Noname manuscript No.
(will be inserted by the editor)

Bidirectional Invariant Representation of Rigid Body Motions and its Application to Gesture Recognition and Reproduction

Dongheui Lee · Raffaele Soloperto · Matteo Saveriano

Received: date / Accepted: date

Abstract In this paper we propose a new bidirectional invariant motion descriptor of a rigid body. The proposed invariant representation is not affected by rotations, translations, time, linear and angular scaling. Invariant properties of the proposed representation enable to recognize gestures in realistic scenarios with unexpected variations (e.g., changes in user's initial pose, execution time or an observation point), while Cartesian trajectories are sensitive to these changes. The proposed invariant representation also allows reconstruction of the original motion trajectory, which is useful for human-robot interaction applications where a robot recognizes human actions and executes robot's proper behaviors using same descriptors. By removing the dependency on absolute pose and scaling factors of the Cartesian trajectories the proposed descriptor achieves flexibility to generate different motion instances from the same invariant representation. In order to illustrate the effectiveness of our proposed descriptor in motion recognition and generation, it is tested on three datasets and experiments on a NAO humanoid robot and a KUKA LWR IV+ manipulator and compared with other existing invariant representations.

Keywords Invariant representation · Rigid body motion · Bidirectional descriptor · Recognition · Generation

Dongheui Lee
Human-centered Assistive Robotics, Technical University of Munich, Germany. E-mail: dhlee@tum.de

Raffaele Soloperto
Department of Electrical, Electronic, and Information Engineering, Università di Bologna, Italy. E-mail: soloperto.raffaele@gmail.com

Matteo Saveriano
Human-centered Assistive Robotics, Technical University of Munich, Germany. E-mail: matteo.saveriano@tum.de

1 Introduction

Future robotic applications will require a close collaboration between humans and robots. To ensure a smooth and effective cooperation, there is a need of representing human and robot actions in a compact and robust way that facilitates the understanding of performed motions as well as the generation of motions which flexibly adapt to different scenarios. Examples include recognizing human actions [44], gesture-based human-robot interaction [38, 4, 18], learning human activities [14], tasks learning by demonstration [1, 17, 33]. Aforementioned applications usually adopt Cartesian trajectories as task descriptors. Cartesian trajectories describe the spatial motion of one or multiple points attached to the human (robot) body. In real scenarios, Cartesian trajectories are affected by two kinds of motion variations, depending on *how* and *where* the same motion is executed [37], which limit the applicability of Cartesian task descriptors in motion recognition and reproduction.

Consider a case where a human takes a bottle from a table with the right hand and a marker attached to the hand is tracked by a camera. The motion is described by the Cartesian trajectory of the hand expressed in camera (reference) frame. There are motion variations depending on *how* the motion is executed:

- *Execution time and speed* - The human can reach the bottle faster or slower.
- *Amplitude* - The bottle can be far or close, hence the trajectory of the hand can be longer or shorter.

There are variations on *where* the motion takes place:

- *Starting pose* - The initial pose of the human can considerably change across different executions.

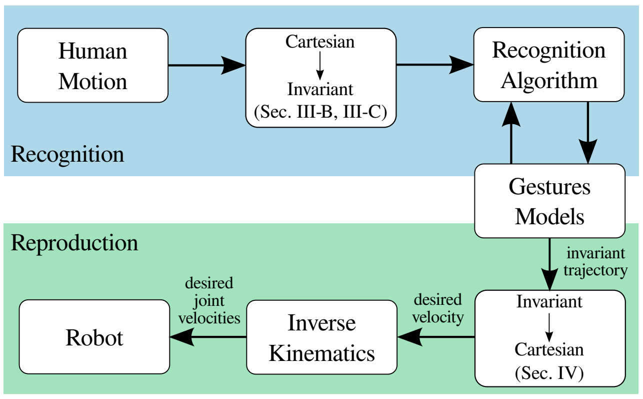


Fig. 1 System overview. In the recognition phase human motions are transformed in invariant trajectories to increase the recognition rate. During the reproduction, invariant trajectories are converted in Cartesian references and passed to the robot's controller.

- *Reference frame* - The camera can be accidentally or voluntarily moved.
- *Reference point* - The marker (a tracking target) can be attached slightly differently to the hand across the motion capturing sessions.

Described motion variations make the problem of gesture recognition hard to solve. Hence, to increase the recognition performance, it is desirable to have an invariant form of motion trajectories, i.e. a representation of motion which is coordinate-free and invariant to time and amplitude scaling. Invariant forms of motion trajectories help to focus on essential aspects of the motion and are beneficial also for a flexible generation of robot motions (see Sec. 8.2). The invariance to the starting pose and the amplitude of motion can be exploited to take the bottle from different starting locations, without providing further demonstrations.

We propose to overcome limitations of Cartesian trajectories by using a novel invariant representation of motion trajectories, namely the Denavit-Hartenberg inspired Bidirectional (DHB) invariant representation. DHB is a numerically robust representation of rigid body motions consisting of six values (three for the position and three for the orientation), which is a minimal representation of rigid body motions. The proposed representation is invariant to affine transformations (rotation, translation, linear and angular scaling), time scalings and viewpoints. The invariant representation is also bidirectional, i.e. it allows to transform the Cartesian space into the *invariant space* and vice-versa, without losing information. Invariance properties, numerical robustness and bi-directionality make DHB representation an useful task descriptor to recognize human actions robustly and to generate robot behaviors flexibly (see Fig. 1). In gesture recognition problems, DHB representation can be effectively used as a feature vector to increase the recognition rate of classification algorithms [2]. Indeed, the same motion, observed from different perspectives, executed in different positions or scaled in time and space, has always the same DHB representation. In motion generation problems, different *affinely transformed* instances of a Cartesian trajectory can be retrieved from the same DHB representation, which allows a flexible generation of robotic tasks in different situations. In this work, we present theoretical foundations and properties of DHB representation, together with an extensive evaluation on both synthetic and real datasets. Experimental results show the effectiveness of the proposed approach in recognizing human gestures and in reproducing and generalizing observed behaviours on real robots[1].

The rest of the paper is organized as follows. Section 2 presents related work. The DHB representation is proposed in Sec. 3. The reconstruction algorithm of Cartesian trajectory from its invariant representation is described in Sec. 4. Invariance properties of our representation are discussed in Sec. 5, singular cases are analyzed in Sec. 6. DHB invariants are theoretically compared with state-of-the-art approaches in Sec. 7. Section 8 presents experimental results and comparisons. Section 9 states the conclusions.

2 Related Work

2.1 Unidirectional invariant representations

Gesture recognition [44] is a classification problem in which motion trajectories have to be assigned to a certain action class. As discussed in Sec. 1, motion variations of Cartesian trajectories complicate the problem of gesture recognition. An effective solution to increase the recognition rate is to create a feature space with invariant properties, such as roto-translations and scale invariance.

In [3] human motions are modeled as temporal trajectories of estimated parameters (state) over time. The state controls stretching, scaling and translations of the gesture model with respect to the incoming data. The condensation algorithm [12] is used to incrementally

[1] This work is based on our preliminary results presented in [36]. Our previous work has been extended in several ways: *(i)* we provide more theoretical insights including a compact closed form of DHB invariants; *(ii)* we theoretically compare DHB with existing invariant representations, in order to underline differences and similarities; *(iii)* we compare the recognition performance of DHB invariants with several state-of-the-art approaches; *(iv)* we report several experiments to show that DHB invariants can be adopted as flexible motion descriptors to execute complex tasks.

match trajectory models to the multi-variate input data. The condensation algorithm does not make any parametric assumption about the state probability distribution and requires a large set of densities to estimate current parameters. To reduce the computational cost of the prediction, in [25] Hidden Markov Models (HMM) [26] are used to learn the prior distribution of both the observation covariance and the state transition probabilities.

Action descriptors invariant to affine transformations are directly computed from image coordinates in [23, 24, 47, 41]. The representation in [23, 24] is invariant under affine and projective transformations. Invariant values are computed by considering five fixed points on the rigid body and by tracking them in the image during the whole motion. Three dimensional invariants under affine transformations are proposed in [47, 41]. Their computation requires to track the same six points in all frames.

Alternative approaches focus on Euclidean group invariants. In [27, 28] gestures are represented by the spatio-temporal curvature of the Cartesian trajectory, that is invariant to roto-translations. This representation is useful to capture the dynamic instants of a trajectory, i.e. points where the velocity and/or the direction of motion change sign.

The invariant descriptor in [32] consists of two values, which describe respectively the linear and angular part of the motion, and is invariant to roto-translations and linear scaling. In this case, the scale invariance is not obtained by introducing an artificial scale depending on the motion length, but it holds for each frame. This approach is used in [20] to recognize human tasks by fusing gesture and object recognition into a dynamic Bayesian network [2].

The Fourier transform is leveraged in [39] to compute a view and scale invariant representation. Cartesian trajectories are partitioned into a temporal pyramid and the Fourier transform is computed for each segment at each level. Low-frequency Fourier coefficients are chosen as feature vectors.

In order to recognize full-body motions, some researchers focused on creating full-body motion descriptors. The 4D Action Feature Model (4D-AFM) is proposed in [46] to recognize full-body gestures observed from arbitrary views. Given a multi-view video sequence, a 3D shape for each frame is computed by projecting the multi-view 2D silhouettes into the 3D space. For each silhouette, a set of features (action sketch) describing changes in direction, speed and shape of the contour is computed. The 4D-AFM consists of a sequence of 3D shapes with actions sketches attached. Even if it is possible to recognize actions from a single view, multi-view video sequences are required to construct the model in the training process.

In [19] actions are modeled as a graph, where each node represents a salient posture. Salient postures are described by a set of 3D points belonging to the human body. In [45] a histogram of 3D joint locations (HOJ3D) is created by dividing the space into bins. The HOJ3D is then projected into a lower dimensional space using linear discriminant analysis [2] and clustered into k classes. HMMs are used for gesture recognition. Recognition performances of DHB and approaches in [19, 45] are compared in Sec. 8.1.3.

The availability of large computational power and big data has made deep neural networks popular recently. CNNs are effectively used in [40] to recognize human actions from input depth maps. In [9] the standard CNN is extended with four extra layers (slice, pool, roll and stack) which realize local invariance to rotations. Inspired by the recognition process in the visual cortex, LeCun [16] presents a hierarchical architecture to learn, from visual inputs, features invariant to affine transformations and illumination. Invariant features are learned in an unsupervised manner using sparse auto-encoders. Jaderberg et al [13] propose to learn warping transformations from input data and this extra learnable layer can be inserted into CNNs to improve the recognition performance. Deep learning approaches have shown high recognition accuracy in typical computer vision problems, like object recognition and scene parsing. Although they have good potential in action recognition problems, motion generation problems remains largely open. Moreover, deep learning approaches require long training time, big amount of training data, and high computational power. This limits their applicability on autonomous robotic systems with limited computational power. On the contrary, the proposed bidirectional DHB features can be extracted in a computationally efficient manner and their invariance properties are mathematically guaranteed for any Cartesian trajectory.

2.2 Bidirectional invariant representations

Besides improving the recognition rate, bidirectional invariant representations are flexible motion descriptors which allow the generation of different trajectory instances from the same descriptor (see Sec. 8.2). Unidirectional invariant representations in Sec. 2.1 are not suitable for motion reproduction, since the invariant trajectory cannot be transformed into a Cartesian one.

Frenet-Serret (FS) invariants [15] are the first example of a bidirectional invariant representation. FS representation uses the curvature, the torsion and their first-order derivatives to describe the motion of a spatial curve. In [42], a method is proposed to calculate FS invariants without using high-order derivatives. The resulting approximate FS representation is numerically robust, invariant to affine transformations and changes in the speed of execution. Original positions are retrieved by numerically integrating the Frenet-Serret equations [43].

FS invariants and their robust version in [42] neglect the orientation part of the motion, which is of importance for recognition and reproduction of complex motion. This limitation is overcome in [37], where Extended Frenet-Serret invariants (EFS) is proposed by considering also the orientation part of the motion. The EFS is not affected by affine transformations, time, linear and angular scale and motion profile. EFS invariants are compared with the bidirectional representation in [6], which is constructed by means of the Instantaneous Screw Axis (ISA). The comparison shows a superior recognition rate of the EFS invariants, due to the higher noise sensitivity of [6].

EFS invariants and the representation in [6] depend on high-order time derivatives of the velocity. The numerical computation of high-order derivatives on real non-smooth signals is sensitive to noise and round-off errors [5]. As experimentally shown in Sec. 8, the sensitivity of numerical derivatives makes difficult to estimate reliably the invariant values in [37, 6] and generates a non-negligible reconstruction error. On the contrary, the proposed DHB representation does not require high-order time derivatives. Moreover, the Cartesian trajectory is exactly reconstructed from its DHB representation. The resulting numerical robustness makes our invariants applicable in realistic human-robot interaction scenarios, where humans are monitored using noisy sensors, as RGB-D cameras.

3 DHB Invariant Representation of Motion

In this section we present the proposed DHB invariant representation of rigid body motions. Position-based invariants are firstly introduced. Then, velocity-based invariants are described and the relation between the two representations is detailed.

3.1 Theoretical Background

Rigid body motions are usually described in the Cartesian space considering the position and the orientation of a frame attached to the body (body frame) with respect to a reference (world) frame in each time instant t. The position $\mathbf{p}_t$ is uniquely defined by the 3D vector connecting the centers of the body and the reference frames. For the orientation, instead, several representations exist. As discussed in Appendix A, the minimum number of parameters needed to represent the orientation is three. This work uses a minimal representation of the orientation, the so-called *rotation vector* $\mathbf{r}_t$.

