Computing Velocities and Accelerations from a Pose Time Sequence in Three-dimensional Space

Florian Sittel Jörg Müller Wolfram Burgard

Department of Computer Science, University of Freiburg, Germany
{sittelf, muellerj, burgard}@informatik.uni-freiburg.de

In recent years, small flying robots have become a popular platform in robotics due to their low cost and versatile use. In the context of autonomous navigation, low-cost robots are often equipped with imprecise sensors and actuators and require a proper calibration and carefully designed and learned models. External systems like motion capture cameras usually provide accurate pose estimates for such devices. However, they do not provide the translational and rotational velocities and accelerations of the object. In this paper, we present an algorithm for accurate calculations of the six-dimensional velocity and the six-dimensional acceleration from a possibly noisy pose time sequence. We compute the velocities and accelerations in a regression using Newton's equation of motion as the model function. Thereby, we efficiently decouple the six individual dimensions and account for fictitious forces in the non-inertial body-fixed frame of reference. In simulation and experiments with a real inertial measurement unit (IMU), we show that our algorithm provides accurate velocity and acceleration estimates compared to the reference data.

1 Introduction

Nowadays, the robotics community has an increasing interest in the third dimension. Especially small flying robots like quadrotors and miniature airships are becoming broadly available. Such robots are usually equipped with cheap sensors and can perform tasks such as environmental monitoring, surveillance, communication, and mapping [1, 3, 4, 6, 7]. However, these favorable properties come at the cost of some challenges with respect to autonomous navigation. Their low-cost and comparably simple actuators and sensors typically induce a substantial motion uncertainty and provide rather low-precision measurement data. This data can still be useful as long as the devices are carefully calibrated and modeled, which requires accurate reference data [5]. Although the poses obtained at high frequency from motion capture systems are a useful reference, often also accurate velocity and acceleration data of the robot is required for calibration and model training.

In this paper, we present an algorithm that computes the velocities and the accelerations from a sequence of poses with corresponding timestamps. We provide a set of algorithms that can be adapted and combined for different tasks in a flexible way. Our approach is hereby inherently useful, as we can efficiently calculate the regressions in each dimension individually.

Additionally, our approach treats fictitious forces in a self-consistent way, i.e., the regression itself provides the necessary data to account for the Coriolis coupling in six dimensions.

Furthermore, for the interested reader, we give a sound background on the theory behind the algorithms, including the applied constant acceleration motion model, the derivation of the regression model and the correct treatment of fictitious forces. Finally, we validate our algorithm in simulation experiments and with real data of an inertial measurement unit (IMU). We compare the velocities and accelerations computed using our algorithm with those generated by an indoor blimp simulator and the measurements of the gyroscopes and the accelerometers of an IMU.

2 Problem Formulation

In this paper, we consider the problem of estimating the time-dependent velocities and accelerations in body-fixed coordinates of a rigid body moving freely in three-dimensional space. The input data is given by an external system, e.g., by a motion capture system, in form of a trajectory of positions $\mathbf{p} = [x, y, z]^T$ and orientations $\mathbf{q} = [q_0, q_1, q_2, q_3]^T$ represented by unit quaternions [2].

The full dynamics of the device can be described by the state

$$\mathbf{x}(t) = \left[\mathbf{p}^T(t), \mathbf{q}^T(t), \mathbf{v}^T(t), \boldsymbol{\omega}^T(t), \mathbf{a}^T(t), \boldsymbol{\alpha}^T(t)\right]^T, \tag{1}$$

with the translational velocity $\mathbf{v} = [v_x, v_y, v_z]^T$, the angular velocity $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T$, the translational acceleration $\dot{\mathbf{v}} = \mathbf{a} = [a_x, a_y, a_z]^T$ and the angular acceleration $\dot{\boldsymbol{\omega}} = \boldsymbol{\alpha} = [\alpha_x, \alpha_y, \alpha_z]^T$. All velocities and accelerations are expressed in the body-fixed frame of reference. The input trajectory is given as a sequence of n poses

$$\left((\mathbf{p}_1, \mathbf{q}_1, t_1), \ldots, (\mathbf{p}_n, \mathbf{q}_n, t_n)\right) \tag{2}$$

with associated time stamps t_i. In this discrete trajectory, the positions and orientations are expressed in the global frame of reference with the z-axis pointing upwards.

We consider the problem of estimating the translational and rotational velocities and accelerations, which are corresponding to each pose of the input trajectory. The result of our algorithm enables us to formulate the input trajectory with the full state information as

$$\left((\mathbf{p}_1, \mathbf{q}_1, \mathbf{v}_1, \boldsymbol{\omega}_1, \mathbf{a}_1, \boldsymbol{\alpha}_1, t_1), \ldots, (\mathbf{p}_n, \mathbf{q}_n, \mathbf{v}_n, \boldsymbol{\omega}_n, \mathbf{a}_n, \boldsymbol{\alpha}_n, t_n)\right). \tag{3}$$

2.1 Assumptions

Assuming a certain locality in the trajectory data, we choose a small time window Δt around the target time, at which the velocity and acceleration is to be determined. If Δt is sufficiently small, we can safely assume a constant acceleration within the window and the dynamics of the object are fully described by its position, its velocities, and its accelerations in every dimension. Of course, Δt must be big enough to get a certain number of data points around the given time step to compute a reasonable regression.

3 Approach

Our approach to compute translational and rotational velocities and accelerations of an object moving in three-dimensional space given a discrete trajectory is the following:

1. Define a time window Δt large enough to span several time steps, but small in comparison to the dynamics of the trajectory.
2. Step through the trajectory and run a regression for each time window according to Algorithm 3.
   The velocity and acceleration in each single regression is calculated as follows:
   a) Separate the problem into one-dimensional regressions on each individual dimension of the translational and rotational data according to Algorithm 2.
   b) Run the one-dimensional regression on the individual dimensions to obtain the first and second derivatives in each dimension according to Algorithm 1.
   c) Transform the results (first and second derivatives) in the individual dimensions to the body-fixed frame of reference according to Algorithm 2.
   d) Account for fictitious forces in the non-inertial body-fixed frame of reference according to Algorithm 2.

For a detailed explanation and a derivation of the individual steps, the reader is referred to the Sections 3.1ff.

Algorithm 1 MotionRegression1D

Input: A sequence of values $\big((x_1, t_1), \ldots, (x_n, t_n)\big)$ with corresponding time stamps and a target time t.

Output: The first derivative $v := \dot{x}(t)$ and second derivative $a := \ddot{x}(t)$ at the target time t.

