Module Interface Specification for Attitude Check

Adrian Sochaniwsky

March 17, 2024

1 Revision History

Date	Version	Notes
March 17, 20241	1.0	Initial document

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at https://github.com/adrian-soch/attitude_check/blob/main/docs/SRS/SRS.pdf.

Contents

3 Introduction

The following document details the Module Interface Specifications for Attitude Check.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at https://github.com/adrian-soch/attitude_check.

4 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol := is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | ... | c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by Attitude Check.

Data Type	**Notation**	**Description**
real	$\mathbb{R}$	any number in (-∞, ∞)
boolean	$\mathbb{B}$	value in $[\text{false} = 0, \text{true} = 1]$
matrix	$\mathbb{R}^{m \times n}$	matrix of any number in (-∞, ∞)
vector	$\mathbb{R}^m$	column vector of any number in (-∞, ∞)
quaternion	$\mathbf{q}$	a quaternion $\in \mathbb{R}^4$, see SRS for details

The specification of Attitude Check uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, Attitude Check uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**
Behaviour-Hiding Module	Estimator Module Input Module Accel To Quat Module Mag To Quat Module
Software Decision Module	Math Module Matrix Math Module Quaternion Module

Table 1: Module Hierarchy

6 MIS of Estimator Module

6.1 Module

estimator

6.2 Uses

Accel To Quat Module (Sec 7)
Mag To Quat Module (Sec 8)
Input Module (Sec 9)
Quaternion Module (Sec 10)
Matrix Math Module (Sec 11)
Math Module (Sec 12)

6.3 Syntax

6.3.1 Exported Access Programs

Name	In	Out	Exceptions
update	$q := \mathbf{q}, \text{acc} := \mathbb{R}^3, \text{gyr} := \mathbb{R}^3, \text{mag} := \mathbb{R}^3, \text{dt} := \mathbb{R}$	$q_{\text{out}} := \mathbf{q}$	ValueError
update	$q := \mathbf{q}, \text{acc} := \mathbb{R}^3, \text{gyr} := \mathbb{R}^3, \text{dt} := \mathbb{R}$	$q_{\text{out}} := \mathbf{q}$	ValueError
get_mag_norm	$q := \mathbf{q},$	${}^E\mathbf{b} := \mathbb{R}^3$	ValueError

6.4 Semantics

6.4.1 State Variables

gain : $\mathbb{R}$

6.4.2 Access Routine Semantics

update($q := \mathbf{q}, \text{acc} := \mathbb{R}^3, \text{gyr} := \mathbb{R}^3, \text{mag} := \mathbb{R}^3, \text{dt} := \mathbb{R}$):

- transition: None

- output: $q_{\text{out}} = q + \left(\dot{q}_\omega - \text{gain}\frac{\text{transpose}(J(q, {}^E\mathbf{b}))f_{g,b}(q,\text{acc}, {}^E\mathbf{b},\text{mag})}{\text{norm}(\text{transpose}(J(q, {}^E\mathbf{b}))f_{g,b}(q,\text{acc}, {}^E\mathbf{b},\text{mag}))}\right)\text{dt}$ where