In our representation, two frames are attached to the rigid body (see Fig. 2). The first frame describes the position (linear velocity) in each time instant t. We refer to this frame as the *linear* frame. The second frame describes the orientation (angular velocity) in each t and it is referred as the *angular* frame. Moreover, to distinguish between the two frames, in the formulas the subscript p refers to the position and r refers to the orientation. Similarly, at velocity level v refers to the linear velocity and ω refers to the angular velocity. Once those frames are created, a minimal set of invariants (six values) are computed. The first two invariants m_p (m_v) and m_r (m_ω) represent respectively the norm of the position (linear velocity) and orientation vector (angular velocity) between the consecutive time instants t and $t+1$. Four more invariants θ_p^i (θ_v^i) and θ_r^i (θ_ω^i), $i = 1, 2$, are used to align the linear (angular) frame in t with the linear (angular) frame in $t+1$.

3.2 Position-based Invariants

3.2.1 Definition of linear and angular frames

The linear frame in Fig. 2(a) is created with the following steps:

- The x axis is the unit vector representing the normalized difference between the position vectors in two adjacent frames:

$$\hat{\mathbf{x}}_{p,t} = \frac{\mathbf{p}_{t+1} - \mathbf{p}_t}{\|\mathbf{p}_{t+1} - \mathbf{p}_t\|} = \frac{\Delta\mathbf{p}_t}{\|\Delta\mathbf{p}_t\|} \ . \tag{1}$$

  By construction $\Delta\mathbf{p}_t$ represents the linear velocity of the body in a unitary time.
- The y axis represents the direction of the common normal between the x axis at the current instant t and the x axis at the next instant $t+1$:

$$\hat{\mathbf{y}}_{p,t} = \frac{\hat{\mathbf{x}}_{p,t} \times \hat{\mathbf{x}}_{p,t+1}}{\|\hat{\mathbf{x}}_{p,t} \times \hat{\mathbf{x}}_{p,t+1}\|} \ . \tag{2}$$

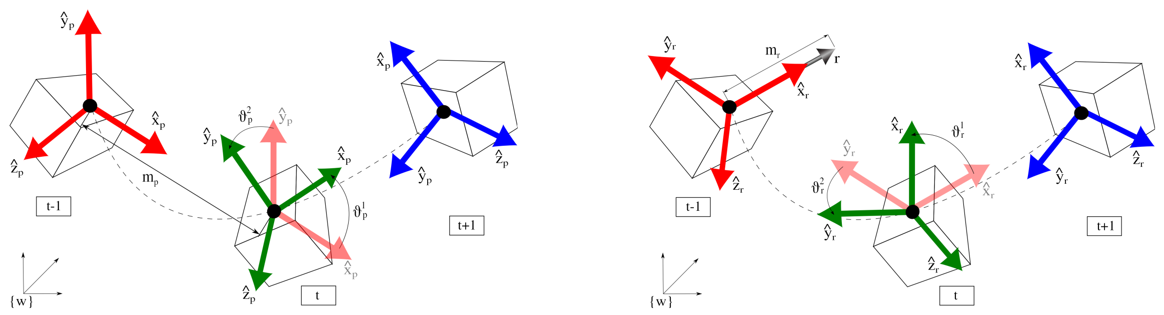
- The z axis is the cross product between the x axis and the y axis:

$$\hat{\mathbf{z}}_{p,t} = \hat{\mathbf{x}}_{p,t} \times \hat{\mathbf{y}}_{p,t} \ . \tag{3}$$

Following similar steps we define the angular frame in Fig. 2(b). The x axis is defined as:

$$\hat{\mathbf{x}}_{r,t} = \frac{\Delta\mathbf{r}_t}{\|\Delta\mathbf{r}_t\|} \ , \tag{4}$$

where the rotation vector $\Delta\mathbf{r}_t$ represents the relative rigid body orientation between consecutive time instants



(a) Linear frame (b) Angular frame

Fig. 2 (a) Linear and (b) angular frames in 3 consecutive time instants.

t and $t+1$. Hence, it represents the angular velocity needed to rotate the body from t to $t+1$ in a unitary time. The y axis and the z axis are then computed by:

$$\hat{\mathbf{y}}_{r,t} = \frac{\hat{\mathbf{x}}_{r,t} \times \hat{\mathbf{x}}_{r,t+1}}{\|\hat{\mathbf{x}}_{r,t} \times \hat{\mathbf{x}}_{r,t+1}\|} , \tag{5}$$

$$\hat{\mathbf{z}}_{r,t} = \hat{\mathbf{x}}_{r,t} \times \hat{\mathbf{y}}_{r,t} . \tag{6}$$

The direction of the axes of the linear (angular) frames is chosen to avoid discontinuities (jumps of $\pm\pi$) between subsequent time instants. Given two consecutive axes $\hat{\mathbf{a}}_t$ and $\hat{\mathbf{a}}_{t+1}$, we check the sign of the scalar product $s = \hat{\mathbf{a}}_t \cdot \hat{\mathbf{a}}_{t+1}$, setting $\hat{\mathbf{a}}_{t+1} = -\hat{\mathbf{a}}_{t+1}$ if $s < 0$.

3.2.2 Invariant values

The computation of the six invariant values, as well as the definition of the linear and angular frames, are inspired by the Denavit-Hartenberg (DH) notation [8]. The difference is that DH notation describes the pose of each body in a kinematic chain. Instead, our approach describes the motion of a rigid body in several instants. Moreover, we separate the position and the orientation by considering two frames, while a single frame is used in the DH notation.

As already mentioned, two invariants represent the norm of the relative positions and orientations between subsequent frames:

$$m_{p,t} = \|\Delta\mathbf{p}_t\| = \Delta\mathbf{p}_t \cdot \hat{\mathbf{x}}_{p,t} , \tag{7}$$

$$m_{r,t} = \|\Delta\mathbf{r}_t\| = \Delta\mathbf{r}_t \cdot \hat{\mathbf{x}}_{r,t} , \tag{8}$$

where $\hat{\mathbf{x}}_{p,t}$ and $\hat{\mathbf{x}}_{r,t}$ are the axes in (1) and (4) respectively. m_p and m_r in (7) and (8) describe the translation of the linear and angular frames. The rotation of linear and angular frames is described by four more values.

Let us consider the linear frame. The $\hat{\mathbf{y}}_{p,t}$ axis lies on the common normal between $\hat{\mathbf{x}}_{p,t}$ and $\hat{\mathbf{x}}_{p,t+1}$. According to the Denavit-Hartenberg notation [8], only the rotations about the x and y axes are needed to align the frames at t and $t+1$. As shown in Fig. 2(a), to align $\hat{\mathbf{x}}_{p,t}$ to $\hat{\mathbf{x}}_{p,t+1}$ the linear frame has to rotate an angle θ^1_p about $\hat{\mathbf{y}}_{p,t}$. The signed value of θ^1_p is computed as:

$$\begin{aligned} \theta^1_{p,t} &= \arctan\left(\frac{\hat{\mathbf{x}}_{p,t} \times \hat{\mathbf{x}}_{p,t+1}}{\hat{\mathbf{x}}_{p,t} \cdot \hat{\mathbf{x}}_{p,t+1}} \cdot \hat{\mathbf{y}}_{p,t}\right) \\ &= \arctan\left(\frac{\|\hat{\mathbf{x}}_{p,t}\|\|\hat{\mathbf{x}}_{p,t+1}\| \sin(\theta^1_{p,t})\hat{\mathbf{y}}_{p,t}}{\|\hat{\mathbf{x}}_{p,t}\|\|\hat{\mathbf{x}}_{p,t+1}\| \cos(\theta^1_{p,t})} \cdot \hat{\mathbf{y}}_{p,t}\right) \\ &= \arctan\left(\frac{\sin(\theta^1_{p,t})}{\cos(\theta^1_{p,t})}\right) , \end{aligned} \tag{9}$$

where the relationships $\mathbf{a}\cdot\mathbf{b} = \|\mathbf{a}\|\|\mathbf{b}\|\cos(\theta_{ab})$, $\mathbf{a}\times\mathbf{b} = \|\mathbf{a}\|\|\mathbf{b}\|\sin(\theta_{ab})\hat{\mathbf{n}}$ and (2) have been used. A further rotation θ^2_p about $\hat{\mathbf{x}}_{p,t+1}$ is required to align $\hat{\mathbf{y}}_{p,t}$ to $\hat{\mathbf{y}}_{p,t+1}$. The signed value of θ^2_p is computed as:

$$\theta^2_{p,t} = \arctan\left(\frac{\hat{\mathbf{y}}_{p,t} \times \hat{\mathbf{y}}_{p,t+1}}{\hat{\mathbf{y}}_{p,t} \cdot \hat{\mathbf{y}}_{p,t+1}} \cdot \hat{\mathbf{x}}_{p,t+1}\right) . \tag{10}$$

Following the same reasoning it is possible to align the angular frames between t and $t+1$ and to compute the last two invariants as:

$$\theta^1_{r,t} = \arctan\left(\frac{\hat{\mathbf{x}}_{r,t} \times \hat{\mathbf{x}}_{r,t+1}}{\hat{\mathbf{x}}_{r,t} \cdot \hat{\mathbf{x}}_{r,t+1}} \cdot \hat{\mathbf{y}}_{r,t}\right) , \tag{11}$$

$$\theta^2_{r,t} = \arctan\left(\frac{\hat{\mathbf{y}}_{r,t} \times \hat{\mathbf{y}}_{r,t+1}}{\hat{\mathbf{y}}_{r,t} \cdot \hat{\mathbf{y}}_{r,t+1}} \cdot \hat{\mathbf{x}}_{r,t+1}\right) . \tag{12}$$

Note that (9) and (11), as well as (10) and (12), are formally the same. For the completeness, full equations are given.

3.3 Velocity-based Invariants

In some cases the linear $\mathbf{v}$ and angular $\boldsymbol{\omega}$ velocities are directly available from the sensors. Instead of integrating the velocity to compute the position-based invariants, it is possible to calculate an invariant set directly from the velocity data. We refer to this set as velocity-based invariants. The approach used to calculate the velocity-based invariants is conceptually similar to the one used for the position-based invariants in Sec. 3.2. The main differences are illustrated as follows.

3.3.1 Definition of linear and angular frames

Also in this case two frames are used to describe the trajectory of the linear/angular velocity vectors in different time instants. The axes of the linear frame are computed as:

$$\hat{\mathbf{x}}_{v,t} = \frac{\mathbf{v}_t}{\|\mathbf{v}_t\|} \ , \tag{13}$$

$$\hat{\mathbf{y}}_{v,t} = \frac{\hat{\mathbf{x}}_{v,t} \times \hat{\mathbf{x}}_{v,t+1}}{\|\hat{\mathbf{x}}_{v,t} \times \hat{\mathbf{x}}_{v,t+1}\|} \ , \tag{14}$$

$$\hat{\mathbf{z}}_{v,t} = \hat{\mathbf{x}}_{v,t} \times \hat{\mathbf{y}}_{v,t} \ . \tag{15}$$

The axes of the angular frame are computed as:

$$\hat{\mathbf{x}}_{\omega,t} = \frac{\boldsymbol{\omega}_t}{\|\boldsymbol{\omega}_t\|} \ , \tag{16}$$

$$\hat{\mathbf{y}}_{\omega,t} = \frac{\hat{\mathbf{x}}_{\omega,t} \times \hat{\mathbf{x}}_{\omega,t+1}}{\|\hat{\mathbf{x}}_{\omega,t} \times \hat{\mathbf{x}}_{\omega,t+1}\|} \ , \tag{17}$$

$$\hat{\mathbf{z}}_{\omega,t} = \hat{\mathbf{x}}_{\omega,t} \times \hat{\mathbf{y}}_{\omega,t} \ . \tag{18}$$

Note the strong similarities between (1)-(3) and (13)-(15), as well as between (4)-(6) and (16)-(18).

3.3.2 Invariant values

Once the frames are defined, velocity-based invariants can be computed. The first two invariants represent the norm of the linear and angular velocities:

$$m_{v,t} = \|\mathbf{v}_t\| = \mathbf{v}_t \cdot \hat{\mathbf{x}}_{v,t} \ , \tag{19}$$

$$m_{\omega,t} = \|\boldsymbol{\omega}_t\| = \boldsymbol{\omega}_t \cdot \hat{\mathbf{x}}_{\omega,t} \ . \tag{20}$$

Following the same reasoning in Sec. 3.2.2, four other values are used to align the linear and angular frames in two consecutive time instants. The resulting invariants for the linear velocity are:

$$\theta^1_{v,t} = \arctan\left(\frac{\hat{\mathbf{x}}_{v,t} \times \hat{\mathbf{x}}_{v,t+1}}{\hat{\mathbf{x}}_{v,t} \cdot \hat{\mathbf{x}}_{v,t+1}} \cdot \hat{\mathbf{y}}_{v,t}\right) \ , \tag{21}$$

$$\theta^2_{v,t} = \arctan\left(\frac{\hat{\mathbf{y}}_{v,t} \times \hat{\mathbf{y}}_{v,t+1}}{\hat{\mathbf{y}}_{v,t} \cdot \hat{\mathbf{y}}_{v,t+1}} \cdot \hat{\mathbf{x}}_{v,t+1}\right) \ . \tag{22}$$

The invariants for the angular velocity are:

$$\theta^1_{\omega,t} = \arctan\left(\frac{\hat{\mathbf{x}}_{\omega,t} \times \hat{\mathbf{x}}_{\omega,t+1}}{\hat{\mathbf{x}}_{\omega,t} \cdot \hat{\mathbf{x}}_{\omega,t+1}} \cdot \hat{\mathbf{y}}_{\omega,t}\right) \ , \tag{23}$$

$$\theta^2_{v,t} = \arctan\left(\frac{\hat{\mathbf{y}}_{\omega,t} \times \hat{\mathbf{y}}_{\omega,t+1}}{\hat{\mathbf{y}}_{\omega,t} \cdot \hat{\mathbf{y}}_{\omega,t+1}} \cdot \hat{\mathbf{x}}_{\omega,t+1}\right) \ . \tag{24}$$

Note that $\theta^1_{v,t}$ and $\theta^2_{v,t}$ are calculated by substituting $\hat{\mathbf{x}}_{p,t}$ with $\hat{\mathbf{x}}_{v,t}$ and $\hat{\mathbf{y}}_{p,t}$ with $\hat{\mathbf{y}}_{v,t}$ in (9) and (10). In the same manner, $\theta^1_{\omega,t}$ and $\theta^2_{\omega,t}$ are calculated by substituting $\hat{\mathbf{x}}_{r,t}$ with $\hat{\mathbf{x}}_{\omega,t}$ and $\hat{\mathbf{y}}_{r,t}$ with $\hat{\mathbf{y}}_{\omega,t}$ in (11) and (12).