// compute various sums over regression window
$(s_x, s_{tx}, s_{t^2x}, s_t, s_{t^2}, s_{t^3}, s_{t^4}) \leftarrow (0, \ldots, 0)$
for $i = 1$ to n **do**
  $t_i \leftarrow t_i - t$
  $s_x \leftarrow s_x + x_i$
  $s_{tx} \leftarrow s_{tx} + x_i \cdot t_i$
  $s_{t^2x} \leftarrow s_{t^2x} + x_i \cdot t_i^2$
  $s_t \leftarrow s_t + t_i$
  $s_{t^2} \leftarrow s_{t^2} + t_i^2$
  $s_{t^3} \leftarrow s_{t^3} + t_i^3$
  $s_{t^4} \leftarrow s_{t^4} + t_i^4$
end for
// matrix inversion pre-factor
$A \leftarrow n\,(s_{t^3}\,s_{t^3} - s_{t^2}\,s_{t^4}) + s_t\,(s_t\,s_{t^4} - s_{t^2}\,s_{t^3}) + s_{t^2}\,(s_{t^2}\,s_{t^2} - s_t\,s_{t^3})$
// velocity at time t
$v \leftarrow A^{-1}\,(s_x\,(s_t\,s_{t^4} - s_{t^2}\,s_{t^3}) + s_{tx}\,(s_{t^2}\,s_{t^2} - n\,s_{t^4}) + s_{t^2x}\,(n\,s_{t^3} - s_t\,s_{t^2}))$
// (constant) acceleration
$a \leftarrow 2A^{-1}\,(s_x\,(s_{t^2}\,s_{t^2} - s_t\,s_{t^3}) + s_{tx}\,(n\,s_{t^3} - s_t\,s_{t^2}) + s_{t^2x}\,(s_t\,s_t - n\,s_{t^2}))$
return (v, a)

Algorithm 2 MotionRegression6D

Input: A sequence of positions and orientations $T = \big((\mathbf{p}_1, \mathbf{q}_1, t_1), \ldots, (\mathbf{p}_n, \mathbf{q}_n, t_n)\big)$ with corresponding time stamps and a target time t as well as the orientation $\mathbf{q}_t$ at time t.

Output: The translational and rotational velocities $\mathbf{v}, \boldsymbol{\omega}$ and accelerations $\mathbf{a}, \boldsymbol{\alpha}$ in the body-fixed frame of reference at the target time t.

```
(v, ω, a, α) ← (0, …, 0)
(q̇, q̈) ← (0, 0)        // initialize quaternion rates with zero in all dimensions
// run motion regressions independently in every dimension
for all i ∈ {x, y, z} do
    (v_i, a_i) ← MotionRegression1D(((p_{i,1}, t_1), …, (p_{i,n}, t_n)), t)
end for
for all i = 0 to 3 do
    (q̇_i, q̈_i) ← MotionRegression1D(((q_{i,1}, t_1), …, (q_{i,n}, t_n)), t)
end for
// Transform the derivatives to the body-fixed frame of reference.
   Here, q̄ is the adjoint of the unit quaternion q and ⊙ is the quaternion product [2].
[0; v] ← q̄_t ⊙ [0; v] ⊙ q_t
[0; a] ← q̄_t ⊙ [0; a] ⊙ q_t
[0; ω] ← 2 q̄_t ⊙ q̇
[0; α] ← 2 q̄_t ⊙ q̈
// account for fictitious forces
a ← a − ω × v
return (v, ω, a, α)
```

Algorithm 3 PoseTimeSequenceRegression

Input: The pose time sequence (trajectory) $T = \big((\mathbf{p}_1, \mathbf{q}_1, t_1), \ldots, (\mathbf{p}_n, \mathbf{q}_n, t_n)\big)$ given as positions and orientations with corresponding time stamps and the regression window Δt.

Output: The sequence of translational and rotational velocities and accelerations $\big((\mathbf{v}_1, \boldsymbol{\omega}_1, \mathbf{a}_1, \boldsymbol{\alpha}_1), \ldots, (\mathbf{v}_n, \boldsymbol{\omega}_n, \mathbf{a}_n, \boldsymbol{\alpha}_n)\big)$ corresponding to the input trajectory.

```
((v_1, ω_1, a_1, α_1), …, (v_n, ω_n, a_n, α_n)) ← ((0, …, 0), …, (0, …, 0))
for i = 1 to n do
    T' ← {(p_j, q_j, t_j) | t_j ∈ [t_i − Δt/2, t_i + Δt/2]}        // extract regression window of t_i
    (v_i, ω_i, a_i, α_i) ← MotionRegression6D(T', t_i)
end for
return ((v_1, ω_1, a_1, α_1), …, (v_n, ω_n, a_n, α_n))
```

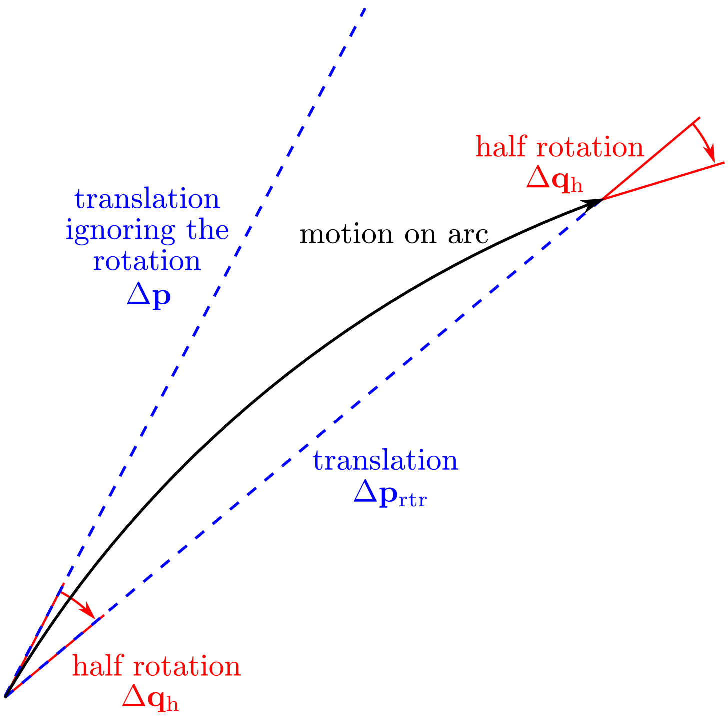


Figure 1: The "rotation-translation-rotation" scheme for a small time window Δt.

