Advanced Realtime Tracking

DTRACK3 Programmer's Guide

ARTTRACK, TRACKPACK, SMARTTRACK & DTRACK

Version 3.2.0
May 2023

Trademarks

The following overview shows the registered trademarks of Advanced Realtime Tracking GmbH & Co. KG:

trademarks	illustrated as	**in Germany**	**in the EU**	**in the U.S.**
A.R.T.®	*ART*	×		
ART®	*ART*	×	×	×
ARTtrack®	*ARTTRACK*	×	×	×
DTrack®	*DTRACK3*	×		
smARTtrack®	*SMARTTRACK*	×		×
ART Advanced Realtime Tracking		×	×	×
ART Advanced Realtime Tracking		×	×	×

Microsoft® and Windows® are trademarks registered in the United States and other countries by the Microsoft Corporation.

The company names and product names written in this manual are trademarks or registered trademarks of the respective companies.

Warranty regarding further use of open source software

Advanced Realtime Tracking GmbH & Co. KG (denoted as ***ART***) provides no warranty for the open source software programs contained in this product, if such programs are used in any manner other than the program execution intended by ***ART***. The licenses define the warranty, if any, from the authors or licensors of the open source software. ***ART*** specifically disclaims any warranties for defects caused by altering any open source software program or the product's configuration. You have no warranty claims against ***ART*** in the event that the open source software infringes the intellectual property rights of a third party. Technical support, if any, will only be provided for unmodified software. The full license text can be obtained from ***ART*** directly.

Disclaimer

In no event shall Advanced Realtime Tracking GmbH & Co. KG be liable for any incidental, indirect, or consequential damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use the software or hardware.

ART
Advanced Realtime Tracking

Advanced Realtime Tracking GmbH & Co. KG
Am Öferl 6
82362 Weilheim i. OB
Germany
☎ +49 (0)881-92530-00
Fax +49 (0)881-92530-01
http://ar-tracking.com

Edition June 2022

The following changes have been done:

- Output of system status data

Edition May 2021

The following changes have been done:

- Description of ***Flystick2+*** capabilities
- Input Control Data (via UDP) for ***Flystick*** feedback

Edition January 2021

The following changes have been done:

- Output of 6DOF and 3DOF covariances including calculation and inference

First Edition October 2019

Contents

Terms and definitions

term	definition
3DOF	three degrees of freedom (i.e. only position)
6DOF	six degrees of freedom (i.e. position and orientation)
5DOF	five degrees of freedom (i.e. one degree less in orientation)
base	imaginary connecting line between two cameras (e.g. the two integrated cameras inside the ***SMARTTRACK2/3/M***)
ART Controller	controller for rackmounting, calculates tracking data and generates the data output stream (compatible to ***ARTTRACK2*** , ***ARTTRACK3*** , ***ARTTRACK5*** , ***ARTTRACK5/C*** , ***ARTTRACK6/M*** , ***TRACKPACK/E***)
ART Controller/M	compact size controller, calculates tracking data and generates the data output stream (compatible to ***ARTTRACK2*** , ***ARTTRACK3*** , ***ARTTRACK5*** , ***ARTTRACK5/C*** , ***ARTTRACK6/M*** , ***TRACKPACK/E***)
ARTTRACK Controller (discontinued)	calculates tracking data and generates the data output stream (compatible to ***ARTTRACK1*** , ***ARTTRACK2*** , ***ARTTRACK3***)
ARTTRACK2 (discontinued)	infrared camera
ARTTRACK3 (discontinued)	infrared camera
ARTTRACK5	infrared camera
ARTTRACK5/C	infrared camera dedicated for multi-sided projections
ARTTRACK6/M	infrared camera dedicated for compact environments or multi-sided projections
ARTTRACK6/M for Active Markers	flashless camera dedicated for use with active targets in compact environments or multi-sided projections
body calibration	teach the system the geometry of a rigid body
body, rigid body	rigid arrangement of multiple markers intended for 6DOF tracking (see also "target")
calibration angle	belongs to the room calibration set and defines origin and orientation of the room coordinate system
ceiling suspension	equipment to mount an infrared camera or ***SMARTTRACK2/3/M*** to the ceiling
DTRACK3	
backend software	software running on the ***ART Controller*** , ***ART Controller/M*** or ***SMARTTRACK2/3/M*** doing all necessary calculations
frontend software	graphical user interface running on customer's computer to control the ***ART Controller*** , ***ART Controller/M*** or ***SMARTTRACK2/3/M***
Field of View (FoV)	is the area of interest captured on the camera's image
finger thimble	a fixture for the finger tip to hold the active marker(s) (not available for ***SMARTTRACK2/3/M***)
Fingertracking (discontinued)	tracks the orientation of the hand and the position of the fingers (not available for ***SMARTTRACK2/3/M***)
FINGERTRACKING2	tracks the orientation of the hand and the position of the fingers (not available for ***SMARTTRACK2/3/M***)
FINGERTRACKING2 Tactile	tracks the orientation of the hand and the position of the fingers and provides Tactile Feedback
Flystick2/2+/3	wireless interaction device for virtual reality (VR) applications
Flystick feedback	vibrational or acoustic signal that can be triggered in a capable ***Flystick***
hand geometry	describes the dimensions of your hand and fingers (not available for ***SMARTTRACK2/3/M***)

term	definition
hybrid tracking	sensor fusion of optical and inertial data into one consolidated output
inertial sensor	an inertial measurement unit simultaneously measures 9 physical properties, i.e. angular rates, linear accelerations (unused) and magnetic field components (unused) along all 3 axes. This is achieved using solid state gyroscopes for measurement of roll, pitch and yaw and optical tracking for drift correction.
infrared optical tracking	position measurement of bodies (subjects or objects) based upon infrared light and optical measurement procedures
license code (license key)	software key to unlock certain capabilities of the tracking system or ***SMARTTRACK2/3/M***
marker	object either made of retro reflective material or LED for position tracking (3DOF)
Measurement Tool	a pointing device which allows to measure the position of the tool's tip with high accuracy
measurement volume	defines the volume where optical tracking is possible
modulated flash	infrared signal which is used for wireless synchronization
motion capture	track movements of a human body
mutual blinding	at least one camera (also from ***SMARTTRACK2/3/M***) sees disturbing reflections caused by the infrared flashes of another camera (or ***SMARTTRACK3***)
prediction	predicts output for the specified time in the future to compensate tracking and rendering latency
Radio Transceiver	exchange data with ***Flystick2/2+/3*** (integrated in ***SMARTTRACK2***)
RadioTransceiver2/3	uses USB port to exchange data with ***Flystick2/2+/3*** or ***FINGERTRACKING2 Tactile***
room calibration	teach the system the position of each camera and define origin and orientation of the room coordinate system
room calibration set	consists of angle tool and wand
SMARTTRACK3	fully integrated stand-alone infrared optical tracking system with two cameras and integrated controller, calculates tracking data and generates the data output stream
SMARTTRACK3/M	compact version of ***SMARTTRACK3***
syncgroup	cameras being in one syncgroup receive the sync signal at the same time. Syncgroups are distinguished by a short time delay between their sync signals in order to avoid mutual blinding. (not available for ***SMARTTRACK2***)
Synccard	unit integrated in ***SMARTTRACK2/3/M*** which serves for synchronizing the cameras
Synccard2	board integrated in the ***ARTTRACK*** Controller (discontinued) which serves for synchronizing the cameras
Synccard3	board integrated in the ***ART Controller*** or ***ART Controller/M*** which serves for synchronizing the cameras
SynccardTP	board integrated in the ***TRACKPACK*** Controller (discontinued) which serves for synchronizing the cameras
Tactile Feedback	system for finger-based interactions in immersive virtual reality applications (wires touch the inside of the finger tips and provide an impression when they are shortened)
target	rigid arrangement of multiple markers intended for 6DOF tracking (see also "rigid body")
tracking	position measurement of bodies that move in a defined space
TRACKPACK (discontinued)	infrared camera
TRACKPACK/C (discontinued)	infrared camera dedicated for multi-sided projections

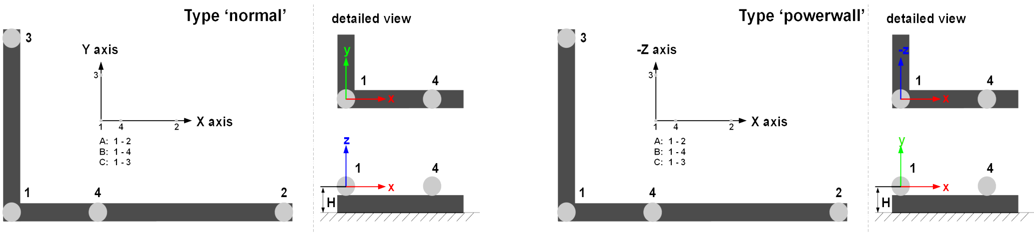
term	definition
TRACKPACK/E	infrared camera
TRACKPACK Controller (discontinued)	calculates tracking data and generates the data output stream (compatible to ***TRACKPACK*** and ***TRACKPACK/C***)
virtual point cloud	used for calculating the relative position of cameras or ***SMART-TRACK2/3/M*** that cover the measurement volume
wand	precalibrated stick carrying two markers. The wand belongs to the room calibration set and is used to generate a virtual point cloud and to scale the measurement volume