3.4 Relation between position- and velocity-based DHB Representations

Recalling that $\Delta\mathbf{p}$ and $\Delta\mathbf{r}$ are the linear and angular velocities between two consecutive frames in a unitary time, a simple relation holds between position and velocity-based invariants in the discrete time domain. Indeed, from (7) and (8), we have:

$$m_{v,t} = \|\mathbf{v}_t\| = \frac{\|\Delta\mathbf{p}_t\|}{\Delta t} = \frac{m_{p,t}}{\Delta t} \ , \tag{25}$$

$$m_{\omega,t} = \|\boldsymbol{\omega}_t\| = \frac{\|\Delta\mathbf{r}_t\|}{\Delta t} = \frac{m_{r,t}}{\Delta t} \ , \tag{26}$$

where Δt is the sample time. Moreover, from (1) and (13) it is easy to verify that $\hat{\mathbf{x}}_{v,t} = \hat{\mathbf{x}}_{p,t}$, while from (4) and (16) it comes that $\hat{\mathbf{x}}_{\omega,t} = \hat{\mathbf{x}}_{r,t}$. Finally, comparing (21)-(24) with (9)-(12) and considering that the y axis definitions are formally the same for all the invariants, it is possible to prove that $[\theta^1_v,\ \theta^2_v,\ \theta^1_\omega,\ \theta^2_\omega] = [\theta^1_p,\ \theta^2_p,\ \theta^1_r,\ \theta^2_r]$.

3.5 DHB Representation in Closed Form

We presented DHB invariants considering the definition of linear and angular frames and their spatial motion. This formulation is useful to understand the physical meaning of each invariant and the reconstruction procedure in Sec. 4. To simplify the formulation, we provide a closed form of DHB invariants, i.e. a set of equations that map positions (velocities) to the invariant space.

Recalling that $\hat{\mathbf{x}}_{v,t} = \frac{\mathbf{v}_t}{\|\mathbf{v}_t\|}$, the expression of $\hat{\mathbf{y}}_{v,t}$ in (14) can be rewritten as:

$$\hat{\mathbf{y}}_{v,t} = \frac{\mathbf{v}_t \times \mathbf{v}_{t+1}}{\|\mathbf{v}_t\|\|\mathbf{v}_{t+1}\|}\frac{\|\mathbf{v}_t\|\|\mathbf{v}_{t+1}\|}{\|\mathbf{v}_t \times \mathbf{v}_{t+1}\|} = \frac{\mathbf{v}_t \times \mathbf{v}_{t+1}}{\|\mathbf{v}_t \times \mathbf{v}_{t+1}\|} \ . \tag{27}$$

Considering (21) and (27), $\theta^1_{v,t}$ can be rewritten as:

$$\begin{aligned}\theta^1_{v,t} &= \arctan\left(\frac{\mathbf{v}_t \times \mathbf{v}_{t+1}}{\|\mathbf{v}_t\|\|\mathbf{v}_{t+1}\|}\frac{\|\mathbf{v}_t\|\|\mathbf{v}_{t+1}\|}{\mathbf{v}_t \cdot \mathbf{v}_{t+1}} \cdot \hat{\mathbf{y}}_{v,t}\right) \\ &= \arctan\left(\frac{\mathbf{v}_t \times \mathbf{v}_{t+1}}{\mathbf{v}_t \cdot \mathbf{v}_{t+1}} \cdot \frac{\mathbf{v}_t \times \mathbf{v}_{t+1}}{\|\mathbf{v}_t \times \mathbf{v}_{t+1}\|}\right) \\ &= \arctan\left(\frac{\|\mathbf{v}_t \times \mathbf{v}_{t+1}\|}{\mathbf{v}_t \cdot \mathbf{v}_{t+1}}\right) \ .\end{aligned} \tag{28}$$

Following similar steps, one can see that closed forms for $\theta^1_{p,t}$, $\theta^1_{r,t}$, $\theta^1_{\omega,t}$ are formally the same as (28).

To express $\theta^2_{v,t}$ in a closed form, let us first focus on the division in (22). Considering (27), it is straightforward to derive that:

$$\begin{aligned}\frac{\hat{\mathbf{y}}_{v,t} \times \hat{\mathbf{y}}_{v,t+1}}{\hat{\mathbf{y}}_{v,t} \cdot \hat{\mathbf{y}}_{v,t+1}} &= \frac{(\mathbf{v}_t \times \mathbf{v}_{t+1}) \times (\mathbf{v}_{t+1} \times \mathbf{v}_{t+2})}{(\mathbf{v}_t \times \mathbf{v}_{t+1}) \cdot (\mathbf{v}_{t+1} \times \mathbf{v}_{t+2})} \\ &= \frac{-(\mathbf{v}_{t+1} \times \mathbf{v}_t) \times (\mathbf{v}_{t+1} \times \mathbf{v}_{t+2})}{-(\mathbf{v}_{t+1} \times \mathbf{v}_t) \cdot (\mathbf{v}_{t+1} \times \mathbf{v}_{t+2})} \\ &= \frac{[\mathbf{v}_{t+1} \cdot (\mathbf{v}_t \times \mathbf{v}_{t+2})]\,\mathbf{v}_{t+1}}{(\mathbf{v}_{t+1} \times \mathbf{v}_t) \cdot (\mathbf{v}_{t+1} \times \mathbf{v}_{t+2})} \ ,\end{aligned} \tag{29}$$

where the relationships $\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a}$ and $(\mathbf{a} \times \mathbf{b}) \times (\mathbf{a} \times \mathbf{c}) = \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})\mathbf{a}$ have been used. By multiplying (29) by $\hat{\mathbf{x}}_{v,t+1}$ and taking the arc tangent, we obtain the closed form expression:

$$\theta^2_{v,t} = \arctan\left(\frac{\|\mathbf{v}_{t+1}\|\mathbf{v}_{t+1} \cdot (\mathbf{v}_t \times \mathbf{v}_{t+2})}{(\mathbf{v}_{t+1} \times \mathbf{v}_t) \cdot (\mathbf{v}_{t+1} \times \mathbf{v}_{t+2})}\right) \ . \tag{30}$$

Algorithm 1 DHB Invariant Representation

```
Position-based invariants
Input: a set of relative positions $\{\Delta\mathbf{p}\}_t$ and rotation vectors $\{\Delta\mathbf{r}\}_t$, $t = 1, \dots, N$
return cartesianToDHB($\{\Delta\mathbf{p}\}_t, \{\Delta\mathbf{r}\}_t$)

Velocity-based invariants
Input: a set of linear $\{\mathbf{v}\}_t$ and angular $\{\boldsymbol{\omega}\}_t$, $t = 1, \dots, N$ velocities
return cartesianToDHB($\{\mathbf{v}\}_t, \{\boldsymbol{\omega}\}_t$)

cartesianToDHB($\{\boldsymbol{l}\}_t, \{\mathbf{a}\}_t$)
 1. $[\{m_l\}_t, \{\theta_l^1\}_t, \{\theta_l^2\}_t]$ = computeInvariants($\{\boldsymbol{l}\}_t$)
 2. $[\{m_a\}_t, \{\theta_a^1\}_t, \{\theta_a^2\}_t]$ = computeInvariants($\{\mathbf{a}\}_t$)
return $\{m_l\}_t, \{\theta_l^1\}_t, \{\theta_l^2\}_t, \{m_a\}_t, \{\theta_a^1\}_t, \{\theta_a^2\}_t$

computeInvariants($\{\mathbf{u}\}_t$):
 3. for each $t \in [1, N-2]$ do
 4.     $m_{u,t} = \|\mathbf{u}_t\|$
 5.     $\theta_{u,t}^1 = \arctan\left(\frac{\|\mathbf{u}_{u,t} \times \mathbf{u}_{u,t+1}\|}{\mathbf{u}_{u,t} \cdot \mathbf{u}_{u,t+1}}\right)$
 6.     $\theta_{u,t}^2 = \arctan\left(\frac{\|\mathbf{u}_{u,t+1}\| \mathbf{u}_{u,t+1} \cdot (\mathbf{u}_{u,t} \times \mathbf{u}_{u,t+2})}{(\mathbf{u}_{u,t+1} \times \mathbf{u}_{u,t}) \cdot (\mathbf{u}_{u,t+1} \times \mathbf{u}_{u,t+2})}\right)$
 7. end for
return $\{m_u\}_t, \{\theta_u^1\}_t, \{\theta_u^2\}_t$
```

Following similar steps, it is easy to show that closed forms for $\theta^2_{p,t}$, $\theta^2_{r,t}$, $\theta^2_{\omega,t}$ are formally the same as (30).

The procedure to calculate position and velocity based invariants is summarized in Algorithm 1. The procedure is analogous for both the position and velocity based invariants. The function *computeInvariants* takes as input a set of vectors (positions, rotation vectors, linear or angular velocities) and returns the related set of three invariants. The function *cartesianToDHB* just calls twice *computeInvariants* to compute the complete set of invariant values (six for each time instant). Note that, in Algorithm 1, $\mathbf{u}$ refers to a generic vector, l stands for *linear* and a stands for *angular*.

4 Trajectory Reconstruction

4.1 Pose Reconstruction

The reconstruction of the pose (position and orientation) of the rigid body in a reference (world) frame requires three steps. First, the pose of the linear and angular frames in each instant t is computed as:

$$\mathbf{H}_{p,t} = \begin{bmatrix} \mathbf{R}_y(\theta_p^1)\mathbf{R}_x(\theta_p^2) & \mathbf{m}_p \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{31}$$

$$\mathbf{H}_{r,t} = \begin{bmatrix} \mathbf{R}_y(\theta_r^1)\mathbf{R}_x(\theta_r^2) & \mathbf{m}_r \\ \mathbf{0}^T & 0 \end{bmatrix}, \tag{32}$$

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 [35], and $\mathbf{0}^T = [0 \ 0 \ 0]$. Note that the matrices in (31) and (32) are formally the same except for the fourth value of the fourth column. The reason will be clarified later.

Secondly, with the known pose of the linear $\mathbf{H}_{p,1}$ and angular $\mathbf{H}_{r,1}$ frames with respect to the world frame in the first time instant, it is possible to calculate:

$$\mathbf{H}^w_{p,t} = \mathbf{H}_{p,1} \cdot \mathbf{H}_{p,2} \cdot \dots \cdot \mathbf{H}_{p,t}, \tag{33}$$

$$\mathbf{H}^w_{r,t} = \mathbf{H}_{r,1} \cdot \mathbf{H}_{r,2} \cdot \dots \cdot \mathbf{H}_{r,t}, \tag{34}$$

where $\mathbf{H}^w_{p,t}$ and $\mathbf{H}^w_{r,t}$ represent the pose of the linear and angular frames with respect to the reference frame in a time instant t.

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

$$\mathbf{p}_t = \mathbf{R}^w_{p,1}\Delta\mathbf{p}_1 + \dots + \mathbf{R}^w_{p,t}\Delta\mathbf{p}_t = \mathbf{H}^w_{p,t}[1:3,4], \tag{35}$$

where $\mathbf{R}^w_{p,i}$ is the rotation of the linear frame with respect to the world frame at $t = i$ and $\mathbf{H}^w_{p,t}[1:3,4]$ are the first three elements of the fourth column of $\mathbf{H}^w_{p,t}$ in (33). The same approach does not directly apply to reconstruct the absolute orientation of the rigid body, because $\Delta\mathbf{r}_t + \Delta\mathbf{r}_{t+1} \neq \mathbf{r}_{t+1}$. Having chosen $\mathbf{H}_{r,t}[4,4] = 0$ in (32), we can extract the relative rotation vector

$$\Delta\mathbf{r}_t = \mathbf{H}^w_{r,t}[1:3,4] \tag{36}$$

from (34). The orientation of the rigid body with respect to the world frame in a time instant t is computed as $\mathbf{R}^w_t = \exp(\Delta\mathbf{r}_1) \cdot \dots \cdot \exp(\Delta\mathbf{r}_t)$, where $\exp(\mathbf{r})$ transforms a rotation vector into a rotation matrix (see Appendix A).

4.2 Velocity Reconstruction

The reconstruction of the velocity of the rigid body is similar to the reconstruction of the position. Firstly, the pose of the linear and angular frames in each time instant is computed as:

$$\mathbf{H}_{v,t} = \begin{bmatrix} \mathbf{R}_y(\theta_v^1)\mathbf{R}_x(\theta_v^2) & \mathbf{m}_v \\ \mathbf{0}^T & 0 \end{bmatrix}, \tag{37}$$

$$\mathbf{H}_{\omega,t} = \begin{bmatrix} \mathbf{R}_y(\theta_\omega^1)\mathbf{R}_x(\theta_\omega^2) & \mathbf{m}_\omega \\ \mathbf{0}^T & 0 \end{bmatrix}, \tag{38}$$

where $\mathbf{m}_v = [m_v \ 0 \ 0]^T$, $\mathbf{m}_\omega = [m_\omega \ 0 \ 0]^T$. Secondly, knowing the pose of the linear $\mathbf{H}_{v,1}$ and angular $\mathbf{H}_{\omega,1}$ frames with respect to the world frame in the first time instant, it is possible to calculate:

$$\mathbf{H}^w_{v,t} = \mathbf{H}_{v,1} \cdot \dots \cdot \mathbf{H}_{v,t}, \quad \mathbf{H}^w_{\omega,t} = \mathbf{H}_{\omega,1} \cdot \dots \cdot \mathbf{H}_{\omega,t}. \tag{39}$$

Finally, the velocity in a time instant t is computed as $\mathbf{v}_t = \mathbf{H}^w_{v,t}[1:3,4]$ and $\boldsymbol{\omega}_t = \mathbf{H}^w_{\omega,t}[1:3,4]$.