3.1 Motion Model

Given a pose time sequence describing the discrete trajectory of a rigid body moving in three-dimensional space, we calculate the velocities and accelerations in all six dimensions (three for translation and three for rotation) at a given time by regression. For this purpose, we define a small time window Δt, during which we assume a constant translational acceleration $\mathbf{a} = \dot{\mathbf{v}}$ and a constant rotational acceleration $\boldsymbol{\alpha} = \dot{\boldsymbol{\omega}}$, all in body-fixed coordinates. As the body-fixed frame of reference is not an inertial system, the motion in the different dimensions is tightly coupled. Therefore, we will introduce a straightforward motion scheme to model the true motion of the body. Afterwards, we calculate the coefficients for velocity and acceleration from the motion scheme to get appropriate correction terms for fictitious forces.

The incremental translational motion in body-fixed coordinates (ignoring the rotation during the translation) during Δt is given by

$$\Delta\mathbf{p} = \mathbf{v}\,\Delta t + \frac{1}{2}\mathbf{a}\,\Delta t^2 \tag{4}$$

as shown in Fig. 1.

The corresponding change in orientation during Δt is given by

$$\Delta\mathbf{q} = \begin{bmatrix} 1 \\ \frac{1}{2}\left(\boldsymbol{\omega}\,\Delta t + \frac{1}{2}\boldsymbol{\alpha}\,\Delta t^2\right) \end{bmatrix}, \tag{5}$$

which is the approximated quaternion built from the incremental rotation $\boldsymbol{\omega}\,\Delta t + \frac{1}{2}\boldsymbol{\alpha}\,\Delta t^2$ around all three axes (see Section 3.4). For half the time window $\Delta t/2$ we get an incremental rotation of

$$\Delta \mathbf{q}_{\mathrm{h}} = \begin{bmatrix} 1 \\ \frac{1}{2}\left(\frac{1}{2}\boldsymbol{\omega}\,\Delta t + \frac{1}{8}\boldsymbol{\alpha}\,\Delta t^2\right) \end{bmatrix} . \tag{6}$$

As long as Δt is not infinitesimally small, which is the case for a time window that contains a reasonable number of data points for regression, the translational motion ignoring the rotation $\Delta\mathbf{p}$ deviates substantially from the actual motion on an arc. According to Figure 1, we obtain a better approximation of the incremental translational motion, by approximating the motion on a curve as the three-step motion:

1. half rotation
2. translation
3. half rotation,

which we call the "rotation-translation-rotation" (rtr) scheme. In this scheme, the change in the position assuming constant acceleration is

$$\begin{bmatrix} 0 \\ \Delta\mathbf{p}_{\mathrm{rtr}} \end{bmatrix} = \Delta\mathbf{q}_{\mathrm{h}} \odot \begin{bmatrix} 0 \\ \Delta\mathbf{p} \end{bmatrix} \odot \Delta\bar{\mathbf{q}}_{\mathrm{h}} . \tag{7}$$

Theorem 1. *Let $\mathbf{v}$ and $\mathbf{a}$ be translational velocity and acceleration and $\boldsymbol{\omega}$ the rotational velocity during a time window Δt. Then the incremental motion $\Delta\mathbf{p}_{rtr}$ in* rtr *scheme is approximated by*

$$\Delta\mathbf{p}_{rtr} \approx \mathbf{v}\,\Delta t + \frac{1}{2}\left(\mathbf{a} + \boldsymbol{\omega}\times\mathbf{v}\right)\Delta t^2 \tag{8}$$

in a second-order Taylor approximation in Δt.

Proof. First, we write (7) as

$$\begin{bmatrix} 0 \\ \Delta\mathbf{p}_{\mathrm{rtr}} \end{bmatrix} = \begin{bmatrix} \Delta\mathbf{q}_{\mathrm{h},0} \\ \Delta\mathbf{q}_{\mathrm{h},1:3} \end{bmatrix} \odot \begin{bmatrix} 0 \\ \Delta\mathbf{p} \end{bmatrix} \odot \begin{bmatrix} \Delta\mathbf{q}_{\mathrm{h},0} \\ -\Delta\mathbf{q}_{\mathrm{h},1:3} \end{bmatrix} . \tag{9}$$

Then, we apply the definition of the quaternion multiplication and the quaternion adjoint (see [2]) and obtain

$$\begin{bmatrix} 0 \\ \Delta\mathbf{p}_{\mathrm{rtr}} \end{bmatrix} = \begin{bmatrix} -\Delta\mathbf{q}_{h,1:3}\cdot\Delta\mathbf{p} \\ \Delta\mathbf{p} + (\Delta\mathbf{q}_{h,1:3}\times\Delta\mathbf{p}) \end{bmatrix} \odot \begin{bmatrix} \Delta\mathbf{q}_{h,0} \\ -\Delta\mathbf{q}_{h,1:3} \end{bmatrix} \tag{10}$$

$$= \begin{bmatrix} -\Delta\mathbf{q}_{h,1:3}\cdot\Delta\mathbf{p} - (\Delta\mathbf{p} + (\Delta\mathbf{q}_{h,1:3}\times\Delta\mathbf{p}))\cdot(-\Delta\mathbf{q}_{h,1:3}) \\ (\Delta\mathbf{q}_{h,1:3}\cdot\Delta\mathbf{p})\,\Delta\mathbf{q}_{h,1:3} + \Delta\mathbf{p} + (\Delta\mathbf{q}_{h,1:3}\times\Delta\mathbf{p}) + (\Delta\mathbf{p} + (\Delta\mathbf{q}_{h,1:3}\times\Delta\mathbf{p}))\times\Delta\mathbf{q}_{h,1:3} \end{bmatrix} \tag{11}$$

$$= \begin{bmatrix} \underbrace{-\Delta\mathbf{q}_{h,1:3}\cdot\Delta\mathbf{p} + \Delta\mathbf{q}_{h,1:3}\cdot\Delta\mathbf{p}}_{0} + \underbrace{(\Delta\mathbf{q}_{h,1:3}\times\Delta\mathbf{p})\cdot\Delta\mathbf{q}_{h,1:3}}_{0} \\ (\Delta\mathbf{q}_{h,1:3}\cdot\Delta\mathbf{p})\,\Delta\mathbf{q}_{h,1:3} + \Delta\mathbf{p} + 2\,(\Delta\mathbf{q}_{h,1:3}\times\Delta\mathbf{p}) - (\Delta\mathbf{q}_{h,1:3}\times\Delta\mathbf{p})\times\Delta\mathbf{q}_{h,1:3} \end{bmatrix} . \tag{12}$$

