of the NAO humanoid robot to reproduce the English letters dataset. The trained models of each letter are transformed into Cartesian references for the robot, as discussed in Sec. II-C. In this dataset only positions are considered. Hence, we are in one of the special cases, namely the pure translation, in Sec. II-E. Note that in the case of pure translations the DS-invariants cannot be converted to a Cartesian trajectory. The trajectory reconstructed using the “human” linear scale $S_{hum} = \int_{t=0}^{t_f} |m_\omega(t)|$ in (32) cannot be executed on the NAO robot, due to its physical limitations. A trajectory suitable for the NAO robot can be generated using a linear scale s_{nao} smaller than S_{hum}. The value of $s_{nao} = 1/12$ is chosen considering the NAO’s arm length and the maximum allowed speed. The results of this simple but effective scaling technique are shown in Fig. 9, where the robot reproduces the letter sequence N-A-O.

IV. Conclusion and Future Work

This paper proposes a new invariant representation, called SoSaLe-invariants, for robust human motion recognition and robot motion reproduction. Two types of the SoSaLe-invariants (position-level and velocity-level) are introduced. The representation is minimal (six values), invariant to changes in the reference frame, reverse motions, linear, angular and time scale. The representation is also bi-directional, giving the possibility to accurately reconstruct the Cartesian trajectory directly from the invariant trajectory. Analytical formulas are provided to compute the invariant values, and to reconstruct the original motion. The proposed approach presents a clear division between position and orientation of the rigid body trajectory, which makes this solution more practical for different applications. Singular cases can be simply detected and their solutions are provided. Compared to state-of-the-art approaches our representation presents a simpler formulation and it is less sensitive to noise. As future work, we plan to extend the approach from one rigid body motion to motion of articulated bodies.



(a) N

(b) A

(c) O

Fig. 9. Snapshots of the reproduction of the letter sequence N-A-O.

Appendix

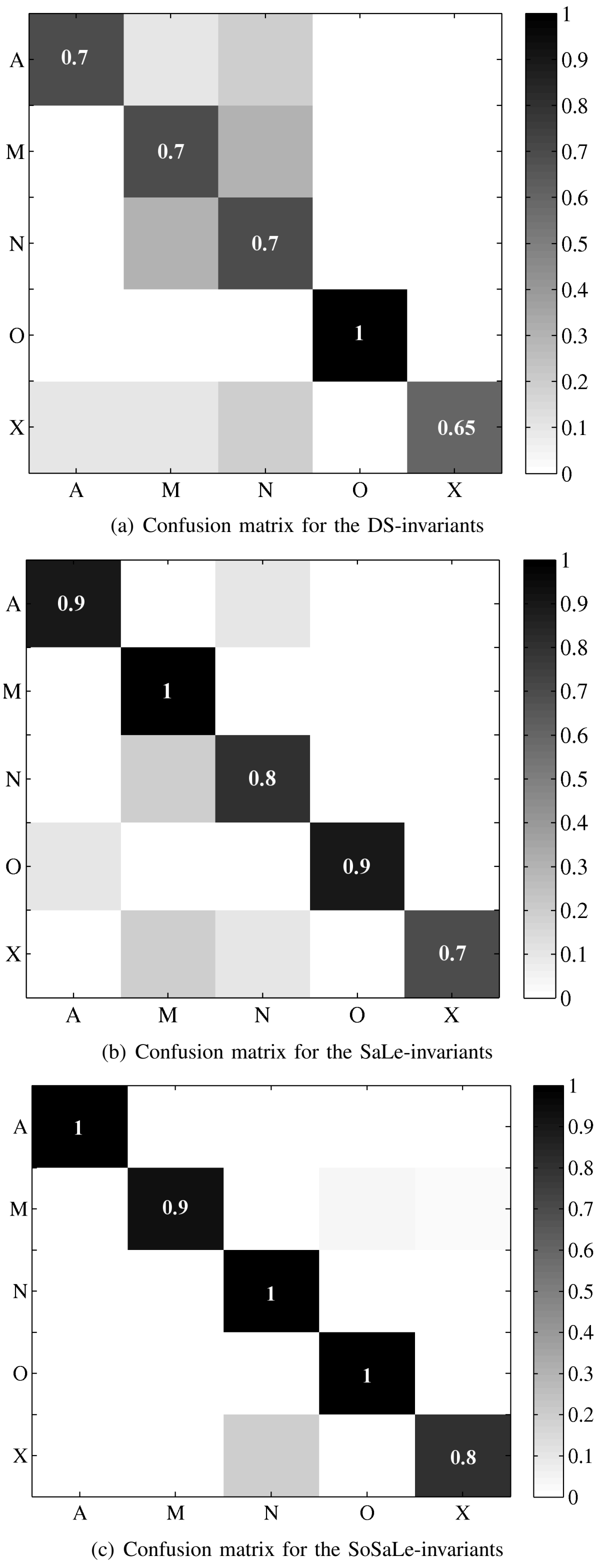
Given a rotation matrix $\mathbf{R}$ the relative rotation vector $\mathbf{r}$ is compute as:

