# Using magnetometer biases to reduce TTFF

# Background

1. Using magnetometer biases saved during last positioning session on the device can significantly decrease time to first position fix in magnetic-only positioning mode of RTFPPL. It is especially important for indoor navigation using iPhones with iOS, which does not give access to WiFi measurements. It also can decrease convergence time of mixed MFP+WFP positioning on Android phones in venues like Bridgestone factories with sparse WiFi APs.
2. Device magnetometer biases can be changed with time. Therefore, a time tag of the biases must be saved as well as a covariance matrix of the biases. The time tag is used for magnetometer biases age calculation.
3. Magnetometer biases must be saved before closing the app and before a long pause.

# How to save the magnetometer biases

After positioning with RTFPPL has been finished, the following procedure must be performed for saving fresh-updated magnetometer biases:

1. Call FPEngine\_getMagneticBias function, which is declared in CFppe.h:

bool FPEngine\_getMagneticBias(CIFPEngine\* fpEngine, CMagneticCalibrationParam \*bias\_cov)

The function fills CMagneticCalibrationParam structure with actual magnetometer biases, their covariance matrix and internal RTFPPL time tag associated with the bias:

bias\_cov.timestamp; // RTFPPL time tag related with the bias in msec

bias\_cov.mX; // x component of magnetic bias in uT

bias\_cov.mY; // y component of magnetic bias in uT

bias\_cov.mZ; // z component of magnetic bias in uT

bias\_cov.covarianceMatrix;// magnetic bias covariance matrix, all elements are in uT\*uT

The function returns validity of magnetometer biases. If FPEngine\_getMagneticBias returns false, magnetometer biases must not be saved.

The function must be called before fpEngine object has been destroyed.

1. Save obtained biases values as well as their covariation matrix and bias system time until use them during the next positioning session. Any convenient way can be used for bias storage. It can be, for example, a JSON file or a text file, or a long-term phone memory.

Note, internal RTFPPL time tag related with the bias is based on the time of TPN packages provided in to RTFPPL and cannot be directly used as bias system time . It can be converted as follows:

Where - RTFPPL time tag associated with the bias, - is a system time of TPN time scale start.

can also be determine as system time when then function FPEngine\_getMagneticBiashas been called.

# How to use the magnetometer biases that has been saved during a previous positioning session

During initialization of RTFPPL, check existence of magnetometer biases saved during a previous positioning session with RTFPPL or obtained from another source. If magnetometer biases are available, then the following procedure is to be performed after fpEngine object creating and before calling FPEngine\_setStartPosition function or start of TPN/WiFi/BLE/Framework-data providing in to RTFPPL.

1. Calculate the age of biases:

Where is current system time and is system time when the biases have been calculated.

1. Fill members of CMagneticCalibrationParam structure with actual magnetometer biases, their covariance matrix and age.

This is a pseudocode example how it can be performed:

bias\_cov.timestamp = dT;// bias age in msec - (see item 1 for details)

bias\_cov.mX = mag\_bias\_X; // in uT

bias\_cov.mY = mag\_bias\_Y; // in uT

bias\_cov.mZ = mag\_bias\_Z; // in uT

bias\_cov.covarianceMatrix = mag\_bias\_covariance\_matrix; // all elements are in uT\*uT

1. Call FPEngine\_setMagneticBias function, which is declared in CFppe.h:

void FPEngine\_setMagneticBias(CIFPEngine\* fpEngine, const CMagneticCalibrationParam\* bias\_cov)