1 Definition of Coordinates and Rotations

1.1 Calibration of Room Coordinate System

The calibration angle tool defines origin and axes of the room coordinate system. This can be done in two different ways (see also figure 1.1 on page 10:

Type	longer arm	shorter arm
'*normal*'	+X axis	+Y axis
'*powerwall*'	+X axis	-Z axis



(a) type 'normal' (b) type 'powerwall'

Figure 1.1: Definition of room coordinate systems

A room calibration of type '*normal*' would result in a coordinate system like the following:

1. The marker located in the intersection of the two arms is defining the origin of the coordinate system (Marker #1).
2. The longer arm of the calibration tool defines the +X axis (Marker #1 $\leftrightarrow$ Marker #2).
3. The shorter arm of the calibration tool defines the +Y axis. (Marker #1 $\leftrightarrow$ Marker #3, i.e., the tool arms define the X/Y plane.)
4. The Z axis is added in order to give a right-handed coordinate system.

1.1.1 Room Adjustment

DTRACK3 allows to modify the room coordinate system by specifying these seven values:

- l_x, l_y, l_z for a translational offset (denoted x, y, z in the GUI),

- η, θ, ϕ for a rotation (denoted rx, ry, rz in the GUI).
- s for a scaling factor.

The offsets are defined as a shift while the rotation and/or scaling of the room coordinate system relative to the original one. Mathematically a point $\vec{x}_{orig}$ in the original room is transformed into a point $\vec{x}_{mod}$ in the modified room coordinate system by:

$$\vec{x}_{mod} = (R^T \cdot \vec{x}_{orig} - \vec{l}) \cdot s$$

where the rotation matrix R is calculated from η, θ and ϕ like defined in section 1.3 on page 13.

1.2 Calibration of Body Coordinate System

During the body calibration ***DTRACK3*** is fixing a local coordinate system (i.e. the (body coordinate system) for each rigid body. Both coordinate systems define the later 6DOF output (see chapter 1.3 on page 13). The calibration can be done in three different ways (to be selected in the menu *Tracking* → *Body Calibration*):

1.2.1 Definition of the Coordinates by the Body itself

Body calibration setting *due to body*:

The body coordinate system is fixed by the markers of the rigid body according to a set of rules:

1. Search the biggest distance between two markers of the rigid body. These two markers (#1 and #2) will define the X axis.
2. Search for a third marker (#3) that has the smallest distance to one of the two markers #1 and #2. The marker that has smallest distance to marker #3 becomes marker #1. It will define the coordinate origin. The other marker will be #2. The positive X axis is directed from marker #1 to marker #2.
3. Marker #3 defines the X/Y plane, together with markers #1 and #2. Marker #3 has a positive Y coordinate.
4. The Z axis is already defined by these rules, resulting in a right-handed coordinate system.

1.2.2 Definition of the Coordinates by the Room Coordinate System, with Origin in the Center of the Markers

Body calibration setting *due to room*:

The origin of the body coordinate system is set to the center of gravity (COG) of all mark-

ers building the rigid body. The axes of the body coordinate system are parallel to the axes of the room coordinate system in the beginning of the body calibration.
This means, the result of a body calibration *due to room* will depend on the angular position of the target during calibration. A 6DOF measurement, following calibration without having moved the body, will give the angular coordinates 0° / 0° / 0°.
If the target was moved during calibration, the angular position of the target at the beginning of the calibration will be taken.

1.2.3 Definition of the Coordinates by the Room Coordinate System, with Origin in a Marker

Body calibration setting *due to room (zero in marker)*:

This is a combination of the first two methods. The direction of the axes of the body coordinate system will be set parallel to the room coordinate system in the moment of body calibration - like done with setting *due to room*. The origin of the body coordinate system is given by one marker of the body, according to the rules given for setting *due to body*.

1.2.4 Coordinate System Definition for 5DOF Targets (with and without cylinder markers)

Body calibration setting *x*/*y*/*z*:

In the body coordinate system all markers of the target are on the selected axis. The origin is in the middle between the two markers with the largest distance to each other. The orientation is defined by the marker with the smallest distance to the origin. Its position has a negative sign. The other two directions are undetermined due to the one degree of freedom.

1.2.5 Coordinate System Definition for two 5DOF Targets with cylinder markers

Body calibration setting *xy*/*yx*/*yz*/*zy*/*zx*/*xz*:

The body is expected to consist of two about perpendicularly connected 5DOF targets. These are placed on the two axes. The origin is placed at the position where the two 5DOF targets intersect. The first axis is assigned to the 5DOF target which includes the marker with the largest distance to the origin. The other 5DOF target is placed in the plane created by the two axes.

1.3 6DOF Results

1.3.1 Position and Orientation

Position and orientation of a target are expressed by an affine transformation ($\vec{s}$, R) that transforms a vector $\vec{x}$ from the body coordinate system to the room coordinate system:

$$\vec{x}_{room} = R \cdot \vec{x}_{body} + \vec{s}$$

This means, the coordinates $\vec{s}$ give the position of the origin of the body coordinate system (marker #1 or center of gravity, as described above) measured in room coordinates.
The 3×3 rotation matrix R describes the rotation part of the transformation. The columns of the matrix R are the axes (X, Y, Z) of the body coordinate system expressed in room coordinates.

1.3.2 Description by Rotation Angles

The rotation matrix can be replaced by three consecutive rotations $R_i(\chi)$ (rotation angle χ, rotation axis i). The angles, as given in the ***DTRACK3*** data output, are defined by the equation:

$$R = R_x(\eta) \cdot R_y(\theta) \cdot R_z(\phi)$$

Expressed in trigonometric functions, this means:

$$R = \begin{pmatrix} \cos\phi\cos\theta & -\sin\phi\cos\theta & \sin\theta \\ \sin\phi\cos\eta + \cos\phi\sin\theta\sin\eta & \cos\phi\cos\eta - \sin\phi\sin\theta\sin\eta & -\cos\theta\sin\eta \\ \sin\phi\sin\eta - \cos\phi\sin\theta\cos\eta & \cos\phi\sin\eta + \sin\phi\sin\theta\cos\eta & \cos\theta\cos\eta \end{pmatrix}$$

Note that per definitionem the angles can only have the values:

$$-180^\circ \leq \phi \leq 180^\circ, -90^\circ \leq \theta \leq 90^\circ, -180^\circ \leq \eta \leq 180^\circ$$

When connecting *DTRACK3* to an application, problems may occur due to the different definitions of rotation angles. To avoid this, we recommend to use rotation matrices.

Note: For orientations close to $\theta = \pm 90^\circ$ the other two axes are driven into a parallel configuration with the system effectively losing one degree of freedom. In this situation the rotation angles will show strange behaviour, e.g. large odd-looking changes. This is also known as the so called "Gimbal Lock".

1.4 6DOF Covariance

DTRACK3 provides an optional estimation of uncertainty for the measured 6DOF results, expressed as a 6×6 covariance matrix. The definition of the 6DOF results is expanded by the following error representation:

$$\vec{x}_{room} = E_R \cdot (R \cdot \vec{x}_{body} + \vec{s} - \vec{c}) + \vec{c} + \vec{e}_T$$

The uncertainty is expressed by the translational vector $\vec{e}_T$ and the rotation matrix E_R in respect to the so-called center of rotational error (CRE) $\vec{c}$. ***DTRACK3*** is determining $\vec{c}$ for every single frame so that the rotational error becomes minimal (so-called point of minimum error). So CRE may vary and is dependent e.g. on the currently tracked markers on the body; it typically lies near the body's center of gravity.

The 6×6 covariance matrix refers to the error vector $\vec{e} = (e_x, e_y, e_z, e_\eta, e_\theta, e_\phi)$, which is constituted by the rotation matrix E_R with the three angles e_η, e_θ and e_ϕ representing rotations around the world coordinate axes and the translational vector (e_x, e_y, e_z). The covariance matrix Σ is calculated as the 'expected values' ($E()$) of:

$$\sigma_{ij} = E(e_i \cdot e_j)$$

Intuitively, the diagonal entries of Σ can be interpreted as the positional and rotational uncertainty in world coordinates. Off-diagonal values describe the statistical dependency between two values.

For a more detailed description of the error model, the mathematical treatment and the application of the covariance matrix please refer to the following publication:

Pustka D., Willneff J., Wenisch O., Lükewille P., Achatz K., Keitler P., Klinker G. (2010). Determining the Point of Minimum Error for 6DOF Pose Uncertainty Representation. 9th IEEE International Symposium on Mixed and Augmented Reality 2010: Science and Technology, ISMAR 2010 - Proceedings. 37 - 45. 10.1109/ISMAR.2010.5643548.