4.3 Variations of Trajectory Reconstruction

The proposed reconstruction algorithm can be extended to generate affine transformed instances of a Cartesian trajectory from the same invariant descriptor. For simplicity, let us focus on the reconstruction of the Cartesian position. In order to generate shorter or longer trajectories, the vector $\mathbf{m}_p$ in (31) can be defined with a scaling factor, i.e. $\mathbf{m}_p^s = [s_p m_p \ 0 \ 0]^T$. In order to generate roto-translated instances of the Cartesian trajectory, the initial pose $\mathbf{H}_{p,1}$ in (33) can be chosen accordingly. Note that a similar approach applies for the reconstruction of the orientation, as well as for the reconstruction of linear and angular velocities.

The flexibility of the reconstruction procedure is useful to reproduce various trajectory instances from a single invariant trajectory as well as to reproduce human demonstrations on different robotics systems. Cartesian trajectories, in fact, depend on the absolute pose of the rigid body in the chosen reference frame. To reproduce demonstrations on a robot, one has to express Cartesian data in the robot reference frame. Even when the reference frame is not given in a public database, DHB invariants can reconstruct the Cartesian trajectory directly by using the current robot reference frame as the initial frame.

5 Invariance Properties

Invariance properties of DHB representations are illustrated in this section. The same properties are valid for both position and velocity-based invariants. Hence, we use $\mathbf{u}$ to refer to a generic vector, the subscript l to refer to linear invariants and a to refer to angular invariants.

Reference frame - Cartesian trajectories are affected by the choice of the reference frame. The reference frame can be, for example, the frame attached to a camera observing the scene (viewpoint). It is interesting to show that DHB is a coordinate-free representation of rigid body motions. Translations of the reference frame (viewpoint) do not affect the relative position between two frames, the orientation of the rigid body, as well as its linear and angular velocities. To prove the invariance to rotations, let us assume the vector $\mathbf{u}$ is rotated by applying an arbitrary rotation matrix $\mathbf{R}$. Recalling that rotations do not change the norm of a vector we have:

$$m_{Rn} = \|\mathbf{R}\mathbf{u}_n\| = \|\mathbf{R}\|\,\|\mathbf{u}_n\| = \|\mathbf{u}_n\| = m_n, \ n = l, a \qquad (40)$$

where the property $\|\mathbf{R}\| = 1$ is used. The vector $\mathbf{u}_n$ in (40) represents one of $\Delta\mathbf{p}$, $\mathbf{v}$, $\Delta\mathbf{r}$ or $\boldsymbol{\omega}$. The other four invariant values $\theta_n^i, \ n = l, a, \ i = 1, 2$ represent the angle between two vectors. It is easy to verify that:

$$\begin{aligned}\theta_{Rl}^1 &= \arctan\left(\frac{\mathbf{R}\hat{\mathbf{x}}_{l,t} \times \mathbf{R}\hat{\mathbf{x}}_{l,t+1}}{\mathbf{R}\hat{\mathbf{x}}_{l,t} \cdot \mathbf{R}\hat{\mathbf{x}}_{l,t+1}} \cdot \mathbf{R}\hat{\mathbf{x}}_{l,t}\right) \\ &= \arctan\left(\frac{(\hat{\mathbf{x}}_{l,t} \times \hat{\mathbf{x}}_{l,t+1})^T}{\hat{\mathbf{x}}_{l,t}^T \mathbf{R}^T \cdot \mathbf{R}\hat{\mathbf{x}}_{l,t+1}} \mathbf{R}^T \cdot \mathbf{R}\hat{\mathbf{y}}_{l,t}\right) = \theta_l^1 \ , \end{aligned} \qquad (41)$$

Following the reasoning in (41) one can show the invariance of θ_l^2, θ_a^1 and θ_a^2.

Time, linear and angular scale - Motions executed at different speeds have different time scales. Hence, time scale invariance is of importance to compare motions performed at different velocities. As suggested in [6], a dimensionless time is defined as $t' = t/t_f$, where t_f is the total duration of the motion. Invariants are made independent on the time scale by multiplying each m_n^i and θ_n^i by t_f and by substituting t with t'.

In order to recognize motions performed by different users it is of importance to guarantee the invariance to scaling factors [32]. It is known that the scaling of two vectors does not affect the angle between them. Since the four θ_n^i represent angles between unit vectors, they are independent on linear and angular scales. The two $m_n^i, \ n = l, a, \ i = 1, 2$ values are made invariant to scaling factors by:

$$m'_{l,t} = \frac{m_{l,t}}{\int_{t=0}^{t_f} |m_{l,t}|}, \quad m'_{a,t} = \frac{m_{a,t}}{\int_{t=0}^{t_f} |m_{a,t}|} \ , \qquad (42)$$

where $\int_{t=0}^{t_f} |m_{l,t}|$ and $\int_{t=0}^{t_f} |m_{a,t}|$ are respectively the linear and angular scale of motion and t_f is the total duration of the motion[2].

Speed invariance - The invariance to the motion profile, or the speed of execution, can be obtained by expressing DHB invariants as a function of a degree of advancement, as described in [37]. Speed invariance can be achieved in theory. In practice, however, the discrete sampling time of real sensors strongly affects motions executed at different speeds and the resulting invariants. This is common for all invariant representations.

Reverse motion - It can be useful, especially for recognition purposes, to have the same representation for motions executed in a direction or in the opposite direction. Invariance to the direction of motion for m_l and m_a is achieved by considering them in the reverse order, i.e. $m_{l,t}^{rev} = m_{l,(t_f - t)}$ and $m_{a,t}^{rev} = m_{a,(t_f - t)}$, where t_f is the total duration of the motion. The four θ_n^i invariants have to be considered in the reverse order and they have to be shifted by 1 or 2 samples, i.e. $\theta_{n,t}^{1,rev} = \theta_{n,(t_f - t + 1)}^1$, $\theta_{n,t}^{2,rev} = \theta_{n,(t_f - t + 2)}^2$ for $n = l, a$.

[2] In the discrete time case, the integral $\int_{t=0}^{t_f} |\bullet|$ in (42) is replaced by $\sum_{t=0}^{t_f} |\bullet|$.

Reference point - Analogously to Cartesian trajectories, $m_{l,t}$, $\theta^1_{l,t}$ and $\theta^2_{l,t}$ depend on the reference point used to describe the translation. Hence, for different reference points, the above invariants will be different for the same motion. To overcome this limitation, one has to ensure that the reference point is kept the same (or slightly varying) during the entire motion[3]. Note that the invariance to the reference point is achieved in [6], while EFS invariants also depend on the reference point.

Table 2 EFS and DS Representations

EFS invariants [37]
$e^1_\omega = \pm\lVert\boldsymbol{\omega}\rVert$
$e^2_\omega = \pm\frac{\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert}{\lVert\boldsymbol{\omega}\rVert^2}$
$e^3_\omega = \pm\frac{\lVert(\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}})\times(\boldsymbol{\omega}\times\ddot{\boldsymbol{\omega}})\rVert}{\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert^2}$
$e^1_v = \pm\lVert\mathbf{v}\rVert$
$e^2_v = \pm\frac{\lVert\mathbf{v}\times\dot{\mathbf{v}}\rVert}{\lVert\mathbf{v}\rVert^2}$
$e^3_v = \pm\frac{\lVert(\mathbf{v}\times\dot{\mathbf{v}})\times(\mathbf{v}\times\ddot{\mathbf{v}})\rVert}{\lVert\mathbf{v}\times\dot{\mathbf{v}}\rVert^2}$
DS invariants [6]
$d^1_\omega = \pm\lVert\boldsymbol{\omega}\rVert$
$d^2_\omega = \pm\frac{\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert}{\lVert\boldsymbol{\omega}\rVert^2}$
$d^3_\omega = \pm\frac{\lVert\boldsymbol{\omega}\rVert}{\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert^2}\lvert(\omega\times\dot{\omega})\cdot\ddot{\omega}\rvert$
$d^1_v = \pm\frac{\mathbf{v}\cdot\boldsymbol{\omega}}{\lVert\boldsymbol{\omega}\rVert}$
$d^2_v = \pm\frac{\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}}{\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert}\cdot\left[\frac{\lVert\boldsymbol{\omega}\rVert^2(\dot{\boldsymbol{\omega}}\times\mathbf{v}+\boldsymbol{\omega}\times\dot{\mathbf{v}})-2(\boldsymbol{\omega}\times\mathbf{v})\cdot(\boldsymbol{\omega}\cdot\dot{\boldsymbol{\omega}})}{\lVert\boldsymbol{\omega}\rVert^4}\right]$
$d^3_v = \mp\left\{\frac{[\dot{\boldsymbol{\omega}}\times(\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}})+\boldsymbol{\omega}\times(\boldsymbol{\omega}\times\ddot{\boldsymbol{\omega}})]}{\lVert\boldsymbol{\omega}\rVert^3\cdot\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert^2}\cdot\frac{\lVert\boldsymbol{\omega}\rVert^2(\dot{\boldsymbol{\omega}}\times\mathbf{v}+\boldsymbol{\omega}\times\dot{\mathbf{v}})-2\boldsymbol{\omega}\cdot\dot{\boldsymbol{\omega}}\cdot(\boldsymbol{\omega}\times\mathbf{v})}{\lVert\boldsymbol{\omega}\rVert^3\cdot\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert^2}\right\}\mp\left\{[\omega\times(\omega\times\dot{\omega})]\cdot\left[\frac{\lVert\boldsymbol{\omega}\rVert^2(\ddot{\boldsymbol{\omega}}\times\mathbf{v}+2\dot{\boldsymbol{\omega}}\times\dot{\mathbf{v}}+\boldsymbol{\omega}\times\ddot{\mathbf{v}})}{\lVert\boldsymbol{\omega}\rVert^3\cdot\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert^2}-\frac{2(\lVert\dot{\boldsymbol{\omega}}\rVert^2+\boldsymbol{\omega}\cdot\dot{\boldsymbol{\omega}})(\boldsymbol{\omega}\times\mathbf{v})}{\lVert\boldsymbol{\omega}\rVert^3\cdot\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert^2}\right]\right\}\pm\left\{\left[\frac{3(\boldsymbol{\omega}\cdot\dot{\boldsymbol{\omega}})}{2\lVert\boldsymbol{\omega}\rVert^2}+\frac{(\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}})\cdot(\boldsymbol{\omega}\times\ddot{\boldsymbol{\omega}})}{\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert^2}\right]\cdot[\omega\times(\omega\times\dot{\omega})]\cdot\left[\frac{\lVert\boldsymbol{\omega}\rVert^2(\dot{\boldsymbol{\omega}}\times\mathbf{v}-\boldsymbol{\omega}\times\dot{\mathbf{v}})-2(\boldsymbol{\omega}\cdot\dot{\boldsymbol{\omega}}(\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}})}{\lVert\boldsymbol{\omega}\rVert^3\cdot\lVert\boldsymbol{\omega}\times\dot{\boldsymbol{\omega}}\rVert^2}\right]\right\}$

6 Special Motions

In some cases one of the axes of the linear (angular) frame cannot be uniquely defined. We refer to these situations as singular cases or singularities. Considering the definition of linear and angular frames in Sec. 3, it is clear that singularities occur when the axes cannot be normalized because the norm (denominator) drops to zero. Singular cases are analyzed and effective solutions are proposed accordingly. The invariant representation of the singular cases is summarized in Tab. 1.

Table 1 Invariant representation of special motions.

Pure translation	m_l	θ^1_l	θ^2_l	0	0	0
Pure rotation	0	0	0	m_a	θ^1_a	θ^2_a
Translation str. line	m_l	0	0	m_a	θ^1_a	θ^2_a
Rotation par. axes	m_l	θ^1_l	θ^2_l	m_a	0	0
Planar motion	m_l	θ^1_l	0	m_a	0	0

Pure translations - In the case of pure translations, the x axis of the angular frame becomes the null vector from eq. (4) or eq. (16). When there is no angular motion between two consecutive instants, the angular frame is kept unchanged from the previous angular frame: $\hat{\mathbf{x}}_{a,t} = \hat{\mathbf{x}}_{a,t-1}$, $\hat{\mathbf{y}}_{a,t} = \hat{\mathbf{y}}_{a,t-1}$. Since the angular frame in the first instant can be easily known (see Sec. 4), we can always find the angular frame. The resulting invariants for the angular part become $\{m_a\}_t = \{\theta^1_a\}_t = \{\theta^2_a\}_t = \{0\}_t$.

Pure rotations - In the case of pure rotations, the x axis of the linear frame becomes the null vector from eq. (1) or eq. (13). Thus the linear frame cannot be defined uniquely. When there is no translational motion between two consecutive instants, the linear frame is kept unchanged from the previous linear frame: $\hat{\mathbf{x}}_{l,t} = \hat{\mathbf{x}}_{l,t-1}$, $\hat{\mathbf{y}}_{l,t} = \hat{\mathbf{y}}_{l,t-1}$. Since the linear frame in the first instant is known, we can always find the linear frame. The resulting invariants for the linear part are $\{m_l\}_t = \{\theta^1_l\}_t = \{\theta^2_l\}_t = \{0\}_t$.

Translations along a straight line - In this case, the x axes of the linear frame are always aligned and the y axes becomes singular from eq. (2) or eq. (14). For translations along a straight line between two consecutive instants t and $t+1$, we set $\hat{\mathbf{y}}_{l,t} = \hat{\mathbf{y}}_{l,t-1}$ and obtain $\theta^1_{l,t} = \theta^2_{l,t} = 0$.

