DTRACK3 and DTRACK4: 6D Covariance Output

(Advanced Realtime Tracking GmbH & Co. KG, 2026)

This document is intended to be used in combination with the documents:

DTRACK3_Programmers-Guide.pdf and
DTRACK3_Users-Guide.pdf
resp. for DTRACK4:
DT4v4.*.*_programmers_guide.pdf and
DT4v4.*.*_users_guide.pdf

which are available from ART.

1 General Description

DTRACK (DTRACK V.3.1.0 and later) optionally provides detailed information about the accuracy of the currently measured 6dof poses. This information is expressed by a 6D covariance matrix which results from the non-linear optimization calculation. Access to the covariance information is regulated by a complimentary special license in DTRACK3. In DTRACK4 covariance information can be accessed without license.

1.1 Access, Installation and Activation of 6D Covariance Output

In order to access the covariance information in DTRACK3, a complimentary license code is necessary. Please ask your ART representative to provide you the license for “Output of Covariance Data”. Please refer to the DTRACK3_Users-Guide, chapter 7.2 how to install license upgrades.

For including the 6D covariance data in DTRACK’s output data stream, please activate the output of the “6DOF standard body covariance (identifier 6dcov)” via the Output Settings dialog. Please refer to DTRACK3_Users-Guide, chapter 9.11, resp. DT4v4.*.*_users_guide.pdf, chapter 9.10, to manage the settings for data output via ethernet.

1.2 Background

The 6D matrix contains the covariance coefficients for the three location parameters X, Y, Z and the three rotation angles O, P, K. This matrix is symmetric and generally dense, with positive elements on the main diagonal. The off-diagonal elements are the correlations between the 6 parameters.

Under the assumption that each parameter is normally distributed it is possible to derive measures for the confidence levels of the 6D poses from the covariance matrix. The main diagonal elements are the variances of each individual parameter, but as correlations between the parameters are most certainly to occur, the analysis of the covariance matrix should consider them as well.

For a one-dimensional normally distributed random variable the percentages of the observed values which lie within one, two and three standard deviations of the mean are around 68%, 95% respectively 99.7%. This so-called 68-95-99.7 rule does not apply for distributions with a higher dimensionality as it is the case with the 6D tracking results, which obviously has a dimensionality of 6. Under the assumption that each of the 6 parameters of the tracked 6D pose is normally distributed, only around 1.4% of the observations lies within one standard deviation. It appears that the 68%-95%-99.7% rule for one-dimensional variables translates to somewhat like a 1.4%-32%-82.6% rule for the 6D pose covariance.

See “Wang B, Shi W, Miao Z (2015) Confidence Analysis of Standard Deviational Ellipse and Its Extension into Higher Dimensional Euclidean Space. PLoS ONE 10(3): e0118537. doi:10.1371/journal. pone.0118537”.

1.3 Basic Mathematics

The mathematics leading to the 6D covariance matrix is described in mentioned DTRACK3 and DTRACK4 programmers guides which are available from ART.

There you can also find further literature, giving a more detailed description of error model and application of the covariance matrix, for the interested reader.

1.4 How can I use the Covariance Information?

1.4.1 General

With the considerations from above in mind, the 6D covariance can be used to extract accuracy information for the 6 parameters of position (X, Y, Z) and rotation (Rx, Ry, Rz). The structure of the matrix contains a 3x3 block of metric information for the position at the top left, a 3x3 block of radian information for the rotation angles at the bottom right as well as the mixed 3x3 blocks (in the off main-diagonal corners, with identical values, just transposed).

A possible way to use the covariance information is to apply a principal component analysis to derive the eigenvalues and eigenvectors. The eigenvalues and eigenvectors can be used to visualize the 3D ellipsoid for the metric location part of the covariance.

By the same principle the rotation angle part of the covariance can be visualized. It is a little less intuitive, compared with the location part, but can nevertheless provide useful information, especially when only part of a 6dof-target is seen by tracking cameras, which can, in certain cases, lead to very much increased angular errors while positional information is still good.

1.4.2 Continuous Monitoring of System Setup

The continuous 6D covariance output provided by DTRACK3 and DTRACK4 can be used for a continuous monitoring of “calibration coherence” of the tracking system during operation. For this purpose we recommend to program a software routine for the assessment, which:

- once takes reference values from the freshly calibrated tracking system when a set of “normal” target movements is carried out,
- continuously monitors the according accuracy values derived from the 6D covariance data stream,
- continuously compares the current values with the reference values and
- warns the user if the current values derive from the reference values by more than a certain factor.

Which accuracy values should be used for this assessment?

→ We recommend to:

1. read the 6D Covariance matrix,
2. carry out the principal component analysis,
3. calculate the sum of the squared semi axes, separate for XYZ position and angles (i.e. 3 summands each),
4. and use the square root of both sums, as calculated in (3.), for assessment.

The proper deviation factor of current values from the initially taken reference values, which may serve as a limit for giving out a warning resp. a „please recalibrate“ message, surely depends on the application and on the occurrence of occlusions which hide bigger parts of a target from the cameras. For smaller volumes (desktop size) and situations where not too many occlusions occur, we got meaningful results with a factor 3. As occlusions especially affect the angular part of the covariance, applying different factors for the positional and the rotational values may be reasonable for certain use cases.