1.5 3DOF Data

Besides the tracking of 6DOF bodies, ***DTRACK3*** is able to calculate the coordinates of single markers, i.e. markers that can not be recognized as part of a rigid body. The output values are the coordinates of these markers measured in room coordinates.
In some situations a rigid body within the measurement volume may (temporarily) not be recognized correctly by the software. In these cases its markers appear as 3DOF objects. 3DOF markers are tracked (as long as possible) and labeled with an ID number. When a 3DOF marker vanishes (or is recognized as part of a 6DOF body) its ID number will not be used any more, as long as the tracking is active.

1.6 3DOF Covariance

DTRACK3 provides an optional estimation of uncertainty for the measured 3DOF results, expressed as a 3×3 covariance matrix Σ . Assuming a translational vector $\vec{e} = (e_x, e_y, e_z)$

representing the error of the measured 3DOF position, the covariance is calculated as the 'expected values' ($E()$) of:

$$\sigma_{ij} = E(e_i \cdot e_j)$$

Intuitively, the diagonal entries of Σ can be interpreted as the positional uncertainty in world coordinates. Off-diagonal values describe the statistical dependency between two values.

Chapter 1

1.7 Measurement Tools Coordinate System

Only available if the Measurement Tool license is present for *DTRACK3*

The Measurement Tool license allows the use of Measurement Tools , i.e. pointing devices with a special target geometry. While tracking, the module calculates position and orientation of the tip of the tool. It is necessary to perform an additional calibration procedure (so called tip calibration) to provide the module with information about the tip.

1.7.1 Orientation of a Measurement Tool

The module modifies the local coordinate system (i.e. the measurement tool coordinate system) of the tool's body as follows:

1. The tip becomes the origin of the coordinate system.
2. The marker with the largest distance to the tip defines the +Z axis.
3. The marker, that is closest to the tip, defines the Y/Z plane.

This definition shall ease the use of the tip orientation. For instance, the orientation of all ***ART*** Measurement Tools is approximately along the -Z axis.

1.7.2 Using a Reference Body

When using a reference body for a Measurement Tool , the module calculates the position of the tip $\vec{x}^S_{ref}$ within the local coordinate system of the reference body:

$$\vec{x}^S_{room} = R_{ref} \cdot \vec{x}^S_{ref} + \vec{s}_{ref}$$

where $\vec{x}^S_{room}$ is the position of the tip in room coordinates, and $(\vec{s}_{ref}, R_{ref})$ position and orientation (see 1.3 on page 13) of the reference body. The orientation of the Measurement Tool is transformed in an analogous way.

1.7.3 Covariance of Measurement Tool tip

DTRACK3 provides an estimation for the current uncertainty of the measured position of the tool tip, expressed as a 3×3 covariance matrix Σ . Assuming a translational vector $\vec{e}^S$ representing the error of the measured tip position, the covariances are calculated as the 'expected values' ($E()$) of:

$$\vec{x}^S_{meas} = \vec{x}^S + \vec{e}^S$$
$$\sigma_{ij} = E(e^S_i \cdot e^S_j)$$

Intuitively, the diagonal entries of Σ_{tip} can be interpreted as the positional uncertainty in world coordinates. Off-diagonal values describe the statistical dependency between two values.

1.8 Fingertracking Coordinate System

The ***Fingertracking*** devices allow to track a human's entire hand including fingers. Given the length of each phalanx and the angles between them, it is possible to reconstruct the entire finger.

1.8.1 Hand Coordinate System

The hand coordinate system is defined as follows (refer to figure 1.2 on page 17 and figure 1.3 on page 17 for left and right hand respectively):

- the origin is in the joint, where the index finger is connected to the back of the hand.
- the +X axis is oriented in direction of the outstretched index finger.
- the +Y axis is defined parallel to the back of the hand, indicating towards the ring finger.
- the +Z axis is added in order to give a right-handed coordinate system.

☞ **Note: The +Z axis points up upwards with respect to the back of the hand for a left hand but downwards for a right hand!**

1.8.2 Finger Coordinate System

The finger coordinate system (used to measure the orientation of the outermost phalanx relative to the hand coordinate system) is defined by these rules:

- the origin is the center of the (partial) sphere that forms the finger tip,
- the +X axis is oriented in direction of the outstretched finger,

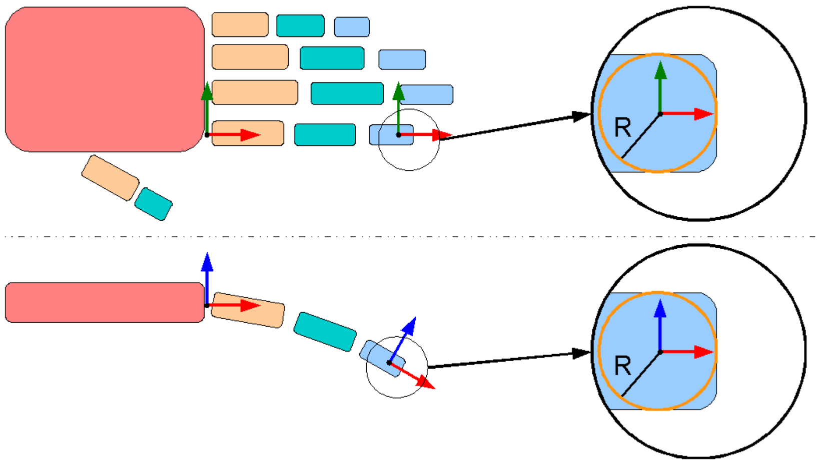


Figure 1.2: Model of a human left hand

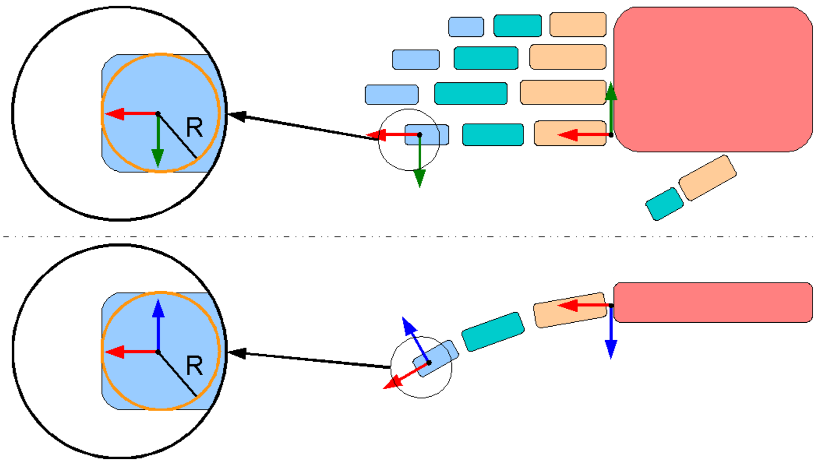


Figure 1.3: Model of a human right hand

- the +Z axis is defined to form a normal on the finger nail pointing upwards,
- the +Y axis is added in order to give a right-handed coordinate system.

☞ **Note: The joint between the innermost phalanxes and the back of the hand can move! This corresponds to a bending of the hand's back.**

2 Flystick Output Codes

DTRACK3 is supporting the following input devices:

- ***Flystick2*** and ***Flystick2+***
- ***Flystick3***

Each of the these devices is equipped with buttons (4 - 8) and a small joystick. Input transactions are transmitted via low-rate wireless personal area network (IEEE 802.15.4) to the controller and added to the 6DOF tracking result of the ***Flystick2/2+/3*** bodies.
There are two types of output formats available, called `6df` and `6df2` (refer to chapters 3.10 on page 29 and 3.9 on page 28 respectively); they differ in the number of carried input controls. The `6df2` format is transferring floating point values (for controllers, e.g. the joystick). Both formats use the same order of buttons (details see below):

Output Format	Number of Buttons	Order of Buttons	Number of Controllers
`6df`	8 (fix)	"right to left"	—
`6df2`	device dependent	"right to left"	device dependent

Table 2.1: Flystick output formats

☞ **Format `6df` is outdated and supported just for backward compatibility. It is recommended to use format `6df2` whenever possible.**

☞ **When using the output format `6df`, the joystick values are transferred into hat switch actions; as a result some of the buttons might not be accessible.**

2.1 Flystick2 Output Codes

Each ***Flystick2*** is equipped with six buttons and a small joystick that produces two analogue values, one for horizontal and one for vertical movement (see table 2.2).