$$
\begin{aligned}
\dot{q}_\omega =& \frac{1}{2}\text{quat\_prod}(q, \text{create\_quat}(0, \text{gyr})) \\
{}^E\mathbf{b} =& \text{get\_mag\_norm}(q, [m_x, m_y, m_z]) \\
[a_x, a_y, a_z] =& \frac{\text{acc}}{|\text{acc}|} \\
[m_x, m_y, m_z] =& \frac{\text{mag}}{|\text{mag}|} \\
f_{g,b} =& \begin{bmatrix} 2(q_x q_z - q_w q_y) - a_x \\ 2(q_w q_x + q_y q_z) - a_y \\ 2(\frac{1}{2} - q_x^2 - q_y^2) - a_z \\ 2b_x(\frac{1}{2} - q_y^2 - q_z^2) + 2b_z(q_x q_z - q_w q_y) - m_x \\ 2b_x(q_x q_y - q_w q_z) + 2b_z(q_w q_x + q_y q_z) - m_y \\ 2b_x(q_w q_y + q_x q_z) + 2b_z(\frac{1}{2} - q_x^2 - q_y^2) - m_z \end{bmatrix} \\
J =& \begin{bmatrix} -2q_y & 2q_z & -2q_w & 2q_x \\ 2q_x & 2q_w & 2q_z & 2q_y \\ 0 & -4q_x & -4q_y & 0 \\ -2b_z q_y & 2b_z q_z & -4b_x q_y - 2b_z q_w & -4b_x q_z + 2b_z q_x \\ -2b_x q_z + 2b_z q_x & 2b_x q_y + 2b_z q_w & 2b_x q_x + 2b_z q_z & -2b_x q_w + 2b_z q_y \\ 2b_x q_y & 2b_x q_z - 4b_z q_x & 2b_x q_w - 4b_z q_y & 2b_x q_x \end{bmatrix}
\end{aligned}
$$

- exception: ValueError

update($q := \mathbf{q}, \text{acc} := \mathbb{R}^3, \text{gyr} := \mathbb{R}^3, \text{dt} := \mathbb{R}$):

- transition: None
- output: $q_{\text{out}} = q + \left(\dot{q}_\omega - \text{gain}\frac{\text{transpose}(J(q))f_g(q,\text{acc})}{\text{norm}(\text{transpose}(J(q))f_g(q,\text{acc}))}\right)\text{dt}$ where

$$
\begin{aligned}
\dot{q}_\omega =& \frac{1}{2}\text{quat\_prod}(q, \text{create\_quat}(0, \text{gyr})) \\
[a_x, a_y, a_z] =& \frac{\text{acc}}{|\text{acc}|} \\
J =& \begin{bmatrix} -2q_y & 2q_z & -2q_w & 2q_x \\ 2q_x & 2q_w & 2q_z & 2q_y \\ 0 & -4q_x & -4q_y & 0 \end{bmatrix} \\
f_g =& \begin{bmatrix} 2(q_x q_z - q_w q_y) - a_x \\ 2(q_w q_x + q_y q_z) - a_y \\ 2(\frac{1}{2} - q_x^2 - q_y^2) - a_z \end{bmatrix}
\end{aligned}
$$

- exception: ValueError

get_mag_norm($q := \mathbf{q}, \text{mag} := \mathbb{R}^3$):

- transition: None
- output: ${}^{E}\mathbf{b} := [b_x, 0, b_z]$ where

$$\begin{aligned}
\text{m_quat} =& \text{quat_prod}(\text{create_quat}(0, m_x, m_y, m_z), \text{quat_conj}(q)) \\
b_x =& \text{norm}([\text{m_quat}_x, \text{m_quat}_y]) \\
b_z =& \text{m_quat}_z
\end{aligned}$$

- exception: ValueError

7 MIS of Accel To Quat Module

7.1 Module

acc2quat

7.2 Uses

Quaternion Module (Sec 10)
Matrix Math Module (Sec 11)
Math Module (Sec 12)

7.3 Syntax

7.3.1 Exported Access Programs

Name	In	Out	Exceptions
acc_to_quat	$\mathbf{a} := \mathbb{R}^3$	$q_{\text{out}} := \mathbf{q}$	-

7.4 Semantics

7.4.1 Access Routine Semantics

acc_to_quat($\mathbf{a} := \mathbb{R}^3$):

- output: $q_{\text{out}} :=$ euler_to_quat(θ, ϕ, ψ) given

$$\begin{aligned} \theta &= \text{atan2}(a_y, a_z) \\ \phi &= \text{atan2}(-a_x, \sqrt{a_y^2 + a_z^2}) \\ \psi &= 0 \end{aligned}$$

  where $[a_x, a_y, a_z] = \dfrac{\mathbf{a}}{|\mathbf{a}|}$

- exception: none

8 MIS of Mag To Quat Module

8.1 Module

mag2quat

8.2 Uses

Quaternion Module (Sec 10)
Matrix Math Module (Sec 11)
Math Module (Sec 12)