Rotations about parallel axes - In the case of rotations about parallel axes, the x axes of the angular frame are always parallel and the y axis becomes singular. For rotations about parallel axes in two consecutive instants t and $t+1$, we set $\hat{\mathbf{y}}_{a,t} = \hat{\mathbf{y}}_{a,t-1}$ and obtain $\theta^1_{a,t} = \theta^2_{a,t} = 0$.

Planar motions - In the case of planar motions, the rigid body translates on a plane π and rotates about an axis orthogonal to π. By construction, the y axes of the linear frame are orthogonal to π, and of course parallel to each other. As a consequence we obtain $\{\theta^2_l\}_t = \{0\}_t$. The angular motion is a rotation about parallel axes (orthogonal to π), resulting in $\{\theta^1_a\}_t = \{\theta^2_a\}_t = \{0\}_t$.

7 Comparison with Existing Representations

This section describes theoretical relationships and differences between DHB and two state-of-the-art bidi-

[3] As shown in Sec. 8.1.1 and Sec. 8.1.3, the proposed DHB descriptor works reasonably well with kinect sensors, which does not ensure tracking of the perfectly same point of a body part.

rectional invariant representations: EFS [37] and DS[4] [6]. To the best of our knowledge, EFS and DS are the only bidirectional representations which consider also the orientation part of the motion. EFS and DS representations transform velocities and their time derivatives into invariants, as shown in Tab. 2[5]. They are compared with velocity-based DHB invariants, since position and velocity-based DHB invariants are practically the same and the same properties hold for both the representations, as stated in Sec. 3.4. Several properties are compared and following analysis are summarized in Tab. 3.

7.1 DHB and DS representations

DS [6] is a bidirectional invariant representation of rigid body motions, constructed by means of the Instantaneous Screw Axis (ISA) [21]. Two invariants represent the translational velocity along the ISA and the rotational velocity about the ISA. Four other invariant values describe the motion of the ISA between two consecutive time instants. Given the twist, i.e. linear $\mathbf{v}_t$ and angular $\boldsymbol{\omega}_t$ velocities, in each time instant, the six invariants are computed as shown in Tab. 2, where the sign of each invariant is chosen in a way that it avoids discontinuities between consecutive time instants.

In the continuous time domain ($\Delta t \longrightarrow 0$), the relationships between DHB and DS are $m_\omega = d^1_\omega$, $\theta^1_\omega \approx d^2_\omega \Delta t$ and $\theta^2_\omega \approx d^3_\omega \Delta t$ (see Appendix B for the proofs). Hence, DHB invariant values related to the angular velocity represent the angular motion of the ISA in the discrete time domain. DHB invariants describe the motion of the ISA using angular velocities sampled at consecutive time instants, while DS invariants use high-order time derivatives. DS invariant values related to the linear velocity are obtained by projecting the linear velocity along the ISA, which guarantees the invariance to the reference point used to describe the translation [6]. As discussed in Sec. 5, the invariance to the reference point is not guaranteed by DHB and EFS representations.

7.2 DHB and EFS representations

The original Frenet-Serret (FS) representation [15] consists of three invariant values, corresponding to e^1_v, e^2_v and e^3_v in Tab. 2. e^1_v represents the linear velocity along the tangent axis of the Frenet-Serret frame, e^2_v and e^3_v describe the changing orientation of the FS frame. e^2_v and e^3_v are closely related to the curvature κ and torsion τ of a space curve, i.e. $e^2_v = \pm\kappa\|\mathbf{v}\|$ and $e^3_v = \pm\tau\|\mathbf{v}\|$. Recall that the curvature describes the change of the orientation angle of the tangent of a space curve per unit arc length, while the torsion describes the change of the orientation of the tangent plane of a space curve per unit arc length [15].

EFS [37] extends the FS representation by considering the orientation of the rigid body. In EFS a second FS frame is attached to the rigid body and three more invariants (e^1_ω, e^2_ω and e^3_ω in Tab. 2) are used to describe the rotation of the rigid body. The original trajectory can be reconstructed from EFS invariants by applying the method in [43] to both the FS frames. From Tab. 2, it is straightforward to show that $d^1_\omega = e^1_\omega$, $d^2_\omega = e^2_\omega$ and $d^3_\omega = e^3_\omega$. Another interesting property is that EFS and DS are exactly the same representation in case of pure translations. In this case, in fact, $\boldsymbol{\omega} = \mathbf{0}$ and only three invariants are defined. According to [6] and Tab. 2 it holds that $d^1_v = e^1_v$, $d^2_\omega = e^2_v$ and $d^3_\omega = e^3_\omega$. Note that the representation in [32] uses a subset (e^2_v and e^2_ω) of EFS invariants.

In the continuous time domain, DHB and EFS are almost same apart from a scaling factor: $m_u = e^1_u$, $\theta^1_u \approx e^2_u \Delta t$ and $\theta^2_u \approx e^3_u \Delta t$ for $u = v, \omega$ (see Appendix C for the proofs). The difference is that DHB is a discrete time version of EFS. EFS invariants describe the motion of the two FS frames assuming a continuous time domain and the effects of the numerical implementation are simply neglected. In contrast, DHB invariants describe the motion of the two (FS) frames directly assuming a discrete time domain: using velocities sampled at consecutive time instants instead of high order time derivatives. DHB and EFS share the same invariance properties (see Tab. 3), because both describe the motion of the two FS frames. In addition, DHB representation is also numerically robust because there is a one-to-one mapping between the representation and its numerical implementation. This methodological difference makes DHB invariants robust for practical use (see Sec. 8).

In [42], a robust discrete time approximation of FS invariants is proposed. Compared to [42], DHB representation considers also the orientation of the rigid body, uses less consecutive samples (3 instead of 5) and provides an accurate approach to reconstruct the motion.

One can see that θ^1_v, θ^2_v, θ^1_ω and θ^2_ω depend on the sampling time Δt. The sampling time dependency does not cause problems in usual cases, where most sensors receives data with relatively constant sampling time. In a special case where trajectories are sampled at very

[4] For simplicity, the acronym of the author name (DS) is used to refer the representation in [6].

[5] Time dependencies are omitted to simplify the notation.

Table 3 Overview and Properties of DS, EFS and DHB Representations.

Representation	Physical meaning	Bidirectional	Numerically robust	Invariant to Initial pose	Invariant to reference point	Invariant to viewpoint	Invariant to time/linear/angular scale
DS	Motion along ISA and motion of ISA	✓	–	✓	✓	✓	✓/ ✓/✓
EFS	Velocities (magnitude), orientation of FS frames	✓	–	✓	–	✓	✓/✓/✓
DHB	Discrete time approximation of EFS	✓	✓	✓	–	✓	✓/✓/✓

different rates, one can simply divide these invariants by the sampling time Δt.

7.3 Sample delay

To compute position-based invariants in a time instant t one has to know the Cartesian trajectory from t to $t+3$, introducing a delay of three samples. Indeed, the angles in (10) and (12) depend on $\hat{\mathbf{y}}_{t+1}$, which depends on $\hat{\mathbf{x}}_{t+2}$. From (1) and (4) it is clear that $\hat{\mathbf{x}}_{t+2}$ depends on the position (orientation) of the rigid body in $t+3$.

Following a similar reasoning, one can see from (13) and (16), that the velocity-based invariants in t require the velocities from t to $t+2$. Hence, a delay of two samples is introduced. Note that, when velocities are numerically computed, both DHB representations have the same delay of three samples.

Representations in [6, 37], instead, depend only on the current time instant, but they require the third-order derivative of position and orientation. When derivatives are numerically computed, the same delay of three samples is introduced. Thus, DS, EFS and DHB have same sample delay.

7.4 Computational cost

The computational cost of DHB invariant representation is[6] $O(N(M(n)(96+4\log(n))))$, where $M(n)$ is the computational cost of the multiplication (division), that depends on the number of digits n used to represent real numbers. N is the number of samples in the Cartesian trajectory. For comparison, the algorithm in [6] has a complexity of $O(N(180M(n)))$, while EFS invariants have a complexity of $O(N(100M(n)))$.

7.5 Trajectory reconstruction error

DHB, DS and EFS are bidirectional invariant representations, i.e. Cartesian twists can be retrieved from their representations. Original velocities are reconstructed from DHB descriptor by applying the algorithm in Sec. 4.2.

[6] The result is obtained from Algorithm 1 by neglecting the summation and subtraction operations.

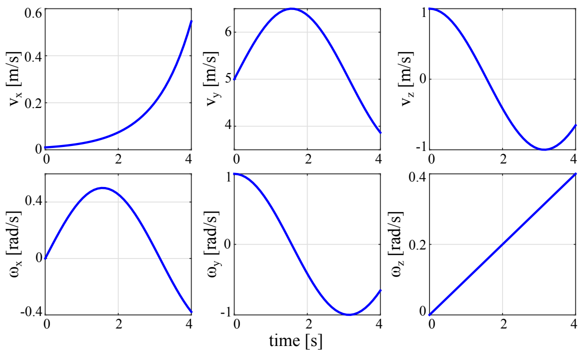


Fig. 3 Synthetic twist trajectory: $v_x = 0.01\exp(t)$, $v_x = 5+1.5\sin(t)$, $v_x = \cos(t)$, $\omega_x = 0.5\sin(t)$, $\omega_y = \cos(t)$ and $\omega_z = 0.1t$, $t = 0,\ldots,4\,\mathrm{s}$.

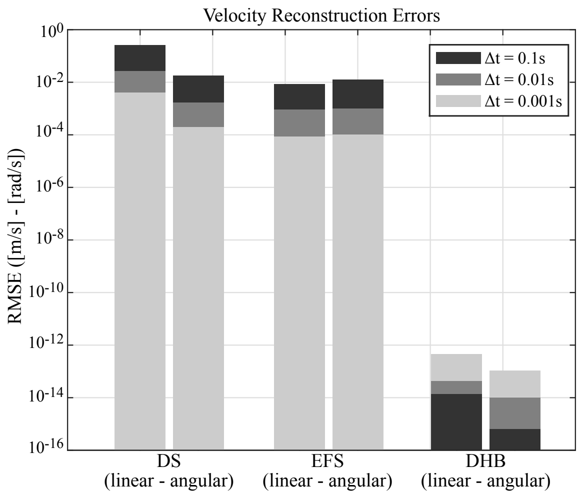


Fig. 4 Errors between the twists in Fig. 3 and the twists reconstructed from DS, EFS and DHB representations for different sampling times Δt. Note the logarithmic scale on the ordinate axis.

The reconstruction result of EFS is obtained by the algorithm in [43][7].

We test the reconstruction performance for all representations by using the synthetic twist trajectory in Fig. 3. The reconstruction error is computed as the root mean square error (RMSE) between original twists in Fig. 3 and twists reconstructed from each invariant representation. The results are shown in Fig. 4 using three different sampling times ($\Delta t = 0.1$, 0.01 and $0.001\,\mathrm{s}$).

[7] The reconstruction procedure in Sec. 4 can also be applied to EFS descriptor. Both reconstruction methods ([43] and Sec. 4) reproduce similar reconstruction errors.

One can see a non-negligible error with DS and EFS descriptors (see also Sec. 8.2.3). The accuracy of DS and EFS depends on the sampling time, i.e. the smaller the sampling time the smaller the error. DS and EFS representations, in fact, use high-order derivatives which are affected by round-off errors. Moreover, the numerical integration step is required to retrieve Cartesian twists from both DS and EFS representations. The linear RMSE for DS representation is always bigger than the linear RMSE for EFS invariants. This is because d_v^2 and d_v^3 are an approximated representation of the kinematics of the ISA [6].

On the other hand, the DHB invariants offer a high reconstruction accuracy for each value of Δt. Fig. 4 shows a negligible increase of reconstruction error for smaller sampling times, which is due to the finite precision of the computing machine[8].

7.6 Noise sensitivity

To show the noise sensitivity of invariant descriptors, a Gaussian noise with increasing power P_{noise} is added to the data in Fig. 3. The Gaussian noise is generated by considering a decreasing signal noise ratio (snr) from 100 dB to 50 dB, where $snr = P_{signal}/P_{noise}$. To measure the noise sensitivity of each representation, we first compute the invariant representation $\{i_n\}_t$ of the noisy twist trajectories. The invariant representation $\{i\}_t$ of the noiseless trajectories is then point-wise subtracted to $\{i_n\}_t$. The resulting set of samples $\{r_n\}_t$ represent the residual noise. The residual signal to noise ratio, i.e. the ratio between the original signal power $P_{\{i\}}$ and the residual noise power $P_{\{r_n\}}$, is used to compactly represent the noise sensitivity.

The described procedure is repeated 100 times for each value of the snr and for each invariant representation. Figure 5 shows the mean and standard deviation of the residual snr. The DHB representation exhibits a reduced noise sensitivity, compared to DS and EFS representations. The reason is that DHB invariants lie at velocity level, while DS and EFS invariants lie at jerk level. The numerical computation of high order derivatives is sensitive to the noise in the data and round-off errors [5]. DS invariants show the highest noise sensitivity. Indeed, the projection of the linear velocity (and its time derivatives) along the ISA axis increases the noise, especially in d_v^2 and d_v^3 in Tab. 2.

[8] A smaller sampling time generates more twist samples and more invariant values. Hence, more products have to be computed in (39) to reconstruct the motion, which increases errors due to the finite precision.

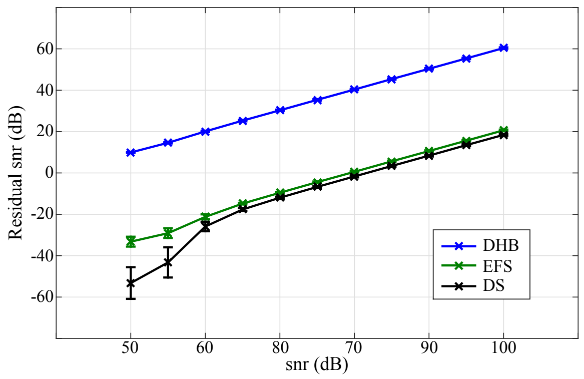


Fig. 5 Noise sensitivity of DHB, DS and EFS invariant representations when a Gaussian noise with increasing power is applied to the twists in Fig. 3. Crosses represent the mean and bars represent the standard deviation of the residual snr over the 100 iterations.