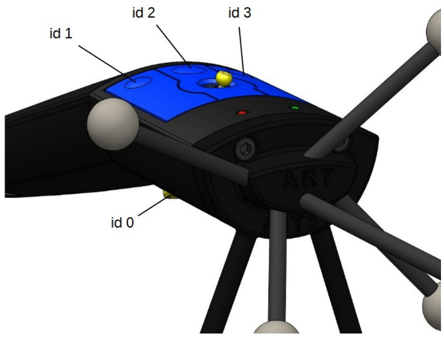
2.2 Flystick2+ Output and Feedback Codes

Each ***Flystick2+*** is equipped with eight buttons and a small joystick that produces two analogue values, one for horizontal and one for vertical movement. Additionally the trigger button produces a third analogue value (see table 2.3 on page 20).

Button	labelled as (in figure 2.1)	`6df` Output	`6df2` Output
front trigger (yellow)	id 0	code 01 (hex)	button code 01 (hex)
outer right button on top (blue)	id 1	code 02 (hex)	button code 02 (hex)
inner right button on top (blue)	id 2	code 04 (hex)	button code 04 (hex)
inner left button on top (blue)	id 3	code 08 (hex)	button code 08 (hex)
outer left button on top (blue)	id 4	–	button code 10 (hex)
button press on joystick (yellow)	id 5	–	button code 20 (hex)
joystick right (yellow)	–	code 80 (hex)	first controller up to 1.0
joystick left (yellow)	–	code 20 (hex)	first controller up to -1.0
joystick up (yellow)	–	code 40 (hex)	second controller up to 1.0
joystick down (yellow)	–	code 10 (hex)	second controller up to -1.0

Table 2.2: Flystick2 output codes

(a) ***Flystick2***



(b) ***Flystick3***

Figure 2.1: Allocation of ID numbers to ***Flystick2*** and ***Flystick3*** buttons

Flystick2+ is capable to produce an acoustic (beep) and vibrational feedback. Six vibrational patterns are supported (see table 2.4 on page 21).

2.3 Flystick3 Output Codes

Each ***Flystick3*** is equipped with four buttons and a small joystick that produces two analogue values, one for horizontal and one for vertical movement (see table 2.5).

Button	labelled as (in figure 2.2)	`6df` Output	`6df2` Output
front trigger (dark gray)	id 0	code 01 (hex)	button code 01 (hex), third controller up to 1.0
outer right button on top (red)	id 1	code 02 (hex)	button code 02 (hex)
inner right button on top (black)	id 2	code 04 (hex)	button code 04 (hex)
inner left button on top (blue)	id 3	code 08 (hex)	button code 08 (hex)
outer left button on top (gray)	id 4	–	button code 10 (hex)
button press on joystick (dark gray)	id 5	–	button code 20 (hex)
upper left button on top (white)	id 6	–	button code 40 (hex)
upper right button on top (white)	id 7	–	button code 80 (hex)
joystick right (dark gray)	–	code 80 (hex)	first controller up to 1.0
joystick left (dark gray)	–	code 20 (hex)	first controller up to -1.0
joystick up (dark gray)	–	code 40 (hex)	second controller up to 1.0
joystick down (dark gray)	–	code 10 (hex)	second controller up to -1.0

Table 2.3: Flystick2+ output codes

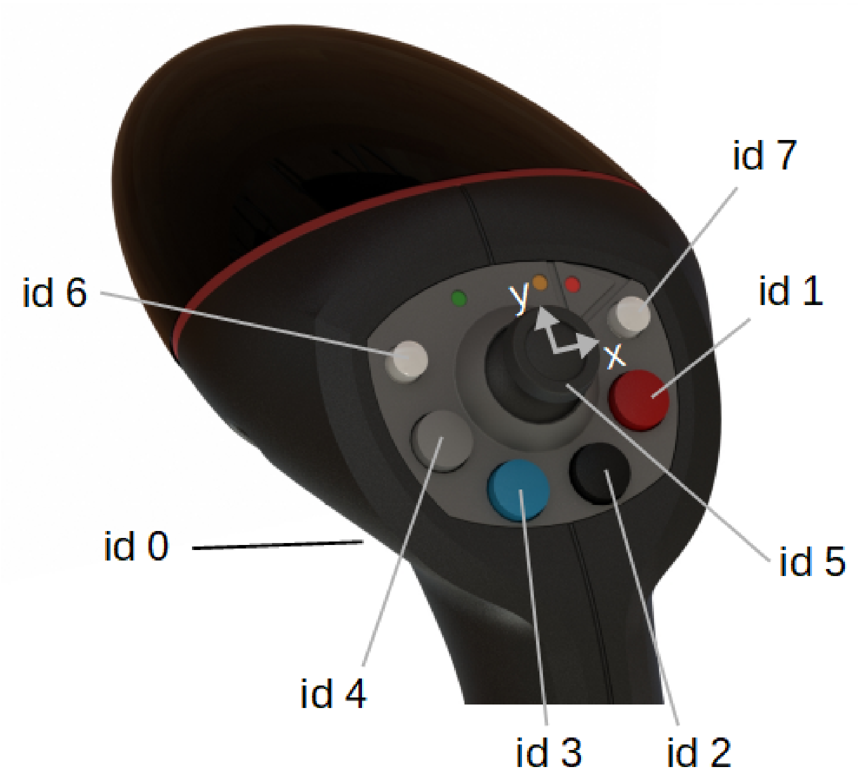


Figure 2.2: Allocation of ID numbers to ***Flystick2+*** buttons

Vibration pattern ID	Feedback
1	single click
2	double click
3	bump
4	transition hum
5	ramp up
6	ramp down

Table 2.4: Flystick2+ vibration pattern

Button	labelled as (in figure 2.1)	`6df` Output	`6df2` Output
bottom trigger (yellow)	id 0	code 01 (hex)	button code 01 (hex)
right button on top (blue)	id 1	code 02 (hex)	button code 02 (hex)
middle button on top (blue)	id 2	code 04 (hex)	button code 04 (hex)
left button on top (blue)	id 3	code 08 (hex)	button code 08 (hex)
joystick right (yellow)	–	code 80 (hex)	first controller up to 1.0
joystick left (yellow	–	code 20 (hex)	first controller up to -1.0
joystick up (yellow)	–	code 40 (hex)	second controller up to 1.0
joystick down (yellow)	–	code 10 (hex)	second controller up to -1.0

Table 2.5: Flystick3 output codes

3 Output of Measurement Data via Ethernet

DTRACK3 uses ethernet (UDP/IP datagrams) to send measurement data to other applications. Each UDP datagram carries all the results of a single measurement coded in ASCII format. It contains several lines separated by CR/LF (hex `0D 0A`). Each line carries data of a specific type and starts with an identifier.
Refer to table 3.1 for all lines or data types that may be included in the output.
A datagram is sent after each measurement of the cameras has been processed according to the cameras synchronization frequency. ***DTRACK3*** provides the ability to decrease the data output frequency by a divisor ($f_{output} = f_{sync}/d_{divisor}$).

***ART* provides free sample source code ('DTrackSDK', available in C++ and Java) to receive and parse the output data. Please contact *ART* to obtain it.**

Please refer to the *DTRACK3* User's-Guide (*Settings* → *Output*) how to configure the IP address and the port of the application (as well as the computer it runs on). You can also set the divisor $d_{divisor}$ there.

All data are given in units millimeter (`mm`) or angular degree (`deg` / °) except 6D covariance ((`rad`

3.1 Frame Counter

Identifier `fr`.

This line is always the first one, it cannot be deactivated. It carries the frame counter (counting at the data output frequency ($f_{output} = f_{sync}/d_{divisor}$).

Example:

```
fr 21753
```

3.2 Timestamp

Identifier `ts`.

A timestamp can be added to each datagram. It shows the time at the measurement of this frame, i.e. the time when the infrared flash of the cameras is triggered. The timestamp uses the internal clock of the controller giving back the seconds since 00:00

Identifier	Description	Type of data	enable/disable
`fr`	frame counter		always enabled
`ts`	timestamp		yes ('*ts*')
`6dcal`	number of calibrated bodies	additional informations	yes ('*6dcal*')
`6d`	6DOF standard body	6DOF	yes ('*6d*')
`6di`	6DOF inertial body	6DOF + inertial status	yes ('*6di*')
`6dcov`	6DOF covariance	6DOF covariance matrix	yes ('*6dcov*', optional, hidden by default) [1]
`3d`	3DOF markers	3DOF	yes ('*3d*')
`3dcov`	3DOF covariance	3DOF covariance matrix	yes ('*3dcov*', optional, hidden by default) [1]
`6df/6df2`	Flystick	6DOF + buttons	yes, alternatively ('*6df / 6df2*')
`6dmt2`	Measurement Tool	6DOF + tip trafo + sphere radius	yes ('*6dmt2*')
`6dmt`	Measurement Tool (Old)	6DOF + tip trafo	yes ('*6dmt*')
`6dmtr`	Measurement Tool reference	6DOF	yes ('*6dmtr*')
`gl`	Fingertracking hand	6DOF + fingers	yes ('*gl*')
`glcal`	number of calibrated Fingertracking hands	additional informations	yes ('*glcal*')
`st`	system status	various status values of ATC and cameras	yes ('*st*')

An optional license code is necessary to unlock these identifiers. Please refer to the ***DTRACK3*** User's Guide (Licenses) for more information.