8.3 Syntax

8.3.1 Exported Access Programs

Name	In	Out	Exceptions
mag_to_quat	$\mathbf{m} := \mathbb{R}^3, \mathbf{a} := \mathbb{R}^3$	$q_{\text{out}} := \mathbf{q}$	ValueError

8.4 Semantics

8.4.1 Access Routine Semantics

mag_to_quat($\mathbf{m} := \mathbb{R}^3, \mathbf{a} := \mathbb{R}^3$):

- output: $q_{\text{out}} :=$ euler_to_quat(θ, ϕ, ψ) given

$$\begin{aligned}
\theta &= \text{atan2}(a_y, a_z) \\
\phi &= \text{atan2}(-a_x, \sqrt{a_y^2 + a_z^2}) \\
\psi &= \text{atan2}(m_z \sin\phi - m_y \cos\phi, m_x \cos\theta + \sin\theta(m_y \sin\phi + m_z \cos\phi))
\end{aligned}$$

  where $[a_x, a_y, a_z] = \dfrac{\mathbf{a}}{|\mathbf{a}|}$ and $[m_x, m_y, m_z] = \dfrac{\mathbf{m}}{|\mathbf{m}|}$

- exception: ValueError

9 MIS of Input Module

9.1 Module

input

9.2 Uses

None

9.3 Syntax

9.3.1 Exported Access Programs

Name	In	Out	Exceptions
input_checker	$q := \mathbf{q}, \mathrm{gyr} := \mathbb{R}^3, \mathrm{acc} := \mathbb{R}^3, \mathrm{mag} := \mathbb{R}^3, \mathrm{dt} := \mathbb{R}, \mathrm{gain} := \mathbb{R}$	y:=$\mathbb{B}$	ValueError
input_checker	$q := \mathbf{q}, \mathrm{gyr} := \mathbb{R}^3, \mathrm{acc} := \mathbb{R}^3, \mathrm{dt} := \mathbb{R}, \mathrm{gain} := \mathbb{R}$	y:=$\mathbb{B}$	ValueError

9.4 Semantics

9.4.1 Access Routine Semantics

input_checker($q := \mathbf{q}, \mathrm{gyr} := \mathbb{R}^3, \mathrm{acc} := \mathbb{R}^3, \mathrm{mag} := \mathbb{R}^3, \mathrm{dt} := \mathbb{R}, \mathrm{gain} := \mathbb{R}$):

- output: y:= true, if input values are within the bounds specified in Section 4.2.8 Input Data Constraints of the SRS.
- exception: ValueError

input_checker($q := \mathbf{q}, \mathrm{gyr} := \mathbb{R}^3, \mathrm{acc} := \mathbb{R}^3, \mathrm{dt} := \mathbb{R}, \mathrm{gain} := \mathbb{R}$):

- output: y:= true, if input values are within the bounds specified in Section 4.2.8 Input Data Constraints of the SRS.
- exception: ValueError

10 MIS of Quaternion Module

10.1 Module

quaternion

10.2 Uses

Matrix Math Module (Sec 11)
Math Module (Sec 12)

10.3 Syntax

10.3.1 Exported Access Programs

Name	In	Out	Exceptions
create_quat	w:=$\mathbb{R}$, $x := \mathbb{R}, y := \mathbb{R}, z := \mathbb{R}$	-	ValueError
quat_prod	$p := \mathbf{q}, q := \mathbf{q}$	$q_{\text{out}} := \mathbf{q}$	ValueError
normalize	-	-	ValueError
quat_conj	-	$q_{\text{conj}} := \mathbf{q}$	-
assert_is_norm	w:=$\mathbb{R}$, $x := \mathbb{R}, y := \mathbb{R}, z := \mathbb{R}$	out:= $\mathbb{B}$	ValueError
quat_to_euler	-	$\mathbf{e} := \mathbb{R}^3$	ValueError
quat_to_rot	-	$\mathbf{R} := \mathbb{R}^{3\times3}$	ValueError

