CCR Guide - Link Budget

Savina Tsichli

July 2024

List of abbreviations

- **CCR**: Corner Cube Retroreflector
- **ILRS**: International Laser Ranging Service
- **OCS**: Optical Cross Section
- **FOV**: Field of View
- **FFDP**: Far Field Diffraction Pattern

1 Analysis Approach

1.1 What is Link Budget Analysis

The Link Budget is an analysis that aims at providing insights into how effectively the satellite's laser ranging system can detect the reflected laser signal under varying conditions. This is done mainly by the laser link equation that estimates the number of photoelectrons received per laser pulse. This equation factors in critical parameters such as the laser pulse energy, optics efficiencies (both transmitting and receiving), atmospheric losses, and the optical characteristics of the CCRs.
The Link Budget analysis and its theoretical background are described in the Word file ALB_PAY-ARPT_CCR_2023-12-01_v2.1.docx. In this file we evaluate the Detection Capability, assess the System Performance and select the equivalent CCRs based on their dimensions and performance under different conditions.

The Excel file GS_data.xlsx contains a list of all the ILRS NASA stations and their parameters. These can be confirmed at NASA website, under the Site Log tab of every station, ex. λ_{bp} is under Site Log > Telescope Information > Dayl. Filt. Bandwidth. However, most of the parameters are under Site Log > Laser System Information.

1.2 Analysis Description

In this analysis, the spacecraft is pointing the +Z face towards the ground station for the whole duration. The three mission candidate orbits were considered with altitudes of 470 km, 500 km, and 530 km.

In order to select a CCR type and diameter compliant with functional and performance requirements (SAT-FKT-301, PL-CCR-INT-305, PL-CCR-PRF-306) and to evaluate payload performance within the ILRS network, the link budget analysis is performed. The laser link equation was used to evaluate the mean number of photoelectrons received for every transmitted laser pulse. The main objective is to determine the detection probability in every operative condition.The analysis keeps into account the atmospheric losses, the ground station characteristics, both for the laser and for the receiver, the attitude of the satellites in terms of incident angle of the laser on the Corner Cube Retroreflector.

2 Laser Link Formulae

The equations used in this chapter are based on the work of (Degnan J. J., 1993) and (Degnan J. , 2023). The descripted model for simulating CCR photon return does not consider velocity aberration. As the spacecraft is going to be in LEO, the higher orbital velocity might reduce photon return. Effects of velocity aberration are going to be investigated in a further analysis. Also, attitude errors are not being considered. The velocity aberration effect causes the beam to be deflected by an amount $\alpha = \frac{2v}{c}$ in the forward direction of the satellite motion, where v is the relative velocity between the satellite and station and c is the speed of light. If α is large compared to the angular width of the retro FFDP (Far Field Diffraction Pattern), the reflected beam will not be seen by the SLR station.

The laser link equation provides an estimate of photon return based on laser energy, range, array cross section, optics efficiency, and weather:

$$N_{\mathrm{pe}} = \eta_q \left(\frac{E_T}{h\nu} \right) \eta_T G_T \sigma_{\mathrm{eff}} \left(\frac{1}{4\pi R^2} \right)^2 A_R \eta_R T_A^2 T_C^2$$

where η_q is the quantum efficiency of the receiver, E_T is the laser pulse energy in mJ, $h\nu$ is the photon energy and is considered $3.73 \cdot 10^{-19}$ J at 532 nm, η_T is the efficiency of the transmitting optics, G_T is the laser beam gain, σ_{eff} is the CCR optical cross section (OCS), R is the slant range to the target, A_R is the telescope aperture, η_R is the efficiency of the receiving optics, T_A is the one-way atmospheric transmission, and T_C is the one-way transmissivity of cirrus clouds (if present). N_{pe} is the number of detected satellite photoelectrons per laser pulse.