Using Grassmann's identity and the anti-commutative behavior of cross products, we can simplify (12) to

$$\begin{bmatrix} 0 \\ \Delta\mathbf{p}_{\text{rtr}} \end{bmatrix} = \begin{bmatrix} 0 \\ 2\left(\Delta\mathbf{q}_{h,1:3} \cdot \Delta\mathbf{p}\right)\Delta\mathbf{q}_{h,1:3} + 2\left(\Delta\mathbf{q}_{h,1:3} \times \Delta\mathbf{p}\right) + \left(1 - \left|\Delta\mathbf{q}_{h,1:3}\right|^2\right)\Delta\mathbf{p} \end{bmatrix} . \tag{13}$$

We exploit the definitions (4) and (6) and split $\Delta\mathbf{p}_{\text{rtr}}$ into

$$\Delta\mathbf{p}_{\text{rtr}} = \underbrace{\Delta\mathbf{p}}_{\mathbf{v}\Delta t + \frac{1}{2}\mathbf{a}\Delta t^2} + \underbrace{2\left(\Delta\mathbf{q}_{h,1:3} \times \Delta\mathbf{p}\right)}_{\frac{1}{2}(\omega\times\mathbf{v})\Delta t^2 + \mathcal{O}(\Delta t^3)} + \underbrace{2\left(\Delta\mathbf{q}_{h,1:3} \cdot \Delta\mathbf{p}\right)\Delta\mathbf{q}_{h,1:3}}_{\mathcal{O}(\Delta t^3)} - \underbrace{\left|\Delta\mathbf{q}_{h,1:3}\right|^2\Delta\mathbf{p}}_{\mathcal{O}(\Delta t^3)} , \tag{14}$$

which is – under omission of terms $\mathcal{O}\left(\Delta t^3\right)$ – equal to

$$\Delta\mathbf{p}_{\text{rtr}} \approx \mathbf{v}\,\Delta t + \frac{1}{2}\left(\mathbf{a} + \boldsymbol{\omega} \times \mathbf{v}\right)\Delta t^2 , \tag{15}$$

and equivalent to (8). □

Since we describe the motion in body-fixed coordinates, the Taylor coefficients of zeroth order are trivially zero, whereas the coefficients of first and second order are the velocities and accelerations (including fictitious forces). Since this is nothing else than a superposition of equations of motion in three dimensions, we can separate the problem by computing the coefficients in each dimension individually, which is described in the following section.

3.2 One-Dimensional Motion Regression

In this section, we derive our regression approach to determine the first and second derivative of the one-dimensional position $x(t)$ of a moving body, i.e., its velocity and the acceleration at a given time. Assuming constant acceleration $a = \ddot{x}(t)$, the one-dimensional motion of a body can be described by its initial position x_0 and velocity $v_0 = \dot{x}(t_0)$ at time t_0 and its acceleration a by Newton's equations of motion via

$$\tilde{x}(t, a, v_0, x_0, t_0) = x_0 + v_0(t - t_0) + \frac{1}{2}a(t - t_0)^2 . \tag{16}$$

In the following we assume $t_0 = 0$ without loss of generality. Given n pairs of time and position (t_i, x_i) for $i = 1, \ldots, n$, we estimate the velocity and acceleration that minimize the squared error given by the residual

$$R(a, v_0, x_0) = \sum_{i=1}^{n} \left(x_i - \tilde{x}\left(t_i, a, v_0, x_0\right)\right)^2 . \tag{17}$$

This is the least-squares estimator for the parameters x_0, v_0, and a, and we can obtain the minimum of the residual by solving

$$\frac{\partial R}{\partial x_0} \stackrel{!}{=} \frac{\partial R}{\partial v_0} \stackrel{!}{=} \frac{\partial R}{\partial a} \stackrel{!}{=} 0 . \tag{18}$$

Putting (16) into the residual (17) results in

$$R(a, v_0, x_0) = \sum_{i=1}^{n} \left(x_i - \left(\frac{1}{2}at_i^2 + v_0 t_i + x_0\right)\right)^2 \tag{19}$$

and finally using (18), we obtain the equations

$$\sum_i x_i t_i^2 = \sum_i \left(\frac{1}{2}at_i^4 + v_0 t_i^3 + x_0 t_i^2\right) \tag{20}$$

$$\sum_i x_i t_i = \sum_i \left(\frac{1}{2}at_i^3 + v_0 t_i^2 + x_0 t_i\right) \tag{21}$$

$$\sum_i x_i = n x_0 + \sum_i \left(\frac{1}{2}at_i^2 + v_0 t_i\right) \ , \tag{22}$$

which collectively minimize the squared error. These equations form a linear system and may be rewritten in the matrix form

$$\begin{bmatrix} \sum_i t_i^2 & \sum_i t_i & n \\ \sum_i t_i^3 & \sum_i t_i^2 & \sum_i t_i \\ \sum_i t_i^4 & \sum_i t_i^3 & \sum_i t_i^2 \end{bmatrix} \begin{bmatrix} \frac{1}{2}a \\ v_0 \\ x_0 \end{bmatrix} = \begin{bmatrix} \sum_i x_i \\ \sum_i x_i t_i \\ \sum_i x_i t_i^2 \end{bmatrix} \ . \tag{23}$$

Renaming various terms according to

$$\sum_i t_i^j =: s_{t^j} \tag{24}$$

$$\sum_i x_i =: s_x \tag{25}$$

$$\sum_i x_i t_i =: s_{tx} \tag{26}$$

$$\sum_i x_i t_i^2 =: s_{t^2 x} \tag{27}$$

gives

$$\begin{bmatrix} s_{t^2} & s_t & n \\ s_{t^3} & s_{t^2} & s_t \\ s_{t^4} & s_{t^3} & s_{t^2} \end{bmatrix} \begin{bmatrix} \frac{1}{2}a \\ v_0 \\ x_0 \end{bmatrix} = \begin{bmatrix} s_x \\ s_{tx} \\ s_{t^2 x} \end{bmatrix} \tag{28}$$

$$\Rightarrow \begin{bmatrix} \frac{1}{2}a \\ v_0 \\ x_0 \end{bmatrix} = \begin{bmatrix} s_{t^2} & s_t & n \\ s_{t^3} & s_{t^2} & s_t \\ s_{t^4} & s_{t^3} & s_{t^2} \end{bmatrix}^{-1} \begin{bmatrix} s_x \\ s_{tx} \\ s_{t^2 x} \end{bmatrix} \ . \tag{29}$$