Table 3.1: UDP/IP datagram types

UTC [1] (midnight).

The timestamp typically shows an accuracy of better than $\Delta t_{err} \pm 0.01ms$ with a *Synccard2/3* (used in *ARTTRACK* , *ARTTRACK5* or *TRACKPACK/E* systems) or a *SMARTTRACK3* . With a *SynccardTP* (used in *TRACKPACK* systems) or a *SMARTTRACK2* one can only expect an accuracy of $\Delta t_{err} \pm 0.5ms$.

The timestamp value is reset to zero when passing midnight (UTC)!

Example:

```
ts 39596.024831
```

3.3 Number of Calibrated Bodies (Additional Informations)

Identifier `6dcal`.

Optionally the number of defined bodies can be included in a data set (i.e. not only the tracked ones). This is done within an additional line:

[1] Coordinated Universal Time = Greenwich Mean Time

Example:

```
6dcal 3
```

☞ **Note: Only bodies are included where the identifier does not feature a counter on its own. In particular, `6dcal` counts all bodies that show up in the output lines `6d`, `6df` and `6dmt`.**

3.4 6DOF Standard Body

Identifier `6d`.

This identifier carries measurement data of all tracked 6DOF standard bodies (i.e. all 6DOF bodies except ***Flystick2/2+/3*** , Measurement Tools , etc.) including all hybrid bodies. Bodies that are not tracked by the system (e.g. being outside the measurement volume), do not appear in the output.

- The first number behind the identifier `6d` is counting the amount of tracked bodies (less or equal to the number of activated bodies).
- The data of each tracked body shows up in three consecutive blocks (each enclosed by `[]`):

$$\texttt{[id qu]}\,\texttt{[}s_x\ \ s_y\ \ s_z\ \ \eta\ \ \theta\ \ \phi\texttt{]}\,\texttt{[}b_0\ \ b_1\ \ b_2\ \ b_3\ \ b_4\ \ b_5\ \ b_6\ \ b_7\ \ b_8\texttt{]}$$

- The blocks contain:
  1. ID number (`id`, starting with 0), quality value (`qu`, unused),
  2. Position ($s_x\ s_y\ s_z$), orientation angles ($\eta\ \theta\ \phi$),
  3. Orientation of the body given as rotation matrix (b_i)
- The nine values $b_0 \dots b_8$ form the rotation matrix R like this:

$$R = \begin{pmatrix} b_0 & b_3 & b_6 \\ b_1 & b_4 & b_7 \\ b_2 & b_5 & b_8 \end{pmatrix}$$

- All numbers are separated by spaces (hex `20`).

When connecting *DTRACK3* to an application, problems may occur due to the different definitions of rotation angles. To avoid this, we recommend to use rotation matrices.

☞ **Note: For orientations close to $\theta = \pm 90°$ the other two axes are driven into a parallel configuration with the system effectively losing one degree of freedom. In this situation the rotation angles will show strange behaviour, e.g. large odd-looking changes. This is also known as the so called “Gimbal Lock”.**

Example (one line):

```
6d 1 [0 1.000][326.848 -187.216 109.503 -160.4704 -3.6963
-7.0913][-0.940508 -0.339238 -0.019025 0.333599 -0.932599 0.137735
-0.064467 0.123194 0.990286]
```

3.5 6DOF Inertial Body (Extended Format)

Identifier `6di`.

Measurement data of all tracked standard 6DOF bodies (i.e. all 6DOF bodies except ***Flystick2/2+/3*** , Measurement Tools , etc.) including all hybrid bodies. Bodies that are not tracked by the system (e.g. being outside the measurement volume), do not appear in the output. This output data format contains values specifically for hybrid bodies (i.e. tracking status, drift error) but can be used for any standard 6DOF body.

- The first number behind the identifier is counting the amount of tracked bodies (less or equal to the number of activated bodies).
- The data of each tracked body shows up in three consecutive blocks (each enclosed by `[]`):

$$\texttt{[id st er][}s_x\ \ s_y\ \ s_z\texttt{][}b_0\ \ b_1\ \ b_2\ \ b_3\ \ b_4\ \ b_5\ \ b_6\ \ b_7\ \ b_8\texttt{]}$$

- The blocks contain:
  1. ID number (`id`, starting with 0), status of the tracking (`st`, 0: not tracked, 1: inertial tracking, 2: optical tracking, 3: inertial and optical tracking), current drift error estimation (`er`, unit: [deg], estimate rising by 10 degree per minute when tracking inertially),
  2. Position (s_x s_y s_z),
  3. Rotation matrix (b_i) of the body's orientation.
- The nine values $b_0 \dots b_8$ form the rotation matrix R like this:

$$R = \begin{pmatrix} b_0 & b_3 & b_6 \\ b_1 & b_4 & b_7 \\ b_2 & b_5 & b_8 \end{pmatrix}$$

- All numbers are separated by spaces (hex `20`).

Example (one line):

```
6di 2 [0 1 2.135][326.848 -187.216 109.503][-0.940508 -0.339238
-0.019025 0.333599 -0.932599 0.137735 -0.064467 0.123194 0.990286] [1 0
0.000][0.000 0.000 0.000][0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000]
```

3.6 6DOF Covariance

Identifier `6dcov`.

Only available if the appropriate license is present for *DTRACK3* .

This identifier carries the covariance of each tracked 6DOF standard body (i.e. all 6DOF bodies except ***Flystick2/2+/3*** , Measurement Tools , etc.) including all hybrid bodies.

- The first number behind the identifier `6dcov` is counting the amount of tracked bodies (less or equal to the number of activated bodies).
- The data of each tracked body shows up in two consecutive blocks (each enclosed by `[]`):

  [id c_x c_y c_z][σ_{11} σ_{12} σ_{13} σ_{14} σ_{15} σ_{16} σ_{22} σ_{23} σ_{24} σ_{25} σ_{26} σ_{33} σ_{34} σ_{35} σ_{36} σ_{44} σ_{45} σ_{46} σ_{55} σ_{56} σ_{66}]

- The blocks contain:
  1. ID number (`id`, starting with 0), position of the center of rotational error CRE (c_i),
  2. the covariance matrix (σ_{ij}); the matrix is symmetrical, thus not all elements are listed.
- The 21 values σ_{ij} form the covariance matrix Σ like this:

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} & \sigma_{14} & \sigma_{15} & \sigma_{16} \\ \sigma_{12} & \sigma_{22} & \sigma_{23} & \sigma_{24} & \sigma_{25} & \sigma_{26} \\ \sigma_{13} & \sigma_{23} & \sigma_{33} & \sigma_{34} & \sigma_{35} & \sigma_{36} \\ \sigma_{14} & \sigma_{24} & \sigma_{34} & \sigma_{44} & \sigma_{45} & \sigma_{46} \\ \sigma_{15} & \sigma_{25} & \sigma_{35} & \sigma_{45} & \sigma_{55} & \sigma_{56} \\ \sigma_{16} & \sigma_{26} & \sigma_{36} & \sigma_{46} & \sigma_{56} & \sigma_{66} \end{bmatrix}$$

- All numbers are separated by spaces (hex `20`).
- The output of the center of rotational error (CRE) $\vec{c}$ and the covariance matrix Σ is given in the room coordinate system.
- All lengths are given in **mm**, all angles in **rad**. Thus the units of the elements σ_{ij} of the covariance matrix Σ are as follows:
  - mm^2 for σ_{ij} with $i \in (1,2,3)$ and $j \in (1,2,3)$.
  - $mm \cdot rad$ for σ_{ij} with $i \in (1,2,3)$ and $j \in (4,5,6)$.
  - rad^2 for σ_{ij} with $i \in (4,5,6)$ and $j \in (4,5,6)$.

☞ **Note: Angles are given in rad instead of deg used for 6DOF standard bodies**

Example (one line):

```
6dcov 1 [0 490.785 747.503 719.391][2.052e-02 1.655e-02 3.055e-02 4.644e-05
1.083e-05 -1.313e-05 3.490e-02 4.708e-02 5.673e-05 -2.623e-05 -9.982e-06
9.499e-02 9.888e-05 -3.749e-05 -1.522e-05 1.502e-05 1.157e-06 -3.161e-06
1.406e-05 -3.849e-06 4.041e-06]
```

3.7 3DOF Marker

Identifier `3d`.

The format of additional 3DOF markers (i.e. markers that can not be recognized as part of a rigid body) is a reduced format of standard 6DOF bodies:

- The first number behind the identifier `3d` is counting the amount of tracked additional 3DOF markers.
- The data of each 3DOF marker shows up in two consecutive blocks (each enclosed by `[]`):

$$[\texttt{id qu}][s_x \; s_y \; s_z]$$

- The blocks contain:
  1. ID number (`id`, starting with `1`), a quality value (`qu`, unused), and the position (s_x s_y s_z).