1.4.3 What can I conclude and what can't be concluded?

The covariance calculation in DTRACK3 and DTRACK4 includes all error components affecting the 6dof calculation. So the monitoring as described in (1.3.2) can give you valuable online information, if one of the calibration steps that contribute to the final DTRACK output is misaligned. Reasons for such a misalignment can be:

1. Camera calibration (= Internal Orientations)
2. Room calibration (= External Orientations)
3. Body geometry or body calibration (i.e. body deformed or badly calibrated)

However the 6D covariances will not tell you which of the different possible reasons for misalignment is really the cause. I.e., you will not be able to decide if camera calibration, room calibration or body calibration leads to bad covariance values. In order to accomplish this decision, ART offers an offline „Calibration Checker" tool, which allows to assess the quality of camera calibration and room calibration on the base of a Wand movement which the user has to carry out while tracking is stopped.

6D covariances will also not tell you if an unfavorable tracking situation is reason for temporarily bad values. For example, the user moves in a car or car mockup, looks after something in the glove compartment, and thus the tracked head target is only partly visible due to occlusions. In such a case the user must be able to judge a possible warning, i.e. to conclude that the warning is not caused by any bad calibration.

2 Here are the Details for the Software Engineer

2.1 Data Format

The data format of the 6D covariance data is given in the documents:
DTRACK3_Programmers-Guide.pdf, chapter 3.6 or
DT4v4.*.*_programmers_guide.pdf
which are available from ART (data format did not change from DTRACK3 to DTRACK4).

2.2 Data Access via DTrack SDK

Data access via DTrackSDK, see in the doc-directory in the DTrackSDK:

- DTrackSDK_v2.9.0/doc/html/

Relevant html-files for body pose data access including 6D covariances:

- class_d_track_s_d_k.html
- class_d_track_parser.html
- struct_d_track_s_d_k___datatypes_1_1_d_track___body___type__d.html

in file DTrackSDK.hpp:

```
/**
 *      \brief DTrack SDK main class derived from DTrackParser.
 *
*       All methods to access tracking data are located in DTrackParser.
*/
class DTrackSDK : public DTrackParser
```

in file DTrackParser.hpp:

```
const DTrack_Body_Type_d * DTrackParser::getBody( int id ) const
```

in file DTrackDataTypes.h:

```
typedef struct{
        int id;
        double quality;
        double loc[3];
        double rot[9];
        double covref[3];
        double cov[36];
} DTrack_Body_Type_d;
```

2.3 Principal Component Analysis

Here is an instruction by means of commented source code:

```
// 6x6 covariance of body pose
// metric values (x, y, z) in mm*mm, angular values (o, p, k) radian*radian, mixed values mm*radian
// matrix is symmetric (xy == xy, etc )

// xx yx zx ox px kx
// xy yy zy oy py ky
// xz yz zz oz pz kz
// xo yo zo oo po ko
// xp yp zp op pp kp
// xk yk zk ok pk kk

// getting covariance of location
// xx yx zx
// xy yy zy
// xz yz zz

// the library Eigen (http://eigen.tuxfamily.org) is used for linear algebra
// for detailed information see:
// https://eigen.tuxfamily.org/dox/group__Eigenvalues__Module.html
// last access 22/04/2021

// example values for covariance matrix
const double cxx = 2.490900e-02;
const double cxy = 5.771700e-03;
const double cxz = -8.664800e-03;

const double cyy = 2.621700e-02;
const double cyz = -1.255000e-02;

const double czz = 4.292500e-02;

// 3x3 covariance matrix, for example location part of 6D covariance
Eigen::Matrix3d cov3D = Eigen::Matrix3d::Zero();

// fill covariance matrix
cov3D( 0, 0 ) = cxx;
cov3D( 0, 1 ) = cxy;
cov3D( 0, 2 ) = cxz;

cov3D( 1, 0 ) = cxy;
cov3D( 1, 1 ) = cyy;
cov3D( 1, 2 ) = cyz;

cov3D( 2, 0 ) = cxz;
cov3D( 2, 1 ) = cyz;
cov3D( 2, 2 ) = czz;
```

```
// diagonal elements must all be positive as validity check
if (   cov3D( 0, 0 ) > std::numeric_limits< double >::epsilon()
    && cov3D( 1, 1 ) > std::numeric_limits< double >::epsilon()
    && cov3D( 2, 2 ) > std::numeric_limits< double >::epsilon() )
{
       // 3D vector with the unsorted values of the squared semi axis
       const Eigen::Vector3d semiAxis = cov3D.eigenvalues().real();

       // semi axis, a should be the biggest, c the smallest
       const double a = std::sqrt( semiAxis( 0 ) );
       const double b = std::sqrt( semiAxis( 1 ) );
       const double c = std::sqrt( semiAxis( 2 ) );

       // with the covariance example from above:
       // a = 0.231124, b = 0.146848, c = 0.138087

       const Eigen::EigenSolver<Eigen::Matrix< double, 3, 3 >> es( cov3D );

       // get direction vectors of semi axis
       Eigen::Matrix3d eigenVecMat;
       eigenVecMat.col( 0 ) = es.eigenvectors().real().col( 0 );
       eigenVecMat.col( 1 ) = es.eigenvectors().real().col( 1 );
       eigenVecMat.col( 2 ) = es.eigenvectors().real().col( 2 );
}
```