8 Experimental Results

8.1 Invariant Motion Recognition

In this experiment, we aim at demonstrating the effectiveness of invariant representations in gesture recognition problems. In order to focus on motion representations, we adopt a widely-used classification algorithm, the k-Nearest Neighbour classifier [2], instead of developing a customized classification algorithm. The k-Nearest Neighbour (k-NN) algorithm is a non-parametric classifier that assigns the query gesture to the class most common among its k nearest neighbors. Nearest neighbors are determined in this work considering the Euclidean distance among feature vectors.

Recognition performances of DHB representation is compared with state-of-the-art representations on three datasets (see Tab. 4). The first dataset, namely the English letters dataset, consists of five gestures performed by the same user. The dataset is relatively simple, but useful to understand how to apply invariant representations in gesture recognition problems. The second dataset, namely the pouring drink dataset, consists of five gestures performed by five different users. This dataset is more challenging than the first due to multiple users. The last dataset, namely the Microsoft research Action 3D dataset, consists of twenty actions performed by ten users. The relatively big number of actions/users makes this dataset a challenging benchmark for gesture recognition algorithms.

We show only velocity-based DHB invariants for gesture recognition, because the two DHB representations are practically the same (shown in Sec. 3.4) and have the same performance in term of gesture recognition.

Table 4 Datasets characteristics.

Dataset	# of Actions	# of Users	# of Repetitions	# of Body parts	Sampling rate [Hz]
English letters	5	1	50	1	30
Pouring drink	5	5	10	1	120
MSR Action3D	20	10	3	20	30

8.1.1 *English letters dataset*

In this section, we compare recognition performances of invariant and non-invariant motion descriptors using the first dataset[9], including the five capital letters A, M, N, O, X. Each alphabet letter is drawn by one user ten times. Motion trajectories are collected by tracking the user's right hand position at 30 Hz with an RGB-D camera. Ten demonstrations of the letters O and X are shown in Fig. 6.

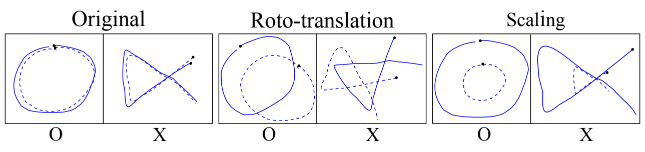


Fig. 6 Original demonstrations of O and X (left), affects by roto-translation (middle) and by scaling (right) are visualized in the x-y coordinates. Black points indicate the starting position of each trajectory.

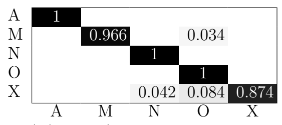
In real scenarios gestures can be observed or executed from different starting poses. To simulate this situation, the dataset is extended to 250 demonstrations (50 demonstrations $\times$ 5 letters). Random affine transformations are applied to the original data and generate 40 additional demonstrations per letter, shown in Fig. 6. Time derivatives are numerically computed with the sampling time $\Delta t = 1/30\,\text{s}$, the same frame rate at which data are collected.

Recognition performances of different motion descriptors are compared using a leave-half-out cross validation approach, where a half of the dataset is used as a training set and the rest as a test set. Training and test sets are randomly selected 100 times. Average recognition rates for different values of k (1-NN, 3-NN and 5-NN) and with different filtering conditions (with original data versus filterd data) are shown in Tab. 5. Confusion matrices for DHB and DS/EFS representations for 1-NN are displayed in Fig. 7. Results are obtained considering all the invariant values of each representation. Recall that EFS and DS are exactly the same for pure translations (See Sec. 7.2).

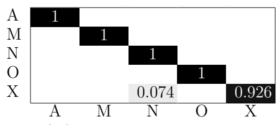
[9] Available on-line: creativedistraction.com/downloads/gesture.zip.

Table 5 Recognition results on the English letters dataset.

	Unfiltered data			Filtered data		
	1-NN	3-NN	5-NN	1-NN	3-NN	5-NN
Twist	78.8%	72.7%	69.6%	79.3%	74.1%	70.4%
DS / EFS	79.56%	79.24%	78.92%	96.96%	96.44%	96.36%
DHB	**94.76%**	**94.36%**	**94.32%**	**98.52%**	**98.32%**	**98.24%**



(a) DS/EFS invariants



(b) DHB invariants

Fig. 7 Confusion matrices for the English letters dataset obtained with filtered data and $k = 1$.

According to Tab. 5, the Cartesian twist shows poor performance compared to DS/EFS and DHB since the Cartesian twist is invariant only to translations. DS/EFS are ill-conditioned on noisy signals due to the numerical computation of high order derivatives. When a moving average filter (window size $w = 5$) is applied, the performance of DS/EFS is improved a lot. A more sophisticated, off-line filtering technique as the linear Kalman smoother [29] used in [37, 7] could further improve the performance. Nevertheless, the usage of off-line filtering techniques limits the applicability of invariant representations in on-line gesture recognition problems and it increases the computational cost of the recognition procedure. The proposed DHB invariants show the best recognition rate in all considered cases, which implies that DHB invariants are well-conditioned also on noisy data.

8.1.2 *Pouring a drink dataset*

The aim of this experiment is to show the recognition capabilities of the DHB invariants among different users. The dataset consists of five actions, namely *pour, release, start, stop* and *take*, performed with the right hand ten times by five users (see Tab. 4). An expert user shows once the gesture to perform. Users are asked to repeat the shown action ten times in a row, arbitrarily changing their pose after each repetition. Cartesian twists of the users right hand are collected at 120 Hz using a Xsens MVN motion capture suit[10]. The origin of the inertial sensor attached to the right hand is used as reference point. Time derivatives are numerically computed with the sampling time $\Delta t = 1/120\,\text{s}$, the same frame rate of the data collection.

Since all representations performed better with filtered data on the English letters dataset, we filter twist trajectories using a moving average filter ($w = 5$) to improve the recognition rate. Performances of DHB, DS

[10] www.xsens.com/products/xsens-mvn.

and EFS representations are compared using a leave-half-out cross validation approach, which uses half of the demonstrations of each user for training and the rest for testing. Training and test sets are randomly selected 100 times. Average recognition rates for different values of k are shown in Tab. 6. If all the six invariants values are considered, DS representation achieves poor recognition performance compared to DHB and EFS. DS invariants have an increased noise sensitivity due to the projection of linear velocities (and their time derivatives) along the ISA.

In order to illustrate the effects of jerk dependent invariants explicitly, another recognition test is performed and the results are shown in Tab. 6 and Fig. 8. Therein, the jerk depended values are removed: d_v^3 and d_ω^3 from DS and e_v^3 and e_ω^3 from EFS. For the fair comparison, we also remove θ_v^2 and θ_ω^2, which describe similar quantities as e_v^3 and e_ω^3, from our DHB invariants. Table 6 shows that jerk depended values of DS invariants degrade the recognition performance and recommends to use a subset instead of the full set of DS invariants. Recognition performance of EFS and DHB invariants are comparable and slightly better performance of DHB on the full set of invariants and EFS on the subset was observed. This means that all the six values in DHB representation contribute to make motions more distinctive.

Table 6 Recognition results on the pouring a drink dataset (filtered data).

	All invariants			Without jerk dependent invariants		
	1-NN	3-NN	5-NN	1-NN	3-NN	5-NN
DS	77.45%	80.38%	80.59%	93.66%	92.72%	91.06%
EFS	94.02%	92.86%	91.5%	94.66%	**93.24%**	**91.95%**
DHB	**95.14%**	**93.45%**	**92.08%**	**94.89%**	93.22%	91.9%

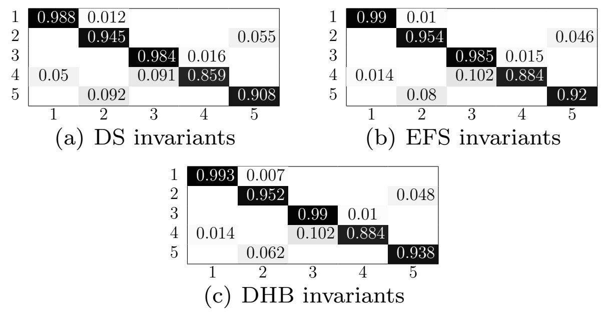


Fig. 8 Confusion matrices for the pouring a drink dataset. Results are obtained with $k = 1$ and without considering jerk dependant invariants in DS and EFS representations.

Table 7 MSR action Dataset and Recognition Protocol

MSR Action3D dataset: Twenty actions are performed 3 times by 10 users. The 20 actions are highArmWave, horizontalArmWave, hammer, handCatch, forwardPunch, highThrow, drawX, drawTick, drawCircle, handClap, twoHandWave, sideBoxing, bend, forwardKick, sideKick, jogging, tennisSwing, tennisServe, golfSwing, and pickUpThrow.

Action Subsets: The MSR Action3D dataset is split into three action subsets (AS1, AS2 and AS3).

AS1	AS2	AS3
horizontalArmWave	highArmWave	highThrow
hammer	handCatch	forwardKick
forwardPunch	drawX	sideKick
highThrow	drawTick	jogging
handClap	drawCircle	tennisSwing
bend	twoHandWave	tennisServe
tennisServe	sideBoxing	golfSwing
pickUpThrow	forwardKick	pickUpThrow

Cross validations: Three tests (T1, T2 and T3) are performed on each action set. In T1, 1/3 of the samples, i.e. one demonstration for each user, are used for training and the rest for testing. In T2, 2/3 of the samples are used as training set and the rest as test set. In T3, demonstrations from half of the users are used for training and the rest for testing. In T1 and T2 data from all the subjects are considered for the training. T3 is a cross subject test.

8.1.3 MicroSoft Research (MSR) Action3D dataset

To evaluate performances in a more challenging scenario, we consider a bigger dataset of full-body motions. The MSR Action3D dataset[11] consists of 20 actions (see Tab. 7) performed 3 times by 10 users. Each user is tracked using an RGB-D sensor at 30 Hz. For each time instant, the position (pure translation) of 20 body parts is stored. Motion recognition is tested by following the experimental protocol described in [45], summarized in Tab. 7.

For each action, the invariant representation of each body part is computed. The origin of the frames attached to each body part is used as reference point. Position trajectories are filtered with a moving average filter ($w = 5$) before computing the invariants. Time derivatives of the position are computed by numerical differentiation with sampling time $\Delta t = 1/30$ s. As underlined in [32] and [39], full-body motions are better recognized when body parts not involved in the movement are not considered. Unused body parts are automatically cut off by applying a thresholding method. We consider a body part P relevant to the current action if P is moved more than $t_r = 0.15$ m, where the value of t_r is chosen empirically. The invariants relative to irrelevant body parts are simply zeroed.

[11] Available on-line: research.microsoft.com/en-us/um/people/zliu/actionrecorsrc.

Table 8 Recognition results of DHB, DS/EFS, HOJ3D [45] and Li et al. [19] on the MSR Action3D dataset.

	T1				T2				T3			
	AS1	AS2	AS3	Average	AS1	AS2	AS3	Average	AS1	AS2	AS3	Average
DS/EFS	89.17%	89.17%	98.58%	92.31%	88.51%	88.73%	97.17%	91.47%	87.92%	86.85%	98.22%	91.0%
DHB	97.65%	91.84%	**100%**	**96.49%**	97.96%	93.74%	**100%**	**97.23%**	**96.42%**	**91.3%**	**99.98%**	**95.9%**
HOJ3D	**98.47%**	**96.67%**	93.47%	96.2%	**98.61%**	**97.92%**	94.93%	97.15%	87.98%	85.48%	63.46%	78.97%
Li et al.	89.5%	89.0%	96.3%	91.6%	93.4%	92.9%	96.3%	94.2%	72.9%	71.9%	79.2%	74.7%

Training and test sets are randomly selected 100 times. Average recognition rates with different motion representations are shown in Tab. 8. Results for DHB and DS/EFS are obtained using a 1-NN classifier and considering all three invariant values[12] for each joint. DHB invariants have, on average, the highest recognition rate in all the considered cases. The unidirectional HOJ3D approach in [45] performs slightly better than DHB invariants only on AS1/AS2 in test T1/T2. Actions in AS1 and AS2 are mainly performed with the right arm, and some actions consist of similar movements executed in different directions, such as forwardPunch and highThrow. Unlikely to DHB, DS and EFS, HOJ3D contains information about the direction of the human motion in Cartesian space. This could explain why HOJ3D can distinguish slightly better similar motions in different directions of the Cartesian space on AS1/AS2 in test T1/T2. However when it comes to the cross subject test, the performance of HOJ3D downgrades significantly because it is not invariant to scale. Finally, the results in Tab. 8 show that our representation has the highest recognition rate on AS3 and in the cross subject test T3, showing good generalization capabilities among different users.

8.2 Invariant Motion Reproduction

The reconstruction procedure in Sec. 4 can be further extended to generate different trajectory instances. The first experiment shows how different instances of a Cartesian motion are generated from the same DHB representation. The second experiment focuses on generating a feasible trajectory for the NAO humanoid robot[13] starting from human demonstrations. The third experiment shows how a complex task, consisting of three different actions, can be effectively executed on a real robot. The last experiment shows how full-body motions can be reproduced from the invariant trajectory of each body part.

[12] There exists 3 invariants to represent translational motion of the MSR Action3D dataset.

[13] www.aldebaran.com/en/cool-robots/nao.