Example (one line):

```
3d 6 [79 1.000][210.730 -90.669 -108.554] [83 1.000][61.235 -165.625
3.217] [87 1.000][123.633 -107.836 0.110] [88 1.000][212.383 -133.640
77.199] [90 1.000][326.455 -187.055 109.589] [91 1.000][303.185
-239.771 114.861]
```

3.8 3DOF Covariance

Identifier `3dcov`.

Only available if the appropriate license is present for *DTRACK3* .

This identifier carries the covariance of each tracked 3DOF single markers (i.e. markers that can not be recognized as part of a rigid body).

- The first number behind the identifier `3dcov` is counting the amount of tracked additional 3DOF markers.

Chapter 3

- The data of each 3DOF marker shows up in two consecutive blocks (each enclosed by `[]`):

$$\texttt{[id]}\,\texttt{[}\sigma_{11}\ \ \sigma_{12}\ \ \sigma_{13}\ \ \sigma_{22}\ \ \sigma_{23}\ \ \sigma_{33}\texttt{]}$$

- The blocks contain:
  1. ID number (`id`, starting with 1),
  2. the covariance matrix (σ_{ij}); the matrix is symmetrical, thus not all elements are listed.
- The 6 values σ_{ij} form the covariance matrix Σ like this:

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{12} & \sigma_{22} & \sigma_{23} \\ \sigma_{13} & \sigma_{23} & \sigma_{33} \end{bmatrix}$$

- All numbers are separated by spaces (hex `20`).
- The output of the covariance matrix Σ is given in the room coordinate system.
- All lengths are given in mm. Thus the units of the elements σ_{ij} of the covariance matrix Σ are in mm^2.

Example (one line):

```
3dcov 1 [1][2.052e-02 1.655e-02 3.055e-02 4.644e-05 1.083e-05 -1.313e-05]
```

3.9 Flystick

Identifier `6df2`.

☞ **Note: This format version replaced the older `6df` format (see 3.10 on page 29). Use it whenever possible.**

This format provides tracking data for Flysticks (***Flystick2/2+/3***) and other ***ART*** radio devices:

- The first number behind the identifier `6df2` is counting the amount of defined ***Flystick*** .
- The second number gives the number of ***Flystick*** data, that are following in the line.
- The data of each ***Flystick*** shows up in four consecutive blocks (each enclosed by `[]`):

$$\texttt{[id qu nbt nct]}\,\texttt{[}s_x\ \ s_y\ \ s_z\texttt{]}\,\texttt{[}b_0\ \ b_1\ \ b_2\ \ b_3\ \ b_4\ \ b_5\ \ b_6\ \ b_7\ \ b_8\texttt{]}\,\texttt{[}bt_0\ \ \ldots\ \ ct_0\ \ ct_1\ \ \ldots\texttt{]}$$

- The blocks contain:
  1. ID number (`id`, starting with 0), quality value (`qu`, see below), the number of available buttons and controllers (`nbt` and `nct`),
  2. Position of the ***Flystick*** (s_x s_y s_z),
  3. Orientation of the ***Flystick*** given as rotation matrix (b_i, like standard bodies),
  4. Status of buttons (bt_i, see below) and controllers (ct_i, see below).
- The quality value (`qu`) can only become the values 1.000 or -1.000. -1.000 means that the ***Flystick*** is not visible at the moment. Even in that case a ***Flystick*** appears in the output data. Then dummy values are used for position (zero) and orientation (zero matrix!). Still information about buttons and controllers are valid as long as the wireless transmission is active.
- When buttons of the ***Flystick*** are pressed the (decimal) numbers bt_i change. They are coded binary (i.e. button 1 activated = bit 0 set, button 2 activated = bit 1 set, ...) with a maximum of 32 buttons per bt_i number.

☞ **Note: The number of bt_i numbers in the block can vary between specific *Flystick* hardware! If the device does not feature any buttons, the output will not contain any bt_i number!**

- Controller elements are transferred into floating point numbers ct_i, reaching from -1.00 to 1.00. In the output line they follow the buttons information (one number for each controller).

☞ **Note: The number of ct_i numbers in the block can vary between specific *Flystick* hardware! If the device does not feature any controller elements, the output will not contain any ct_i number!**

Example (one line) for two devices, one ***Flystick2*** (ID 0) and one ***Flystick3*** (ID 1, currently outside the measurement volume):

```
6df2 2 2 [0 1.000 6 2][-228.992 270.818 92.561][0.758006 -0.652230
0.004807 -0.651759 -0.757133 0.044271 -0.025236 -0.036691 -0.999008][5
0.13 -1.00] [1 -1.000 4 2][0.000 0.000 0.000][0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000][1 1.00 0.00]
```

3.10 Flystick (Old Format)

Identifier `6df`.

☞ **Note: Supported for backward compatibility. It is recommended to use the format `6df2` (see chapter 3.9 on page 28) whenever possible.**

☞ **Refer to chapter 2.1 on page 18 to find out which buttons of the *Flystick2/2+/3* are NOT transmitted when using the old output format `6df`.**

This compatibility format for Flysticks is similar to the format of standard 6DOF bodies:

- The first number behind the identifier `6df` is counting the amount of defined ***Flystick*** .
- The data of each ***Flystick*** shows up in three consecutive blocks (each enclosed by `[]`):

$$\texttt{[id qu bt]}\,\texttt{[}s_x\ \ s_y\ \ s_z\ \ \eta\ \ \theta\ \ \phi\texttt{]}\,\texttt{[}b_0\ \ b_1\ \ b_2\ \ b_3\ \ b_4\ \ b_5\ \ b_6\ \ b_7\ \ b_8\texttt{]}$$

- The blocks contain:
  1. ID number (`id`, starting with 0), quality value (`qu`, see below) and button information (`bt`, see below),
  2. Position ($s_x\ s_y\ s_z$) and orientation angles ($\eta\ \theta\ \phi$) and
  3. Orientation of the ***Flystick*** given as rotation matrix (b_i, like 6DOF standard bodies).
- The quality value (`qu`) can only become the values 1.000 or -1.000. -1.000 means that the ***Flystick*** is not visible at the moment. Even in that case a ***Flystick*** appears in the output data. Then dummy values are used for position (zero) and orientation (zero matrix!). Still information about buttons and controllers are valid as long as the wireless transmission is active.
- When buttons of the ***Flystick*** are pressed the (decimal) number `bt` changes. It is coded binary (i.e. button 1 activated = bit 0 set, button 2 activated = bit 1 set, . . .).

Example (one line):

```
6df 1 [0 1.000 2][261.103 116.520 41.085 19.6522 -57.3530 116.5992]
[-0.241543 0.968868 -0.054332 -0.482366 -0.168461 -0.859619
-0.842010 -0.181427 0.508039]
```

3.11 Measurement Tool

Identifier `6dmt2`.

Only available if the Measurement Tool license is present for *DTRACK3* .

Note: This format version replaced the older `6dmt` format (see 3.12 on page 32). Use it whenever possible!

The output format for measurement tools is defined as follows:

- The first number behind the identifier `6dmt2` is counting the amount of defined Measurement Tools .
- The second number gives the number of Measurement Tool data that are following in the line.

- The data of each Measurement Tool shows up in five consecutive blocks (each enclosed by `[]`):

$$\texttt{[id qu nbt rd][}s_x \; s_y \; s_z\texttt{][}b_0 \; b_1 \; b_2 \; b_3 \; b_4 \; b_5 \; b_6 \; b_7 \; b_8\texttt{][bt][}\sigma_{11} \; \sigma_{12} \; \sigma_{13} \; \sigma_{22} \; \sigma_{23} \; \sigma_{33}\texttt{]}$$

- The blocks contain:
  1. ID number (`id`, starting with 0), quality value (`qu`, see below), number of buttons (`nbt`) and the radius of the Measurement Tool tip sphere (`rd`, if applicable),
  2. Measured position ($s_x \; s_y \; s_z$) of the tip,
  3. Orientation of the Measurement Tool tip given as rotation matrix (b_i, like standard bodies),
  4. Button status (`bt, see below`)
  5. Covariance matrix (σ_{ij}) of the position of the tool tip (in mm^2).

- The quality value (`qu`) can only become the values 1.0 and -1.0. -1.0 means that the Measurement Tool is not visible at the moment. Even in that case a Measurement Tool appears in the output data. Then dummy values are used for position (zero) and orientation (zero matrix!). Still information about the button is valid.

- When a button on the Measurement Tool is pressed the (decimal) number bt changes. It is coded binary (i.e. button 1 activated = `0x01`). This is also a signal for ***DTRACK3*** that the duration of a measurement has started and a measurement of point is ongoing. All other values `0x02`, `0x04`, ... show a designated button press on the Measurement Tool

- The nine values $b_0 \ldots b_8$ form a rotation matrix R according to the scheme described in chapter 3.4 on page 24.