$$\mathbf{r} = \theta\hat{\mathbf{r}}\ , \tag{37}$$

where

$$\theta = \arccos\left(\frac{trace\,(\mathbf{R}) - 1}{2}\right)\ , \tag{38}$$

$$\hat{\mathbf{r}} = \frac{1}{2\sin\theta}\cdot\begin{bmatrix}\mathbf{R}\,(3,2) - \mathbf{R}\,(2,3)\\ \mathbf{R}\,(1,3) - \mathbf{R}\,(3,1)\\ \mathbf{R}\,(2,1) - \mathbf{R}\,(1,2)\end{bmatrix}\ . \tag{39}$$



(a) Confusion matrix for the DS-invariants

(b) Confusion matrix for the SaLe-invariants

(c) Confusion matrix for the SoSaLe-invariants

Fig. 10. Recognition results for the English letters dataset.

The unit vector $\hat{\mathbf{r}}$ represents the axis around the rigid body has rotated of an angle θ.

Given a rotation vector $\mathbf{r}$ the relative rotation matrix $\mathbf{R}$ is computed using the *exponential map*:

$$\mathbf{R} = \exp(\mathbf{r}) = \mathbf{I} + \frac{\mathbf{S}(\mathbf{r})}{\theta}\sin(\theta) + \frac{\mathbf{S}^2(\mathbf{r})}{\theta^2}(1-\cos(\theta)) \ , \quad (40)$$

where the skew-symmetric matrix $\mathbf{S}(\mathbf{r})$ is given by:

$$\mathbf{S}(\mathbf{r}) = \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} . \quad (41)$$

Acknowledgments

This work has been partially supported by the European Community within the FP7 ICT-287513 SAPHARI project and Technical University of Munich, Institute for Advanced Study, funded by the German Excellence Initiative.

References

[1] D. Lee and Y. Nakamura, “Mimesis model from partial observations for a humanoid robot,” *The International Journal of Robotics Research*, vol. 29, no. 1, pp. 60–80, 2010.

[2] D. Lee and C. Ott, “Incremental kinesthetic teaching of motion primitives using the motion refinement tube,” *Autonomous Robots*, vol. 31, no. 2, pp. 115–131, 2011.

[3] Y. Piao, K. Hayakawa, and J. Sato, “Space-time invariants and video motion extraction from arbitrary viewpoints,” in *Proceedings of the IEEE International Conference on Pattern Recognition*, 2002, pp. 56–59.

[4] ——, “Space-time invariants for recognizing 3d motions from arbitrary viewpoints under perspective projection,” in *Proceedings of the IEEE International Conference on Image and Graphics*, 2004, pp. 200–203.

[5] A. Zisserman and S. Maybank, “A case against epipolar geometry,” in *Applications of Invariance in Computer Vision*. Springer Berlin / Heidelberg, 1994, pp. 69–88.

[6] C. Rao, A. Yilmaz, and M. Shah, “View-invariant representation and recognition of actions,” *International Journal of Computer Vision*, pp. 203–226, 2002.

[7] C. Rao, M. Shah, and T. S. Mahmood, “Action recognition based on view invariant spatio-temporal analysis,” in *ACM Multimedia*, 2003.

[8] S. Wu and Y. F. Li, “On signature invariants for effective motion trajectory recognition,” *International Journal of Robotic Research*, vol. 27, no. 8, pp. 895–917, 2008.

[9] M. Saveriano and D. Lee, “Invariant representation for user independent motion recognition,” in *Proceedings of the IEEE/International Symposium on Robot and Human Interactive Communication*, 2013, pp. 650–655.

[10] V. Magnanimo, M. Saveriano, S. Rossi, and D. Lee, “A bayesian approach for task recognition and future human activity prediction,” in *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication*, 2014, pp. 726–731.

[11] J. De Schutter, “Invariant description of rigid body motion trajectories,” *Journal of Mechanisms and Robotics*, vol. 2, no. 1, pp. 1–9, 2010.

[12] J. De Schutter, E. Di Lello, J. De Schutter, R. Matthysen, T. Benoit, and T. De Laet, “Recognition of 6 dof rigid body motion trajectories using a coordinate-free representation,” in *IEEE International Conference on Robotics and Automation*, 2011, pp. 2071–2078.

[13] J. Denavit and R. S. Hartenberg, “A kinematic notation for lower-pair mechanisms based on matrices,” *Transaction of the ASME Journal of Applied Mechanics*, vol. 22, no. 2, pp. 215–221, 1965.

[14] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics - Modelling, Planning and Control*. Springer, 2009.