This is now readily solved for the acceleration a and the velocity v_0 by

$$a = 2A^{-1}\left(s_x\left(s_{t^2}^2 - s_t s_{t^3}\right) + s_{tx}\left(n s_{t^3} - s_t s_{t^2}\right) + s_{t^2 x}\left(s_t^2 - n s_{t^2}\right)\right) \tag{30}$$

$$v_0 = A^{-1}\left(s_x\left(s_t s_{t^4} - s_{t^2} s_{t^3}\right) + s_{tx}\left(s_{t^2}^2 - n s_{t^4}\right) + s_{t^2 x}\left(n s_{t^3} - s_t s_{t^2}\right)\right) \ , \tag{31}$$

where

$$A = n\left(s_{t^3}^2 - s_{t^2} s_{t^4}\right) + s_t\left(s_t s_{t^4} - s_{t^2} s_{t^3}\right) + s_{t^2}\left(s_{t^2}^2 - s_t s_{t^3}\right) \ . \tag{32}$$

3.3 Fictitious Forces

As one can see in the above result (8) for the coefficients of the Taylor expansion of (7), the velocity $\mathbf{v}$ is directly given by the regression, whereas the translational acceleration that we get from the regression is equivalent to a term

$$\mathbf{a}_{\text{regression}} = \mathbf{a} + \boldsymbol{\omega} \times \mathbf{v}. \tag{33}$$

This is a result of the Coriolis effect, which has an influence on the trajectory since the body does not move in an inertial frame of reference. As the Coriolis acceleration is given by

$$\mathbf{a}_{\text{Coriolis}} = -2\,\boldsymbol{\omega} \times \mathbf{v}, \tag{34}$$

the coefficients of the acceleration can be expressed as

$$\mathbf{a}_{\text{regression}} = \mathbf{a} - \frac{1}{2}\mathbf{a}_{\text{Coriolis}}\;. \tag{35}$$

The reason that we only need to subtract half of the Coriolis acceleration lies in the fact that the Coriolis effect has two different origins, each accounting for a term $-\boldsymbol{\omega} \times \mathbf{v} = \frac{1}{2}\mathbf{a}_{\text{Coriolis}}$. In general, we have to deal with the Coriolis effect when we have a motion described in a rotating frame of reference, e.g., the body-fixed coordinates we use here. In this case, one origin of fictitious force – the one that applies here – is the motion of the body in time due to the rotating frame of reference, which depends on the angular velocity $\boldsymbol{\omega}$ of the body. The second half of the Coriolis effect is a result from the change of velocity of the object inside the rotating frame of reference. This results from the different absolute velocities at different positions inside the rotating system. However, since we define the body-fixed coordinates as the frame of reference, the origin of the coordinate system is always at the center of mass of the body, i.e. the body never moves inside the rotating frame of reference. Therefore, the second term is equal to zero.

With this result, the acceleration we get from the regression can easily be separated into the real acceleration of the body and acceleration resulting from the Coriolis effect (see (35)). Rewriting this to

$$\begin{aligned} \mathbf{a} &= \mathbf{a}_{\text{regression}} + \frac{1}{2}\mathbf{a}_{\text{Coriolis}} \qquad (36)\\ &= \mathbf{a}_{\text{regression}} - \boldsymbol{\omega} \times \mathbf{v} \qquad (37) \end{aligned}$$

gives us the true body-fixed acceleration from the regression corrected for fictitious forces. Since $\boldsymbol{\omega}$ and $\mathbf{v}$ are given directly by the regression itself, this approach is self-consistent.

3.4 Quaternions from Incremental Rotations

Theorem 2. *Let $\boldsymbol{\omega}$ be the rotational velocity and $\boldsymbol{\alpha}$ the rotational acceleration during a short time interval Δt. Then, the incremental rotation during Δt is approximated by the quaternion*

$$\Delta\mathbf{q} = \begin{bmatrix} 1 \\ \frac{1}{2}\left(\boldsymbol{\omega}\,\Delta t + \frac{1}{2}\boldsymbol{\alpha}\,\Delta t^2\right) \end{bmatrix}, \tag{38}$$

assuming constant angular acceleration.

Proof. The quaternion rate from the rotational velocity

$$\dot{\mathbf{q}} = \frac{1}{2}\,\mathbf{q} \odot \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}, \tag{39}$$

and the quaternion acceleration

$$\ddot{\mathbf{q}} = \frac{1}{2}\,\mathbf{q} \odot \begin{bmatrix} 0 \\ \boldsymbol{\alpha} \end{bmatrix} \tag{40}$$

are given by Diebel [2]. Using this notation, the new orientation after Δt can be written using the incremental orientation $\Delta\mathbf{q}$ during Δt as

$$\begin{aligned} \mathbf{q} \odot \Delta\mathbf{q} &= \mathbf{q} + \dot{\mathbf{q}}\,\Delta t + \frac{1}{2}\ddot{\mathbf{q}}\,\Delta t^2 && (41) \\ &= \mathbf{q} + \frac{1}{2}\,\mathbf{q} \odot \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}\,\Delta t + \frac{1}{4}\,\mathbf{q} \odot \begin{bmatrix} 0 \\ \boldsymbol{\alpha} \end{bmatrix}\,\Delta t^2\,. && (42) \end{aligned}$$

The incremental rotation can then be written as

$$\Delta\mathbf{q} = \bar{\mathbf{q}} \odot \mathbf{q} \odot \Delta\mathbf{q} \tag{43}$$

assuming $\mathbf{q}$ is a unit quaternion and $\bar{\mathbf{q}}$ denotes its conjugate. Using (42) this can finally be simplified to

$$\begin{aligned} \Delta\mathbf{q} &= \bar{\mathbf{q}} \odot \mathbf{q} + \bar{\mathbf{q}} \odot \frac{1}{2}\,\mathbf{q}\,\begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}\,\Delta t + \bar{\mathbf{q}} \odot \frac{1}{4}\,\mathbf{q} \odot \begin{bmatrix} 0 \\ \boldsymbol{\alpha} \end{bmatrix}\,\Delta t^2 && (44) \\ &= \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}\,\Delta t + \frac{1}{4}\begin{bmatrix} 0 \\ \boldsymbol{\alpha} \end{bmatrix}\,\Delta t^2 && (45) \\ &= \begin{bmatrix} 1 \\ \frac{1}{2}\boldsymbol{\omega}\Delta t + \frac{1}{4}\boldsymbol{\alpha}\Delta t^2 \end{bmatrix}\,, && (46) \end{aligned}$$

which is equivalent to (38). □

4 Experimental Validation

4.1 Simulated data

To test our method, we generated a bulk of trajectory data by a physical simulation of a blimp flying in three-dimensional space. The data has been generated by integration of Newton's equations of motion over time with randomly chosen controls of the blimp. The output of the simulation includes trajectories, velocities and accelerations in all six dimensions with appropriate timestamps. Thereafter, we put the trajectory data with timestamps into the regression and compared the resulting velocities and accelerations with the ones given by the simulation. We present an excerpt of the plots, showing the comparisons for three-dimensional translational velocity and acceleration (see Figure 2), as well as three-dimensional rotational velocity and acceleration (see Figure 3), all with respect to the body-fixed frame of reference.