10.4 Semantics

10.4.1 State Variables

quat : $\mathbf{q}$

10.4.2 Access Routine Semantics

create_quat(w, x, y, z):

- transition: quat := $\mathbf{q}$ where $\mathbf{q} = [w, x, y, z]$
- exception: ValueError when $|\text{quat}| \neq 1$

quat_prod(p, q):

- output:

$$q_{\text{out}} := \begin{bmatrix} p_w q_w - p_x q_x - p_y q_y - p_z q_z \\ p_w q_x + p_x q_w + p_y q_z - p_z q_y \\ p_w q_y - p_x q_z + p_y q_w + p_z q_x \\ p_w q_z + p_x q_y - p_y q_x + p_z q_w \end{bmatrix}$$

- exception: ValueError

quat_conj():

- output: $q_{\text{conj}} := [q_w, -q_x, -q_y, -q_z]$
- exception: -

normalize():

- transition: $\text{quat} := \left[\frac{\text{quat}_w}{d}, \frac{\text{quat}_x}{d}, \frac{\text{quat}_y}{d}, \frac{\text{quat}_z}{d}\right]$ where $d = \sqrt{\text{quat}_w^2 + \text{quat}_x^2 + \text{quat}_y^2 + \text{quat}_z^2}$
- exception: ValueError

assert_is_norm():

- output: out:= $(1 == \sqrt{w^2 + x^2 + y^2 + z^2})$
- exception: ValueError

quat_to_euler():

- output:

$$\mathbf{e} := \begin{bmatrix} \text{yaw} \\ \text{pitch} \\ \text{roll} \end{bmatrix} = \begin{bmatrix} \text{atan2}(2q_yq_w - 2q_xq_z, 1 - 2q_y^2 - 2q_z^2) \\ \text{asin}(2q_xq_y + 2q_zq_w) \\ \text{atan2}(2q_xq_w - 2q_yq_z, 1 - 2q_x^2 - 2q_z^2) \end{bmatrix}$$

  See https://www.euclideanspace.com/maths/geometry/rotations/conversions/quaternionToEuler/index.htm for 2 special cases.

- exception: ValueError

quat_to_rot():

- output:

$$\mathbf{R} := \begin{bmatrix} 1 - 2q_y^2 - 2q_z^2 & 2q_xq_y - 2q_zq_w & 2q_xq_z + 2q_yq_w \\ 2q_xq_y + 2q_zq_w & 1 - 2q_x^2 - 2q_z^2 & 2q_yq_z - 2q_xq_w \\ 2q_xq_z - 2q_yq_w & 2q_yq_z + 2q_xq_w & 1 - 2q_x^2 - 2q_y^2 \end{bmatrix}$$

- exception: ValueError

11 MIS of Matrix Math Module

11.1 Module

matrix

11.2 Uses

None

11.3 Syntax

11.3.1 Exported Access Programs

Name	In	Out	Exceptions
*	$\mathbb{R}^{m\times n} \times \mathbb{R}^{n\times m}$	$m := \mathbb{R}^{n\times n}$	ValueError
*	$\mathbb{R}^{m\times n} \times \mathbb{R}$	$m := \mathbb{R}^{m\times n}$	ValueError
+	$\mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n}$	$m := \mathbb{R}^{m\times n}$	ValueError
transpose	$\mathbb{R}^{m\times n}$	$m := \mathbb{R}^{n\times m}$	ValueError
norm	$\mathbf{x} := \mathbb{R}^n$	$\mathbf{y} := \mathbb{R}^n$	ValueError

11.4 Semantics

11.4.1 Access Routine Semantics

transpose($\mathbb{R}^{m\times n}$):