2.1 Optical Cross Section - OCS

The peak optical cross section of a corner cube retroreflector can be computed as:

$$\sigma_S = \rho \frac{\pi^3 D^4}{4\lambda^2}$$

where ρ is the reflectivity, D is the diameter of the CCR, and λ is the laser wavelength. OCS is a function of the incidence angle θ_{inc}:

$$\sigma_{\mathrm{eff}}(\theta_{\mathrm{inc}}) = \eta^2(\theta_{\mathrm{inc}})\sigma_S$$

Where $\eta(\theta_{\mathrm{inc}})$ is the reduction factor:

$$\eta(\theta_{\mathrm{inc}}) = \frac{2}{\pi} \left(\sin^{-1} \mu - \sqrt{2} \tan \theta_{\mathrm{ref}} \right) \cos \theta_{\mathrm{inc}}$$

where θ_{ref} is the internal refracted angle:

$$\theta_{\mathrm{ref}} = \sin^{-1} \left(\frac{\sin \theta_{\mathrm{inc}}}{n} \right)$$

$$\theta_{ref} = \sin^{-1} \left(\frac{\sin \theta_{inc}}{n} \right)$$

where n is the cube index of refraction (1.455 for fused silica). The quantity μ is equal to:

$$\mu = \sqrt{1 - \tan^2 \theta_{ref}}$$

2.2 Transmitter Gain - G_T

The transmitter gain for a Gaussian beam is:

$$G_T = \frac{8}{\theta_e^2} \exp \left(-2 \left(\frac{\theta_p}{\theta_e} \right)^2 \right)$$

where θ_e is the far field divergence half angle of the beam and θ_p is the beam pointing error.

2.3 Slant Range - R

The slant range is given by the expression:

$$R = -(R_E + h_{GS})\cos\theta_{\text{zen}} + \sqrt{(R_E + h_{GS})^2\cos^2\theta_{\text{zen}} + 2R_E(h_S - h_{GS}) + h_S^2 - h_{GS}^2}$$

where R_E is the Earth radius, h_{GS} is the station altitude, h_S is the satellite altitude, and θ_{zen} is the zenith angle.

2.4 Atmospheric Transmission - T_A

The atmospheric transmission is a function of the zenith angle and can be computed as:

$$T_A(\lambda, V, 0) = \exp\left[-\sigma(\lambda, V, 0)h_{GS}\sec(\theta_{\text{zen}})\right]\exp\left(-\frac{h_{GS}}{h_{SPL}}\right)$$

where $\sigma(\lambda, V, 0)$ is the attenuation coefficient at wavelength λ and with sea level visibility V, h_{GS} is the ground station altitude, and h_{SPL} is the scale height (1.2 km).

2.5 Cirrus Transmittance - T_C

Cirrus transmittance is a function of the zenith angle, θ_{zen}, and the mean cirrus cloud thickness, t:

$$T_C = \exp\left[-0.14(t\sec\theta_{\text{zen}})^2\right]$$

where $t = 1.341\,\text{km}$ is the mean cirrus cloud thickness.

2.6 Photoelectron Generated from Background Noise

To evaluate signal detection capabilities, background noise must be determined:

$$N_B = \frac{\eta_q}{h\nu}N_{\lambda BP}\Omega_R A_R \eta_R \tau_g$$

where $N_{\lambda BP}$ is the background spectral radiance (night sky provides a best-case noise background of 3×10^{-6} W/ster m^2, while sunlit clouds provide a worst-case noise background of 1.4×10^{-5} W/ster m^2), Ω_R is the detector FOV solid angle in steradians, λ_{BP} is the width of the bandpass filter, and τ_g is the temporal width of the range gate.

The detector field of view angle is:

$$\Omega_R = \left(\frac{\theta_{\text{FOV}}}{2}\right)^2$$

Where θ_{FOV} is the detector FOV half angle in radians.