4.2 Experimental data

After the promising tests in simulation, we additionally produced a trajectory of test data by mounting an Xsens MTi IMU onto an object equipped with motion capture markers. While capturing the pose time sequence with the motion capture system, we held the object in one hand, walking through the room and exposed it to quite large accelerations in different dimensions with abrupt changes in direction. In this setting, the time window for the regression was set to 0.1 s with a temporal resolution of the motion capture system of 300 Hz.

One has to keep in mind that the algorithm above directly delivers mechanized data from a given pose time sequence. An IMU, however, "feels" the real velocities and accelerations, i.e., all fictitious and real forces affecting the system.

We therefore have to add all forces affecting the IMU to the resulting data from the regression given by

$$\mathbf{a}_{\text{reg,IMU}} = \mathbf{a}_{\text{reg}} + \underbrace{\boldsymbol{\omega} \times \mathbf{v}}_{\text{Coriolis}} + \underbrace{\boldsymbol{\alpha} \times \mathbf{p}_{\text{I}}}_{\text{rot. acc.}} + \underbrace{\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{p}_{\text{I}})}_{\text{centrifugal}} + \underbrace{\tilde{\mathbf{g}}\,(\mathbf{q})}_{\text{gravity}} \tag{47}$$

for being able to compare the acceleration measured by the accelerometer of the IMU to the acceleration calculated from regression. Here, $\mathbf{p}_{\text{I}}$ is the position of the IMU relative to the frame of reference of the object and $\tilde{\mathbf{g}}\,(\mathbf{q})$ is the orthogonal projection of the gravity of Earth $[0, 0, g]^T$ into the body-fixed frame of reference given by

$$\begin{bmatrix} 0 \\ \tilde{\mathbf{g}}\,(\mathbf{q}) \end{bmatrix} = \bar{\mathbf{q}} \odot \begin{bmatrix} 0 \\ 0 \\ 0 \\ g \end{bmatrix} \odot \mathbf{q}\,. \tag{48}$$

Since the orientation of the IMU was the same as the orientation of the object, no correction term was needed to correct for this. However, if – in another setting – the orientation of the IMU were different, translational and rotational acceleration would additionally have to be rotated by the inverse relative orientation of the IMU with respect to the body-fixed frame of reference.

Figure 4 shows the translational acceleration $\mathbf{a}_{\text{reg,IMU}}$ of our regression approach compared to the acceleration measured by the accelerometer integrated into the IMU, and Figure 5 shows the rotational velocity of our regression approach compared to the rotational velocity measured by the gyroscopes of the IMU.