8.2.1 Multiple trajectories from the same descriptor

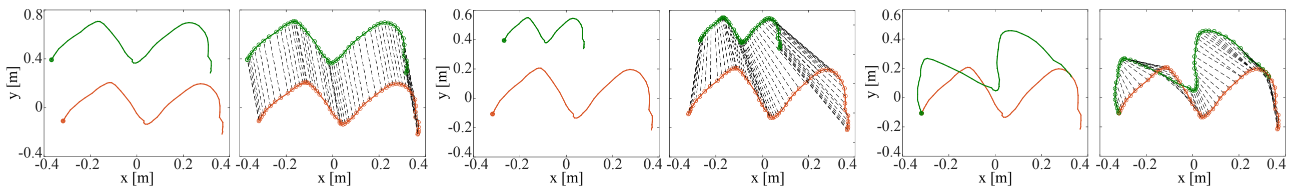
Affine transformed instances of a Cartesian trajectory can be generated from the same invariant descriptor by reconstructing the Cartesian motion in an arbitrary reference frame with an arbitrary scaling factor. Consider one demonstration of the letter M in Sec. 8.1.1, which is shown in Fig. 9(c) as a red curve. The demonstration is converted into the DHB descriptor and the invariant representation can be reproduced with a various form of affine transformations, shown in Fig. 9(c) as green lines. In Fig. 9(a), the motion is reconstructed in a translated ($\mathbf{t} = [-0.05, 0.5, 0]\,\text{m}$) initial frame. The same translation $\mathbf{t}$ is applied in Fig. 9(b) together with a scaling factor $s = 0.5$. The initial frame is rotated of $-30\,\text{deg}$ around the z axis in Fig. 9(c).

The correspondences between the original demonstration and the generated trajectories are visualized with the black dashed lines. The correspondences are calculated by a distance metric based on dynamic time warping (DTW) [30]. In particular multi dimensional DTW (MD-DTW) algorithm in [31] was used to find the optimal non-linear match (see Fig. 9).

8.2.2 English letters reproduction

This experiment illustrates how to generate a robot hand motion from a human hand motion using the proposed DHB descriptor. One human demonstration of the letter N in Sec. 8.1.1 is reproduced by a small humanoid robot NAO. The robot executes the trajectory with the right hand (see Fig. 10). Even with unknown camera frame, the DHB representation can directly generate the Cartesian trajectory in the robot reference frame without any processing of the DHB representation.

Due to the physical limitations of the NAO robot, the Cartesian velocities reconstructed considering the "human" linear scale s_{hum} in (42) cannot be executed by the robot. Comparing the maximum velocity of the robot hand, as well as its arm length, with the velocity required to execute the original motion, a suitable trajectory for NAO is generated considering a scaling factor $s_{nao} = (1/12)s_{hum}$ (see Fig. 10). Reconstructed velocities are converted into joint angle references by



(a) Translation (b) Translation and scaling (c) Rotation

Fig. 9 Affinely transformed instances (green solid lines) of the letter M generated from the same DHB representation. The orange solid line is the original letter. Black dashed lines are the MD-DTW alignment between original and generated trajectories. For a better visualization, only the x and y axes are shown.

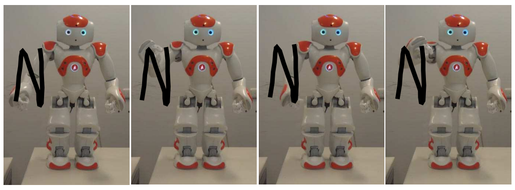


Fig. 10 Snapshots of the reproduction of the letter N.

means of inverse kinematics algorithms [35] and sent to the robot at 30 Hz. Note that, as for roto-translations, scaled motion variants are generated from the same invariant descriptor.

The procedure to find a suitable scaling s_{nao} can be automatized. When the physical scaling of a robot and a human is roughly known, it is a good starting point. [14] When the relative physical scaling is unknown, a robot scaling is started with $s_{nao} = s_{hum}$. With the current s_{nao}, one generates desired velocities and positions and check if robot constraints are violated. In case of the constraint violations, one reduces the scaling s_{nao} and continues until no violations occur. This procedure can be also used to reproduce the same motion on different robots.

8.2.3 Pouring a drink

This experiment shows how to combine different invariant descriptors to execute a complex task. A pouring-a-drink task, whose subtasks were originally demonstrated by a human in Sec. 8.1.2, was executed by the Kuka light-weight (LWR) manipulator [34]. The task is a sequence of three actions: *take* the bottle, *pour* the liquid in a cup, *release* the bottle. The DHB representation of each action (first demonstration from the first user) is used as motion descriptor.

As detailed in previous experiments, the invariance to roto-translations allows to generate the same Cartesian trajectory from different initial configurations. The end-effector pose at the beginning of each action is

[14] For example, for full body motions of a human/humanoid, their heights are the reference. For hand motion, the length of its arm/manipulation are useful.

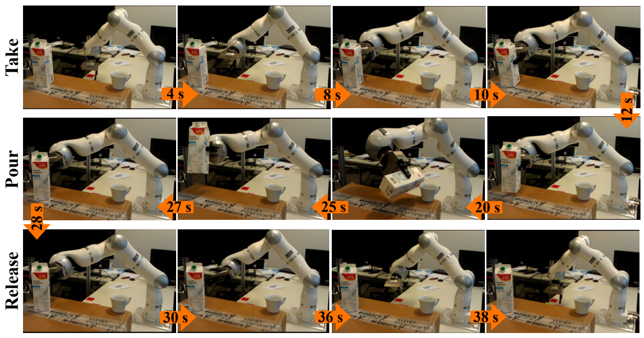


Fig. 11 Snapshots of the pouring a drink task. The task consists of three consecutive actions: take, pour and release.

used to reconstruct on-line the desired velocities. In this way, the three actions are sequentially executed without jumps. Hence, the invariance to roto-translations of DHB representation simplifies the problem of on-line generating feasible sequential tasks.

The desired Cartesian pose is directly sent to the LWR through the fast research interface [34]. The desired pose is computed by numerically integrating the reconstructed velocity. The robot is controlled at 1 KHz while data are sampled at 120 Hz, which is the data collection sampling rate of the Xsens MVN motion capture suit. To match the time difference between the control loop and the data sampling, consecutive poses are linearly interpolated. Thirty extra samples are added in each time interval to match the maximum velocity increment that the robot allows. Different scaling factors s_{lwr}, as well as the invariance to roto-translations, are used to reach the bottle (cup) placed in different locations. A suitable affine transformation can be automatically selected by recognizing and tracking the bottle (cup) using an external camera sensor. The execution of the pouring task is shown in Fig. 11. Note that, since orientation control is needed to execute the pouring task, the approach in [43] is not applicable for this task.

In order to grasp the bottle and to pour the liquid into the cup, reaching the desired positions, in other words accurate motion generation, is of importance.

Hence, in this case, it is interesting to compare the reconstruction errors of different bidirectional representations. As a proof of concepts, we consider the action *take*. For simplicity, the same reference frame used to collect the data was used and the orientation is neglected. The accuracy of motion generation is measured by considering the distance between the final reconstructed position and the original one (the desired cup location). Three filtering techniques are tested, namely the moving average filter, the anisotropic diffusion [22] and the linear Kalman smoother [29]. The action consists of 295 velocities sampled at 120 Hz.

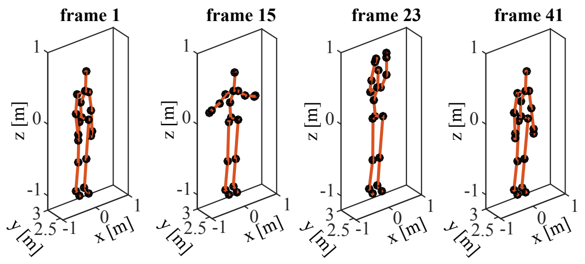
As shown in Tab. 9, the proposed reconstruction algorithm showed high accuracy. This is another empirical proof of the numerical robustness of DHB representation. Although the moving average filter improves the reproduction performance of the DS and EFS invariants, its error was larger than 141 mm and it is still not accurate enough for the pouring task. The anisotropic diffusion, which iteratively smooths the trajectory with a Gaussian kernel, is more effective. Iterating 1000 times, the error drops down to 1.15 mm. Our Matlab implementation of the anisotropic diffusion takes about 0.4 s to perform 1000 iterations over the 295 samples of the considered action, which implies its limited on-line use. The best result for DS/EFS invariants is obtained with the Kalman smoother, which is an off-line filtering technique.

Table 9 Norm of the reconstruction errors [mm] for the take action obtained with different filtering techniques.

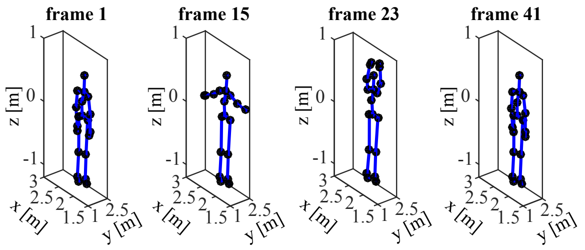
Moving average filter	**DS / EFS**	**DHB**
w=5	464	**2.7e-12**
w=10	191	**3e-12**
w=50	148	**2e-12**
w=100	141	**1.5e-12**
Anisotropic diffusion	**DS / EFS**	**DHB**
i=10	178	**2.7e-12**
i=100	14.5	**3.7e-12**
i=1000	1.15	**2.3e-12**
Linear Kalman smoother	0.56	**2.9e-12**

8.2.4 Full-body motion reproduction

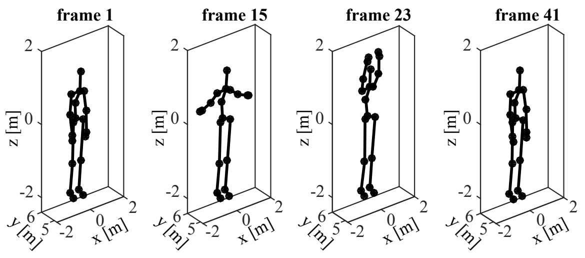
Full-body motion transfer from a human demonstrator to a humanoid robot is more complicated than reproducing the motion of a single rigid body. Humans and robots have different kinematic structures. The number of joints, the joint limits and the length of each body part can vary significantly. Mapping human motions to a humanoid robot considering differences in kinematics, dynamics and balancing stability is beyond the scope of this work but can be found in [10, 11]. Since this work



(a) Original action.



(b) Reconstructed action - Rotation (45 deg around z).



(c) Reconstructed action - Scaling ($s = 2$).

Fig. 12 Different instances of the *twoHandWave* action generated from the same DHB representation. The black bullets indicates the 20 body parts.

focuses on motion representations, we consider a simplified skeleton model: the same kinematic model of a human and a robot but with the different length scaling between them.

For a given motion in the dataset we directly compute the position of each body part from the relative invariant representation. Different motion instances can be generated from the same invariant representation of each body part. The experimental results with *twoHandWave* action of the MSRAction3D dataset are shown in Fig. 12. The original *twoHandWave* action of a human is visualized in Fig. 12(a). The same action performed by a robot using a different baselink pose can be reproduced from the same DHB descriptor. Fig. 12(b) shows the obtained results by applying to the reference frame a rotation of 45 deg along the z axis. Similarly the same action for a different size can be planned. Fig. 12(c) is reconstructed with a scaling factor of 2 (twice the original one) applied to all the body parts.

9 Conclusion

We presented an extended study on Denavit-Hartenberg inspired Bidirectional (DHB) invariants, a novel representation of rigid body motions. Two representations are described, namely position and velocity-based invariants. Position-based invariants are useful when the Cartesian pose is directly available from the sensors. If the Cartesian velocity can be measured, velocity-based invariants directly apply without a preliminary numerical integration. A clear separation is introduced between position (linear velocity) and orientation (angular velocity) of the rigid body motion, which makes these representations suitable for various applications. Singular cases can be easily detected and solved. The DHB representations have the following properties:

- minimal representation (six values)
- invariance to affine transformations
- invariance to time scale
- invariance to the direction of execution (reverse motions)
- numerical robustness
- bi-directionality

We have analytically proven the invariance properties of the proposed DHB representations. The DHB invariants are theoretically compared with other state-of-the-art representations, including the analytical similarity between DHB and EFS. An extensive evaluation and comparison is conducted on synthetic and real datasets. In the motion recognition tests using three datasets, the proposed DHB descriptor showed better recognition performance in most of the cases compared to existing uni- and bi-directional representations. The effectiveness and flexibilities of the DHB descriptor for robot motion generation were demonstrated by experiments with NAO, KUKA LWR robot, and a humanoid robot. Moreover the numerical robustness of DHB was shown with synthetic and real sensor data.

A Rigid Body Motion Representation

To represent rigid body motions it is convenient to attach an orthogonal frame to the rigid body (body frame) and to describe the pose (position and orientation) of the body frame wrt a fixed frame (world frame). In each time instant the position of the rigid body is represented by the vector $\mathbf{p}$ connecting the origin of the body frame with the origin of the world frame. The axes of the body frame can be projected along the axes of the world frame by the means of the *direction cosines*. Hence, the orientation of the rigid body is described by collecting the direction cosines into a 3×3 rotation matrix $\mathbf{R}$. It is possible to show that a minimal representation of the orientation consists of 3 values [35]. In this work, we use the *rotation vector* to represent the orientation.