- output: $m := \mathbb{R}^{m\times n}$
- exception: ValueError

$[\mathbf{A}^T]_{i,j} = [\mathbf{A}]_{j,i}$

$\mathbb{R}^{m\times n} * \mathbb{R}^{n\times m}$:

- output: $m := \mathbb{R}^{n\times n}$
- exception: ValueError

Let $\mathbf{A} = [a_{i,j}]_{m\times n}$ and $\mathbf{B} = [b_{i,j}]_{n\times m}$. Then $\mathbf{C} = \mathbf{A}*\mathbf{B}$ with $c_{i,j} = a_{i,0}b_{0,j} + a_{i,1}b_{1,j}...a_{i,n}b_{n,j}$.

$\mathbb{R}^{m\times n} * \mathbb{R}$:

- output: $m := \mathbb{R}^{m\times n}$
- exception: ValueError

Let $\mathbf{A} = [a_{i,j}]_{m\times n}$ and $k = \mathbb{R}$. Then $\mathbf{C} = \mathbf{A} * k$ with $c_{i,j} = ka_{i,j}$.

$\mathbb{R}^{m\times n} + \mathbb{R}^{m\times n}$:

- output: $m := \mathbb{R}^{m\times n}$
- exception: ValueError

Let $\mathbf{A} = [a_{i,j}]_{m\times n}$ and $\mathbf{B} = [b_{i,j}]_{m\times n}$. Then $\mathbf{A} + \mathbf{B} = [a_{i,j} + b_{i,j}]_{m\times n}$.

norm(x:=$\mathbb{R}^n$):

- output: $y := \mathbb{R}^n$ where $y = \dfrac{\mathbf{x}}{|\mathbf{x}|}$
- exception: ValueError

12 MIS of Math Module

12.1 Module

math

12.2 Uses

None

12.3 Syntax

12.3.1 Exported Constants

$$\text{PI} := 3.141592654$$

$$\text{RAD2DEG} := \frac{180}{\text{PI}}$$

$$\text{DEG2RAD} := \frac{\text{PI}}{180}$$

12.3.2 Exported Access Programs

Name	**In**	**Out**	**Exceptions**
sin	x:=$\mathbb{R}$	out:=$\mathbb{R}$	ValueError
cos	x:=$\mathbb{R}$	out:=$\mathbb{R}$	ValueError
asin	x:=$\mathbb{R}$	out:=$\mathbb{R}$	ValueError
atan2	x:=$\mathbb{R}$, y:=$\mathbb{R}$	out:=$\mathbb{R}$	ValueError
euler_to_quat	$\mathbf{e} := \mathbb{R}^3$	$y := \mathbf{q}$	ValueError
rot_to_quat	$\mathbf{R} := \mathbb{R}^{3\times 3}$	$y := \mathbf{q}$	ValueError

12.4 Semantics

12.4.1 Access Routine Semantics

sin(x:=$\mathbb{R}$):

- output: $y = \sin(x)$
- exception: ValueError

cos(x:=$\mathbb{R}$):

- output: $y = \cos(x)$
- exception: ValueError

asin(x:=$\mathbb{R}$):

- output: $y = \sin(x)$
- exception: ValueError

atan2(y:=$\mathbb{R}$, x:=$\mathbb{R}$):

- output: See https://en.wikipedia.org/wiki/Atan2#Definition_and_computation
- exception: ValueError if $x = 0, y = 0$

euler_to_quat($\mathbf{e} := \mathbb{R}^3$):

- output: See https://www.euclideanspace.com/maths/geometry/rotations/conversions/eulerToQuaternion/index.htm
- exception: ValueError

rot_to_quat($\mathbf{R} := \mathbb{R}^{3\times3}$):

- output: See https://www.euclideanspace.com/maths/geometry/rotations/conversions/matrixToQuaternion/
- exception: ValueError

References

Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.

Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL http://citeseer.ist.psu.edu/428727.html.