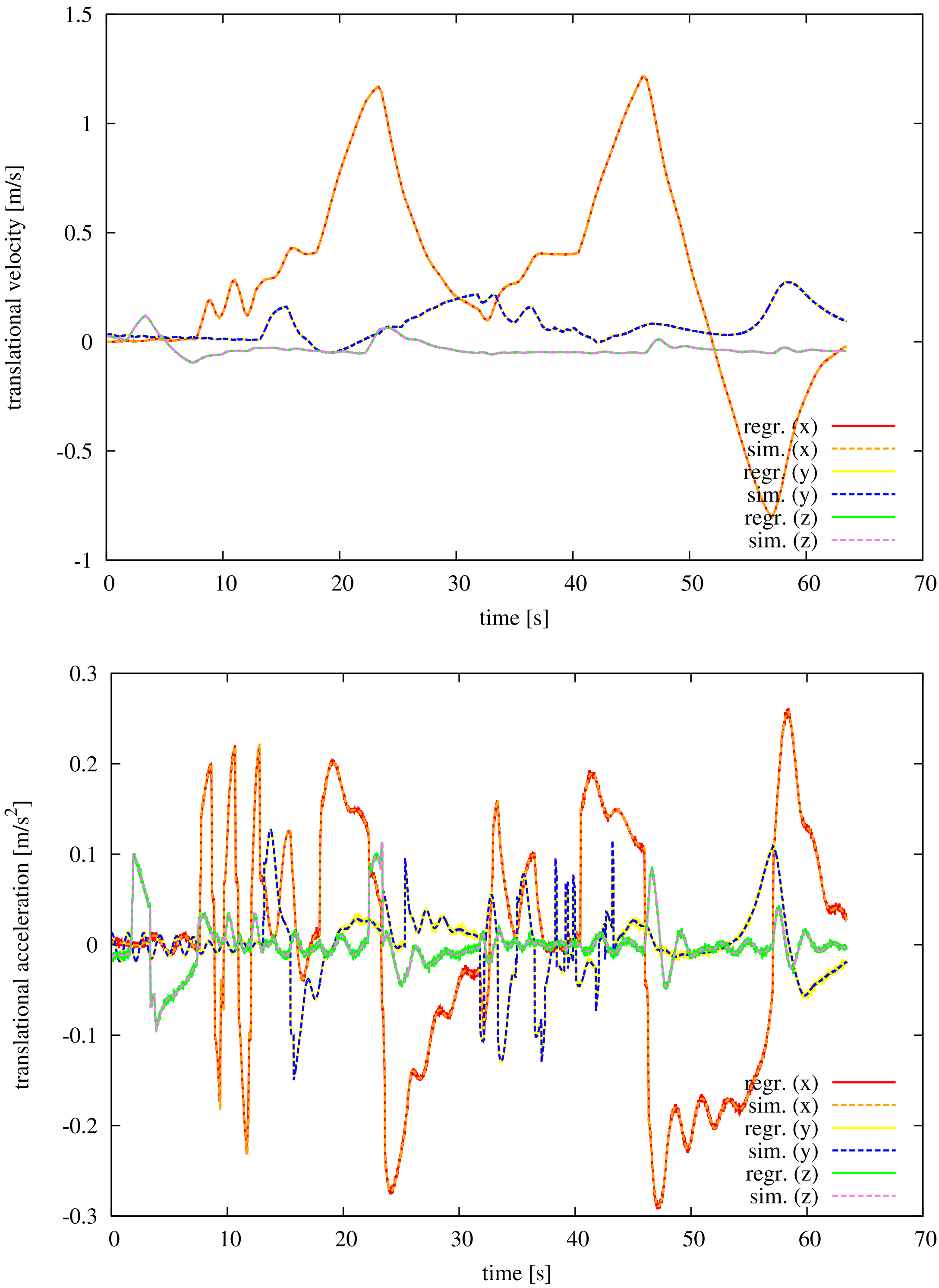
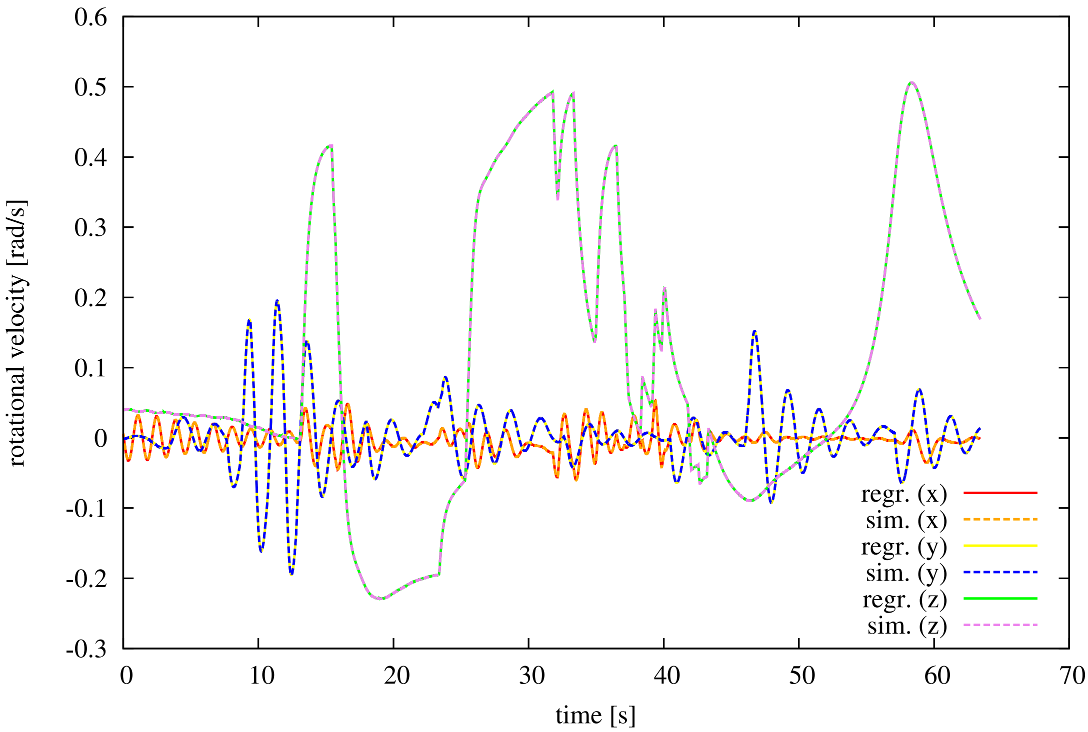


Figure 2: The translational velocities (top) and accelerations (bottom) of simulated data and the results of our regression approach on the pose time sequence generated during the simulation.



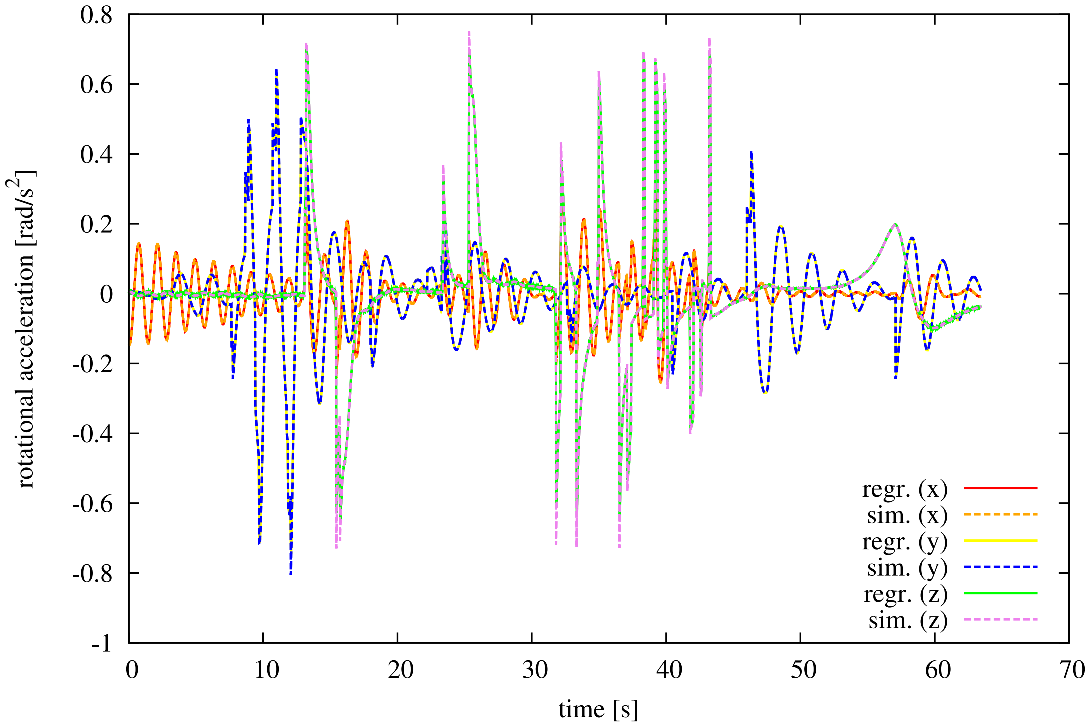


Figure 3: The rotational velocities (top) and accelerations (bottom) of simulated data and the results of our regression approach on the pose time sequence generated during the simulation.