The rotation vector $\mathbf{r} = \theta\hat{\mathbf{r}}$ is computed from $\mathbf{R}$ as:

$$\theta = \arccos\left(\frac{trace\,(\mathbf{R}) - 1}{2}\right), \hat{\mathbf{r}} = \frac{1}{2\sin\theta}\begin{bmatrix}\mathbf{R}\,(3,2) - \mathbf{R}\,(2,3)\\ \mathbf{R}\,(1,3) - \mathbf{R}\,(3,1)\\ \mathbf{R}\,(2,1) - \mathbf{R}\,(1,2)\end{bmatrix}$$

The rotation matrix $\mathbf{R}$ is computed from $\mathbf{r}$ as:

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

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

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

B Proofs of the relationships in Sec. 7.1

1) $m_\omega = d_\omega^1$ derives from (20) and d_ω^1 in Tab. 2.
2) $\theta_\omega^1 \approx d_\omega^2 \Delta t$. For $\Delta t \longrightarrow 0$, we can neglect the arc tangent in (28). Hence, we can rewrite θ_ω^1 in (23) as:

$$\begin{aligned}\theta_\omega^1 &\approx \frac{\|\boldsymbol{\omega}_t \times \boldsymbol{\omega}_{t+1}\|}{\boldsymbol{\omega}_t \cdot \boldsymbol{\omega}_{t+1}} = \frac{\|\boldsymbol{\omega}_t \times (\boldsymbol{\omega}_t + \Delta\boldsymbol{\omega}_t)\|}{\boldsymbol{\omega}_t \cdot (\boldsymbol{\omega}_t + \Delta\boldsymbol{\omega}_t)} \\ &\approx \frac{\|\boldsymbol{\omega}_t \times \Delta\boldsymbol{\omega}_t\|}{\|\boldsymbol{\omega}_t\|^2}\frac{\Delta t}{\Delta t} \approx \frac{\|\boldsymbol{\omega}_t \times \dot{\boldsymbol{\omega}}_t\|}{\|\boldsymbol{\omega}_t\|^2}\Delta t = d_\omega^2 \Delta t\ .\end{aligned} \tag{43}$$

3) $\theta_\omega^2 \approx d_\omega^3 \Delta t$. Recall that $\mathbf{a}\times\mathbf{b} = -\mathbf{b}\times\mathbf{a}$ and that $\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c}) = \mathbf{c}\cdot(\mathbf{a}\times\mathbf{b})$. θ_ω^2 in (23) can be re-written as:

$$\begin{aligned}\theta_\omega^2 &= \arctan\left(\frac{\|\boldsymbol{\omega}_{t+1}\|\boldsymbol{\omega}_{t+2} \cdot (\boldsymbol{\omega}_{t+1} \times \boldsymbol{\omega}_t)}{(\boldsymbol{\omega}_{t+1} \times \boldsymbol{\omega}_t)\cdot(\boldsymbol{\omega}_{t+1} \times \boldsymbol{\omega}_{t+2})}\right) \\ &= \arctan\left(\frac{\|\boldsymbol{\omega}_{t+1}\|\,(\boldsymbol{\omega}_t \times \boldsymbol{\omega}_{t+1})\cdot\boldsymbol{\omega}_{t+2}}{(\boldsymbol{\omega}_t \times \boldsymbol{\omega}_{t+1})\cdot(\boldsymbol{\omega}_{t+1} \times \boldsymbol{\omega}_{t+2})}\right)\end{aligned} \tag{44}$$

The denominator of (44) can be re-written as:

$$\begin{aligned}&(\boldsymbol{\omega}_t \times \boldsymbol{\omega}_{t+1})\cdot(\boldsymbol{\omega}_{t+1} \times \boldsymbol{\omega}_{t+2}) \\ &\approx (\boldsymbol{\omega}_t \times \Delta\boldsymbol{\omega}_t)\cdot[(\boldsymbol{\omega}_t \times \Delta\boldsymbol{\omega}_t) \times (\boldsymbol{\omega}_t \times 2\Delta\boldsymbol{\omega}_t)] \\ &= (\boldsymbol{\omega}_t \times \Delta\boldsymbol{\omega}_t)\cdot[2(\boldsymbol{\omega}_t \times \Delta\boldsymbol{\omega}_t) - (\boldsymbol{\omega}_t \times \Delta\boldsymbol{\omega}_t)]\,\frac{\Delta t^2}{\Delta t^2} \\ &\approx (\boldsymbol{\omega}_t \times \dot{\boldsymbol{\omega}}_t)\cdot(\boldsymbol{\omega}_t \times \dot{\boldsymbol{\omega}}_t)\,\Delta t^2 = \|\boldsymbol{\omega}_t \times \dot{\boldsymbol{\omega}}_t\|^2 \Delta t^2\end{aligned} \tag{45}$$

Considering that $\ddot{\boldsymbol{a}}_t \approx (\boldsymbol{a}_{t+2} + \boldsymbol{a}_t)/\Delta t^2$, the numerator of (44) can be re-written as:

$$\begin{aligned}&\|\boldsymbol{\omega}_{t+1}\|\,(\boldsymbol{\omega}_t \times \boldsymbol{\omega}_{t+1})\cdot\boldsymbol{\omega}_{t+2} \approx \|\boldsymbol{\omega}_t\|\,(\boldsymbol{\omega}_t \times \Delta\boldsymbol{\omega}_t)\cdot \\ &(\ddot{\boldsymbol{\omega}}_t\Delta t^2 - \boldsymbol{\omega}_t) \approx \|\boldsymbol{\omega}_t\|\,(\boldsymbol{\omega}_t \times \dot{\boldsymbol{\omega}}_t)\cdot\ddot{\boldsymbol{\omega}}_t\Delta t^3\end{aligned} \tag{46}$$

Finally, combining (45), (46) and (44), and neglecting the arc tangent, we obtain that $\theta_\omega^2 \approx d_\omega^3 \Delta t$ for $\Delta t \longrightarrow 0$.

C Proofs of the relationships in Sec. 7.2

1) $m_v = e_v^1$ derives from (19) and e_v^1 in Tab. 2. $m_\omega = e_\omega^1$ derives from (20) and e_ω^1 in Tab. 2.
2) $\theta_\omega^1 \approx e_\omega^2 \Delta t$ derives from (43) recalling that $e_\omega^2 = d_\omega^2$. $\theta_v^1 \approx e_v^2 \Delta t$ can be proven by following similar steps as in (43) and considering e_v^2 in Tab. 2.
3) $\theta_v^2 \approx e_v^3 \Delta t$ and $\theta_\omega^2 \approx e_\omega^3 \Delta t$. Following similar steps as in (45) and (46), and recalling that $(\mathbf{a}\times\mathbf{b})\times(\mathbf{a}\times\mathbf{c}) = [\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c})]\,\mathbf{a}$, it is possible to prove that $\theta_v^2 \approx e_v^3 \Delta t$ and $\theta_\omega^2 \approx e_\omega^3 \Delta t$.

Acknowledgments

This work has been supported by the Technical University of Munich, International Graduate School of Science and Engineering.

References

1. Billard A, Calinon S, Dillmann R, Schaal S (2008) Robot programming by demonstration. In: Siciliano B, Khatib O (eds) Springer Handbook of Robotics, Springer Berlin Heidelberg, pp 1371–1394
2. Bishop CM, et al (2006) Pattern recognition and machine learning. Springer New York
3. Black M, Jepson D (1998) A probabilistic framework for matching temporal trajectories: Condensation-based recognition of gestures and expressions. In: European Conference on Computer Vision, Lecture Notes in Computer Science, vol 1406, Springer Berlin Heidelberg, pp 909–924
4. Burger B, Ferrané I, Lerasle F, Infantes G (2011) Two-handed gesture recognition and fusion with speech to command a robot. Autonomous Robots 32(2):129–147
5. Chartrand R (2011) Numerical differentiation of noisy, nonsmooth data. ISRN Applied Mathematics 2011:1–12
6. De Schutter J (2010) Invariant description of rigid body motion trajectories. Journal of Mechanisms and Robotics 2(1):1–9
7. De Schutter J, Di Lello E, De Schutter J, Matthysen R, Benoit T, De Laet T (2011) Recognition of 6 dof rigid body motion trajectories using a coordinate-free representation. In: International Conference on Robotics and Automation, pp 2071–2078
8. Denavit J, Hartenberg RS (1965) A kinematic notation for lower-pair mechanisms based on matrices. Transaction of the ASME Journal of Applied Mechanics 22(2):215–221
9. Dieleman S, De Fauw J, Kavukcuoglu K (2016) Exploiting cyclic symmetry in convolutional neural networks. International Conference on Machine Learning
10. Hu K, Lee D (2012) Biped locomotion primitive learning, control and prediction from human data. In: 10th International IFAC Symposium on Robot Control (SYROCO)
11. Hu K, Ott C, Lee D (2014) Online human walking imitation in task and joint space based on quadratic programming. In: IEEE Int. Conf. on Robotics and Automation, IEEE, pp 3458–3464
12. Isard M, Blake A (1996) Contour tracking by stochastic propagation of conditional density. In: European Conference on Computer Vision, pp 343–356
13. Jaderberg M, Simonyan K, Zisserman A, Kavukcuoglu K (2015) Spatial transformer networks. In: Advances in Neural Information Processing Systems, pp 2017–2025
14. Koppula HS, Gupta R, Saxena A (2013) Learning human activities and object affordances from rgb-d videos. International Journal of Robotic Research 32(0):951–970
15. Kühnel W (2006) Differential geometry: curves - surfaces - manifolds. American Mathematical Society
16. LeCun Y (2012) Learning invariant feature hierarchies. In: European Conference on Computer Vision, pp 496–505
17. Lee D, Nakamura Y (2010) Mimesis model from partial observations for a humanoid robot. International Journal of Robotics Research 29(1):60–80
18. Lee D, Ott C, Nakamura Y (2009) Mimetic communication with impedance control for physical human-robot interaction. In: IEEE Int. Conf. on Robotics and Automation, pp 1535–1542
19. Li W, Zhang Z, Liu Z (2010) Action recognition based on a bag of 3d points. In: Conference on Computer Vision and Pattern Recognition Workshops, pp 9–14
20. Magnanimo V, Saveriano M, Rossi S, Lee D (2014) A bayesian approach for task recognition and future human activity prediction. In: International Symposium on Robot and Human Interactive Communication, pp 726–731
21. Murray RM, Sastry SS, Zexiang L (1994) A Mathematical Introduction to Robotic Manipulation, 1st edn. CRC Press
22. Perona P, Malik J (1990) Scale-space and edge detection using anisotropic diffusion. Transactions on Pattern Analysis and Machine Intelligence 12(7):629–639
23. Piao Y, Hayakawa K, Sato J (2002) Space-time invariants and video motion extraction from arbitrary viewpoints. In: International Conference on Pattern Recognition, pp 56–59
24. Piao Y, Hayakawa K, Sato J (2004) Space-time invariants for recognizing 3d motions from arbitrary viewpoints under perspective projection. In: International Conference on Image and Graphics, pp 200–203
25. Psarrou A, Gong S, Walter M (2002) Recognition of human gestures and behaviour based on motion trajectories. Image and Vision Computing 20(5–6):349–358
26. Rabiner LR (1989) A tutorial on hidden markov models and selected applications in speech recognition. In: Proceedings of the IEEE, pp 257–286
27. Rao C, Yilmaz A, Shah M (2002) View-invariant representation and recognition of actions. International Journal of Computer Vision 50(2):203–226
28. Rao C, Shah M, Syeda-Mahmood T (2003) Action recognition based on view invariant spatio-temporal analysis. In: ACM Multimedia
29. Rauch HE, Striebel CT, Tung F (1965) Maximum likelihood estimates of linear dynamic systems. Journal of the American Institute of Aeronautics and Astronautics 3(8):1445–1450
30. Sakoe H, Chiba S (1978) Dynamic programming algorithm optimization for spoken word recognition. Transactions on Acoustics, Speech, and Signal Processing pp 43–49
31. Sanguansat P (2012) Multiple multidimensional sequence alignment using generalized dynamic time warping. WSEAS Transactions on Mathematics 11(8):668–678
32. Saveriano M, Lee D (2013) Invariant representation for user independent motion recognition. In: International Symposium on Robot and Human Interactive Communication, pp 650–655
33. Saveriano M, An S, Lee D (2015) Incremental kinesthetic teaching of end-effector and null-space motion primitives. In: International Conference on Robotics and Automation, pp 3570–3575
34. Schreiber G, Stemmer A, Bischoff R (2010) The fast research interface for the kuka lightweight robot. In: ICRA workshop on Innovative Robot Control Architectures for Demanding (Research) Applications, pp 15–21
35. Siciliano B, Sciavicco L, Villani L, Oriolo G (2009) Robotics - Modelling, Planning and Control. Springer
36. Soloperto R, Saveriano M, Lee D (2015) A bidirectional invariant representation of motion for gesture recognition and reproduction. In: International Conference on

Robotics and Automation, pp 6146–6152
37. Vochten M, De Laet T, De Schutter J (2015) Comparison of rigid body motion trajectory descriptors for motion representation and recognition. In: International Conference on Robotics and Automation, pp 3010–3017
38. Waldherr S, Romero R, Thrun S (2000) A gesture based interface for human-robot interaction. Autonomous Robots 9(2):151–173
39. Wang J, Liu Z, Wu Y, Yuan J (2012) Mining actionlet ensemble for action recognition with depth cameras. In: Conference on Computer Vision and Pattern Recognition, pp 1290–1297
40. Wang P, Li W, Gao Z, Tang C, Zhang J, Ogunbona P (2015) Convnets-based action recognition from depth maps through virtual cameras and pseudocoloring. In: Proceedings of the 23rd ACM international conference on Multimedia, pp 1119–1122
41. Weiss I (1993) Geometric invariants and object recognition. International Journal of Computer Vision 10(3):207–231
42. Wu S, Li YF (2008) On signature invariants for effective motion trajectory recognition. International Journal of Robotic Research 27(8):895–917
43. Wu S, Li YF (2010) Motion trajectory reproduction from generalized signature description. Pattern Recognition 43(1):204–221
44. Wu Y, Huang TS (2001) Vision-based gesture recognition: A review. In: Gesture-Based Communication in Human-Computer Interaction, Lecture Notes in Computer Science, Springer Berlin Heidelberg, pp 103–115
45. Xia L, Chen CC, Aggarwal JK (2012) View invariant human action recognition using histograms of 3d joints. In: Conference on Computer Vision and Pattern Recognition Workshops, pp 20–27
46. Yan P, Khan SM, Shah M (2008) Learning 4d action feature models for arbitrary view action recognition. In: International Conference on Computer Vision and Pattern Recognition, pp 1–7
47. Zisserman A, Maybank S (1994) A case against epipolar geometry. In: Applications of Invariance in Computer Vision, Lecture Notes in Computer Science, vol 825, Springer Berlin Heidelberg, pp 69–88