- $\sigma_{11} \ldots \sigma_{33}$ denote the (symmetric) covariance matrix Σ of the Measurement Tool tip, given in units mm^2 :

$$\Sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{12} & \sigma_{22} & \sigma_{23} \\ \sigma_{13} & \sigma_{23} & \sigma_{33} \end{pmatrix}$$

Example (one line):

```
6dmt2 1 1 [0 1.000 4 2.000][326.848 -187.216 109.503][0.911812 -0.038421
0.408806 0.095040 0.988324 -0.119094 -0.399457 0.147444 0.904817][0]
[8.178e-04 9.166e-04 1.084e-03 4.463e-02 9.025e-03 1.286e-02]
```

3.12 Measurement Tool (Old Format)

Identifier `6dmt`.

Only available if the Measurement Tool license is present for *DTRACK3*

Note: Supported for backward compatibility. It is recommended to use format `6dmt2` (see chapter 3.11 on page 30) whenever possible.

The compatibility format for Measurement Tools is similar to the format of standard 6DOF bodies:

- The first number behind the identifier `6dmt` is counting the amount of defined Measurement Tools .
- The data of each Measurement Tool shows up in three consecutive blocks (each enclosed by `[]`):

$$\texttt{[id qu bt][}s_x\ s_y\ s_z\texttt{][}b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7\ b_8\texttt{]}$$

- The blocks contain:
  1. ID number (`id`, starting with 0), quality value (`qu`, see below) and button information (`bt`, see below),
  2. Measured position (s_x s_y s_z) of the tip,
  3. Orientation of the Measurement Tool tip given as rotation matrix (b_i, like standard bodies),
- The quality value (`qu`) can only become the values 1.0 and -1.0. -1.0 means that the Measurement Tool is not visible at the moment. Even in that case a Measurement Tool appears in the output data. Then dummy values are used for position (zero) and orientation (zero matrix!). Still information about the button is valid.
- When a button on the Measurement Tool is pressed the (decimal) number bt changes. It is coded binary (i.e. button 1 activated = `0x01`). This is also a signal for ***DTRACK3*** that the duration of a measurement has started and a measurement of point is ongoing. All other values `0x02`, `0x04`, . . . show a designated button press on the Measurement Tool
- The nine values $b_0 \ldots b_8$ form a rotation matrix R according to the scheme described in chapter 3.4 on page 24.

Example (one line):

```
6dmt 1 [0 1.000 0][326.848 -187.216 109.503][0.911812 -0.038421
0.408806 0.095040 0.988324 -0.119094 -0.399457 0.147444 0.904817]
```

3.13 Measurement Tool Reference

Identifier `6dmtr`.

Only available if the Measurement Tool license is present for *DTRACK3*)

The output format for measurement tool references is similar to the format of standard 6DOF bodies:

- The first number behind the identifier `6dmtr` is counting the amount of defined Measurement Tool references.
- The second number gives the number of tracked Measurement Tool references.
- The data of each tracked Measurement Tool reference shows up in three consecutive blocks (each enclosed by `[]`):

$$\texttt{[id qu]}[s_x \ s_y \ s_z][b_0 \ b_1 \ b_2 \ b_3 \ b_4 \ b_5 \ b_6 \ b_7 \ b_8]$$

- The blocks contain:
  1. ID number (`id`, starting with 0), quality value (`qu`, see below),
  2. Measured position ($s_x \ s_y \ s_z$) of the Measurement Tool reference,
  3. Orientation of the Measurement Tool reference given as rotation matrix (b_i, like standard bodies)
- The quality value (`qu`) can only become the values 1.0 and -1.0. -1.0 means that the Measurement Tool reference is not visible at the moment. Even in that case a Measurement Tool reference appears in the output data. Then dummy values are used for position (zero) and orientation (zero matrix!).
- The nine values $b_0 \ldots b_8$ form a rotation matrix R according to the scheme described in chapter 3.4 on page 24.

Example (one line):

```
6dmtr 1 1 [0 1.000][-485.245 -67.217 -38.328][0.681257 -0.315034
0.660790 -0.477531 -0.875410 0.074967 0.554845 -0.366620 -0.746817]
```

3.14 Fingertracking

Identifier `gl`.

This format provides tracking data to reconstruct the hand with all fingers and phalanxes. The output format for ***Fingertracking*** is defined as follows:

- The first number behind the identifier `gl` is counting the amount of tracked hands.

Chapter 3

- The data of each tracked hand shows up in twelve consecutive blocks (3 fingers per hand) and eighteen (5 fingers per hand) respectively (each enclosed by `[]`):

  `[id qu lr nf]`[s_x s_y s_z][b_0 b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8] [s^f_x s^f_y s^f_z][b^f_0 b^f_1 b^f_2 b^f_3 b^f_4 b^f_5 b^f_6 b^f_7 b^f_8] [r^f_o l^f_o α^f_{om} l^f_m α^f_{mi} l^f_i]...

- The blocks contain (refer to figure 1.2 on page 17):
  1. ID number (`id`, starting with 0), a quality value (`qu`, unused), a number to distinguish left and right hands (`lr`, see below) and the number of tracked fingers (`nf`, see below),
  2. Position of the back of the hand (s_x s_y s_z) given in the room coordinate system,
  3. Orientation of the back of the hand given as rotation matrix (b_i, like standard bodies),
  4. For each finger (starting with the thumb):
     a) Position of the tip of the finger (s^f_x s^f_y s^f_z) given in the hand's coordinate system,
     b) Orientation of the outermost phalanx (b^f_i, like standard bodies), given in the hand's coordinate system,
     c) Radius of the tip of the finger (r^f_o), length of the outermost phalanx (l^f_o), angle between the outermost and the middle phalanx (α^f_{om}), length of the middle phalanx (l^f_m), angle between the middle and the innermost phalanx (α^f_{mi}) followed by the length of the innermost phalanx (l^f_i).

- The value to distinguish left and right hands (`lr`) will show 0 to denote a left hand and 1 for a right hand.

- The number of tracked fingers (`nf`) will become either 3 or 5 corresponding to the amount of finger thimbles used.

- The radius of the finger tip (r^f_o) is used to identify its surface.

- The angles between the single phalanxes (α^f_{om} and α^f_{mi}) as well as their respective lengths (l^f_o l^f_m l^f_i) are calculated using tracked markers and empirical data.

☞ **Note: The total number of blocks corresponds to the amount of supported finger thimbles per hand (e.g. 3-Finger Tactile Feedback or 5-Finger AR).**

☞ **The data shown in the *Fingertracking display* is NOT the position of the finger tip! For calculating the finger tip position you need to consider radius R shown in figure 1.2 on page 17.**

Example (one line):

```
gl 1 [0 1.000 0 5][105.463 130.815 223.663][-0.912174 -0.337275 0.232774
0.147960 -0.800755 -0.580430 0.382160 -0.495012 0.780331][10.9 -46.5
-53.9][0.8984 -0.0329 -0.4380 0.4378 -0.0127 0.8990 -0.0352 -0.9994
```

```
0.0030][10.8 24.4 -18.4 41.7 -9.7 62.6][74.1 -10.0 -44.1][0.4692 -0.0948
-0.8780 0.1029 0.9933 -0.0523 0.8771 -0.0658 0.4758][8.8 18.8 -16.5 28.0
-35.4 46.7][90.0 13.7 -49.1][0.5119 -0.0692 -0.8563 0.0727 0.9967 -0.0371
0.8560 -0.0433 0.5152][9.4 20.4 -19.0 30.6 -26.3 51.0][84.1 33.3
-44.6][0.6038 -0.0949 -0.7915 0.0981 0.9942 -0.0443 0.7911 -0.0509
0.6096][8.8 19.1 -16.7 28.8 -20.4 48.1][66.3 59.1 -40.2][0.7676 -0.0718
-0.6368 0.0675 0.9972 -0.0311 0.6373 -0.0191 0.7704][7.7 15.0 -5.7 23.6
-7.5 39.3]
```

3.15 Number of Calibrated Fingertracking Hands (Additional Informations)

Identifier `glcal`

An optional extra output line carries the number of defined hands.

Example (one line):

```
glcal 1
```

3.16 System Status

Identifier `st`.

This optional output line provides various values about the current status of Controller and cameras. The format is defined as follows:

- The first number behind the identifier `st` is counting the number of following groups of blocks (currently 3).
- Each block group, denoting a specific type of status values, may consist of 2 or more blocks; currently three types of status values are supported.
- **General status values** show up in two consecutive blocks (each enclosed by `[]`):

  `[0 3][nc nb nm]`

  The blocks contain:

  1. ID number (`0`), number of following status values (`3`),
  2. Number of cameras (`nc`), number of currently tracked 6DOF bodies (`nb`), number of currently found additional 3DOF markers (`nm`).