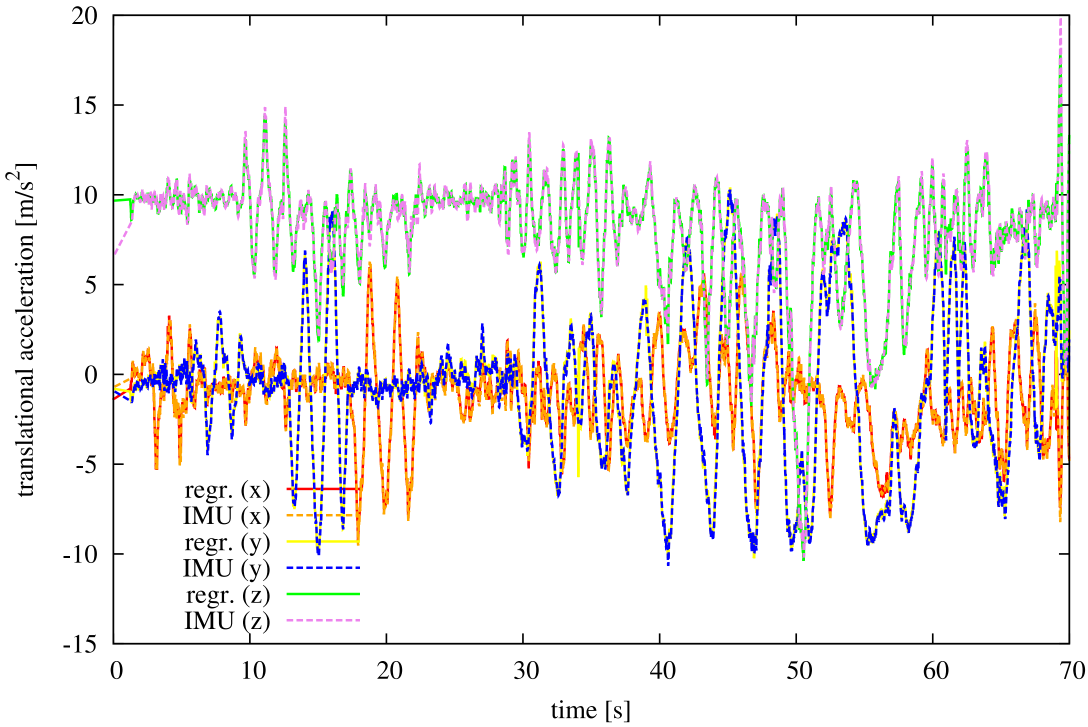


Figure 4: The translational accelerations measured by the accelerometers of an IMU compared to regression from motion capture data.

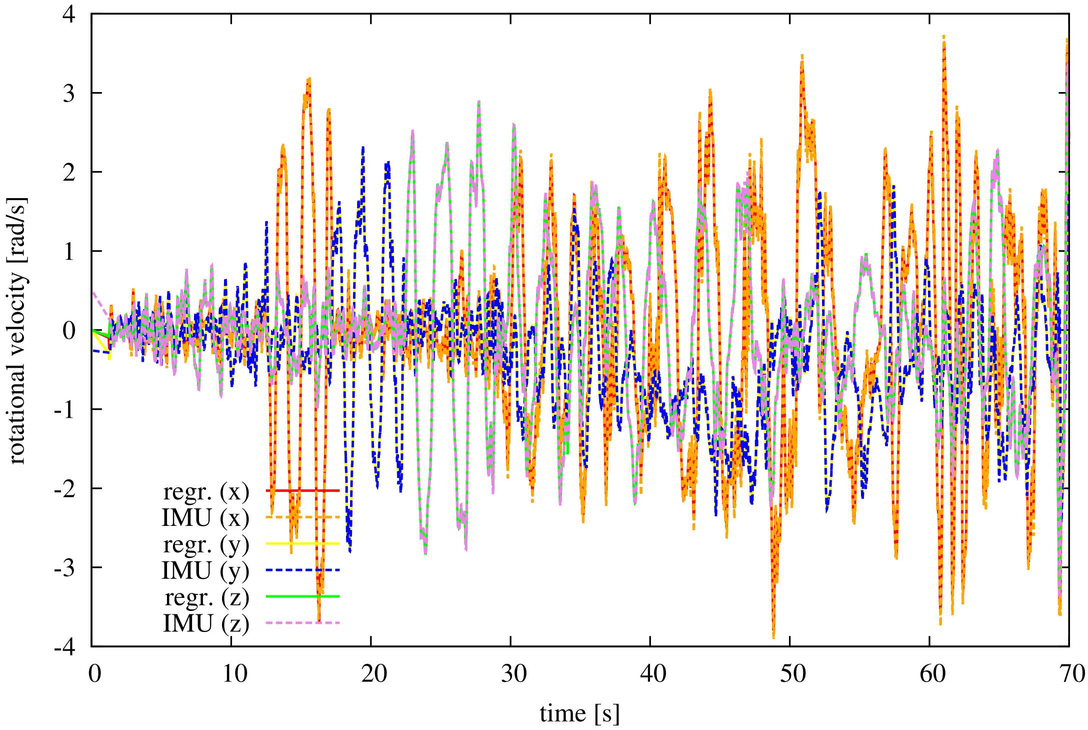


Figure 5: The rotational velocities measured by the gyroscopes of an IMU compared to regression from motion capture data.

5 Conclusions

We presented a regression-based approach to calculate the velocity and acceleration of a rigid body moving in three-dimensional space from a given pose time sequence. Our approach is especially useful for optical motion capture systems, which usually provide accurate pose estimates at high frequency for rigid bodies equipped with retroreflective markers. The experimental validation shows that our regression provides accurate velocities and accelerations taking into account fictitious forces in the moving body-fixed frame of reference.

References

[1] A. Bry, A. Bachrach, and N. Roy. State estimation for aggressive flight in GPS-denied environments using onboard sensing. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 1–8, 2012.

[2] J. Diebel. Representing attitude: Euler angles, unit quaternions, and rotation vectors. Technical report, Stanford University, 2006.

[3] J. Ko, D.J. Klein, D. Fox, and D. Haehnel. GP-UKF: Unscented kalman filters with gaussian process prediction and observation models. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007.

[4] D. Mellinger, N. Michael, and Kumar V. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *Int. Journal of Robotics Research*, 31(5):664–674, 2012.

[5] J. Müller, O. Paul, and W. Burgard. Probabilistic velocity estimation for autonomous miniature airships using thermal air flow sensors. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 39–44, 2012.

[6] J. Müller, A. Rottmann, and W. Burgard. A probabilistic sonar sensor model for robust localization of a small-size blimp in indoor environments using a particle filter. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2009.

[7] S. Weiss, D. Scaramuzza, and R. Siegwart. Monocular-SLAM-based navigation for autonomous micro helicopters in GPS-denied environments. *Journal of Field Robotics*, 28(6):854–874, 2011.