- **Message statistics** show up in two consecutive blocks (each enclosed by `[]`):

  `[1 5][nce ncw noe now ni]`

  The blocks contain:

Chapter 3

1. ID number (1), number of following status values (5),
2. Number of camera-related error messages (`nce`), number of camera-related warning messages (`ncw`), number of other error messages (`noe`), number of other warning messages (`now`), number of info messages (`ni`). All these values are counting since booting the Controller.

- **Camera status values** show up in several ($\texttt{nc} + 1$) consecutive blocks (each enclosed by `[]`):

  ```
  [2 nc 3][cid ns nu mi]...
  ```

  The blocks contain:

  1. ID number (2), number of cameras (`nc`), number of following status values per camera (3),
  2. For each camera one block is added containing a camera ID (`cid`, starting with 0), number of 2DOF reflections seen by this camera (`ns`), number of seen 2DOF reflections used for 6DOF tracking (`nu`), intensity of the brightest pixel (`mi`, between 0 and 10).

☞ **Please note that the number of available block IDs and also the number of status values inside a particular block may be increased in future *DTRACK3* releases.**

Example (one line):

```
st 3 [0 3][4 2 5] [1 5][0 0 0 6 9] [2 4 3][0 14 8 3][1 14 9 3][2 10 4 1]
[3 15 6 1]
```

4 Input of Control Data via Ethernet

DTRACK3 is using ethernet (UDP/IP datagrams) to receive control data from other applications. The UDP/IP datagrams have to be sent to the fixed port 50110 of the corresponding controller.

The following data types are supported:

Identifier	Type of data
`tfb`	Fingertracking Tactile Control Command
`ffb`	Flystick Feedback Control Command

Table 4.1: Input UDP/IP datagrams

4.1 Fingertracking Tactile Control Command

Identifier `tfb`.

☞ **This command works on tactile feedback devices (e.g. *FINGERTRACKING2 Tactile*) only.**

The input format for Tactile Feedback Control Commands is:

- The first number behind the identifier `tfb` is counting the amount of tactile commands `n` followed hereafter.
- Each tactile command sets the feedback of one finger at one hand; it shows up in one block (enclosed by `[]`):

$$[hid_n\ \ fid_n\ \ pd_n\ \ st_n]\ \ [hid_{n+1}\ \ fid_{n+1}\ \ pd_{n+1}\ \ st_{n+1}]\ \ \ldots$$

- The block contains:
  1. Hand ID number (hid, starting with 0, corresponds with H1...H4 of the ***DTRACK3*** frontend,
  2. Finger ID number (fid, running from 0 (thumb) to 4 (little finger, pinkie),
  3. Penetration depth (pd, a value between 0.0 and 1.0 denoting how deep the finger penetrates the virtual object,
  4. Strength (st, a value between 0.0 and 1.0 corresponding to the strength of the feedback.

Chapter 4

☞ **Feedback is turned off when time-out is reached (i.e. several seconds without sending commands). Repeat control commands even if values are not changing!**

☞ **Note: Vibro-tactile feedback devices (i.e. *FINGERTRACKING2 Tactile*) do not show a penetration depth. For compatibility with legacy code, a given penetration depth is multiplied with the strength. This results in a down-scaling of the strength by a low penetration value.**

Example (one line):

```
tfb 2 [0 0 1.0 0.5][0 1 1.0 0.8]
```

Result: Set vibration on 2 fingers

- Hand H1 (Index 0) Thumb (Finger 0) to vibration intensity 0.5 ,
- Hand H1 (Index 0) Index finger (Finger 1) to vibration intensity 0.8 .

The penetration depth is not used in this case, it is set to 1.0 on both fingers.

4.2 Flystick Feedback Control Command

Identifier `ffb`.

☞ **This command works on *Flystick* devices with feedback capabilities (e.g. *Flystick2+*) only.**

The input format for Flystick Feedback Control Commands is:

- The first number behind the identifier `ffb` gives the number of following feedback commands.
- Each feedback command allows to activate feedback at one ***Flystick*** ; it shows up in two blocks (enclosed by `[]`):

$$[fid_n \;\; bd_n \;\; bf_n \;\; vp_n \;\; nav_n][av_{n,0} \;\; av_{n,1} \;\; \ldots \;\;] \quad \ldots$$

- The block contains:
  1. ***Flystick*** ID number (fid, starting with 0, corresponds with F1 ... of the ***DTRACK3*** frontend,
  2. Beep duration (bd), given in ms (milliseconds); a value of 0 denotes no active beeper,
  3. Beep frequency (bf), given in Hz (Hertz); a value of 0 denotes no active beeper,
  4. Vibration pattern ID (vp): an ID value starting with 1 corresponding to a vibration pattern (specific to device); 0 denotes no vibration,
  5. Number of additional values (nav): intended for future use,
  6. Additional values (av): intended for future use; empty, if nav equals 0.

☞ **To activate a beep, both duration and frequency must have non-zero values.**

☞ **Beep and vibration can be started simultaneously.**

Example (one line):

```
ffb 1 [0 500 4400 5 0][]
```

Chapter 4

5 Remote Command Strings

The following commands may be used in combination with DTrackSDK to control the tracking system remotely (e.g. with your media control) and without the ***DTRACK3*** front-end.

Command string	**Description**
dtrack2 tracking start	Start tracking
dtrack2 tracking stop	Stop tracking
dtrack2 system shutdown	Shut down controller or ***SMARTTRACK2*** (WOL active, not available for ***SMARTTRACK3***)
dtrack2 system reboot	Reboot controller (not available for ***SMARTTRACK2***)
dtrack2 get config active_config	Get currently active config ID, e.g. 'config20160808102933552'
dtrack2 set config active_config <ID>	Select an existing configuration. Warning: Must use <ID>, not name of configuration, except for default configuration (see next entry). Find under *Edit* button, e.g. 'config20160808102933552'
dtrack2 set config active_config standard	Set back to default configuration
dtrack2 get status active	Check whether tracking is currently measuring: **none** (stopped), **mea** (measuring), **cal** (running a calibration), **wait** (waiting for external sync input) or **err** (error in starting measurement)
dtrack2 get system access *supported on 2nd connection*	Check whether control connection can be established: **full** if accessible, **none** if not. **none** usually means another front-end is connected
dtrack2 get system accessed_by *supported on 2nd connection*	Ask who is currently connected with full access, e.g. to find front-end blocking shutdown from Crestron/AMX
dtrack2 set output net <channel id> udp <host> <port>	Configure where UDP output data has to be sent to
example: dtrack2 set output net ch02 udp 10.0.0.23 5003	
dtrack2 set output net <channel id> multicast <host> <port>	Configure where UDP multicast output data has to be sent to
example: dtrack2 set output net ch02 multicast 224.0.0.11 5003	
dtrack2 set output active <channel id> <output type> <yes/no>	Activate or deactivate a specific output data identifier
example: dtrack2 set output active ch02 6d yes	

Table 5.1: Remote command strings

A TCP connection to port 50105 of the controller has to be opened to send the commands (see table 5.1 on page 40). The controller replies with either `dtrack2 ok` or with an error message due to wrong commands and `set` requests. In case of a `get` request the resulting reply is a string of the corresponding `set` command.

☞ **Please ensure that the commands are sent exactly in the given way, with exactly one blank (`0x20`) between the elements and no trailing blanks, CR or LF characters, but including a trailing `NUL` (`0x00`) character.**

ℹ **Table 5.1 shows just a short extract of all available remote commands. Please contact *ART* if you want to perform further operations.**

Multiple Remote Connections

DTRACK3 allows multiple clients to connect to the controller, but only the first connection is allowed to send commands except for those supported on secondary connections. Therefore, scripts can test whether another client is connected to the controller. To do this, the command `dtrack2 get system access` checks the availability. It must return **full**; if it returns **none**, another process is already connected to the command port. In this case, the request `dtrack2 get system accessed_by` can be used to find the host of the connection.

5.1 DTrackSDK and DTrackCLI

It is recommended to use DTrackSDK (available in C++ and Java) to implement command support into own software and to use the DTrackCLI software to send commands from scripts. Both DTrackSDK and DTrackCLI are available for Windows and Linux and can be downloaded after registering on the ***ART*** web page at `http://ar-tracking.com/`

When using the DTrackCLI tool, please make sure to send commands and parameters as one argument by using quotes:

```
DTrackCLI myATC -set "config active_config" "config20170724121806418"

DTrackCLI myATC -get "config active_config"
config20170724121806418

DTrackCLI atc-301604025 -cmd "dtrack2 set config active_config config20160808102933552"

DTrackCLI atc-301604025 -get "config active_config"
config20160808102933552
```

To switch back to the default configuration use the following command:

```
DTrackCLI atc-301604025 -set "config active_config" "standard"

DTrackCLI atc-301604025 -get "config active_config"
standard
```

Chapter 5

Linux For Linux shell scripts, additionally the 'nc' command can be used, e.g:

```
\$ echo -n "dtrack2 system shutdown" | nc 10.10.9.44 50105
dtrack2 ok
```

List of Figures

Chapter 5

List of Tables

Index

Chapter 5