2.7 Probability of signal detection

Since the photon detection follows the Poisson probability distribution, the probability per laser pulse of detecting m number of photons (where N is the average number of photoelectrons):

$$P(m, N) = \frac{N^m}{m!}e^{-N}$$

Then the probability of detecting one photon from the background noise, which is the false alarm probability:

$$P_{FA} = 1 - e^{-N_B}$$

The photon detection probability, which is the probability of detecting one photon from the background noise and actual signals, is:

$$P_{PD} = 1 - e^{-N}$$

Where $N = N_{PE} + N_B$ is the total photoelectrons number. Finally, the signal detection probability, the probability of detecting a signal from the background noise, is:

$$P_{SD} = (1 - P_{FA})P_{PD}$$

3 Matlab Code

The MATLAB code calculates the link budget and detection probabilities for the CCRs. This code can be found in the matlab file ”budget.m”.

After defining the constants of the energy of a single photon at 532nm, CCR diameter, wavelength, reflectivity of the coated CCR, altitude (500km), background noise at night and day, the file containing the properties of each station is imported. Then, after defining the quantities mentioned in the formulae above, the final probabilities are calculated by two `for` loops ranging through 20° to 90°. The parameters used are:

- Photon Energy hv at 532nm,
- CCR Diameter d,
- Wavelength l,
- Reflectivity r_o,
- Altitude alt_{alba} at 500 km,
- Background Noise: 2 values for day and night background noise.

A station is considered suitable for laser ranging if at any time detection probability per pulse exceeds 50% in a zenith angle range, for both night and daytime laser ranging. The stations used for this are:

Stations	Location
MATM	Matera, IT
POT3	Potsdam, DE
GRZL	Graz, AT
GRSM	Grasse, FR
SFEL	San Fernando, ES
WETL	Wettzell, DE
ZIML	Zimmerwald, CH
BORL	Borowiec, PL
HERL	Herstmonceux, GB
SISL	Simosato, JP
STL3	Mt Stromlo, AU
SHA2	Shanghai, CN
KUN2	Kunming, CN
HRTL	Hartebeesthoek, ZA
HARL	Hartebeesthoek, ZA
BRAL	Brasilia, BR
SJUL	San Juan, AR
AREL	Arequipa, PE
JFNL	Wuhan, CN
SEJL	Sejong City, KR
GMSL	Golosiiv, UA
BEIL	Beijing, CN
THTL	Tahiti, PF
HA4T	Haleakala, US
MONL	Mon. Peak, US
GODL	Greenbelt, US
YARL	Yarragadee, AU
KTZL	Katzively, UA
IRKL	Irkutsk, RU
BADL	Badary, RU
ZELL	Zelenchukskaya, RU
SVEL	Svetloe, RU
BAIL	Baikonur, KZ
ARKL	Arkhyz, RU
RIGL	Riga, LV
ALTL	Altay, RU
MDVS	Mendeleevo 2, RU
SIML	Simeiz, UA

Table 1: Stations used and their locations.

Based on the formulas mentioned above we calculate the values below:

- Efficiency n,
- Gain G_t: Transmitter gain calculation based on beam divergence and pointing error,

- Optical Cross Section *ocs_peak*,
- Receiver Area A_r,
- Receiver Solid Angle ω_r.

Then, Night and Day Detection Probabilities $P_{SD_{night}}$, $P_{SD_{day}}$ are calculated for each station and exported in the folder "probabilities".

The main loop of the code iterates over zenith angles from 90° to 45° and from 44° to 20°; it performs calculations for each angle to determine the number of photons received and the probabilities of detection. More specifically, it calculates and saves the probability of detecting a signal from the satellite by the stations based on given parameters and data read from the excel file "GS_data". The script integrates parameters from multiple ground stations within the ILRS networks, simulating slant ranges, atmospheric and cirrus transmittance and computing detection probabilities for each station. This is a technique used to measure the distance between the ground-based stations and our satellite. To calculate these for each CCR we change the reflectivity and the diameter accordingly. The results are then exported to the excel files "coated.xlsx", "uncoated.xlsx" and "MRR.xlsx". Each one has two sheets, for Day and Night conditions.

Then, the efficiency of the CCRs for every station during day and night are calculated. You can find the exported plots corresponding to the efficiency of the uncoated and coated CCR as well as the MRR for both night-time and day-time conditions in the folder "efficiencies" and the code in "efficiencies". To calculate them for each condition, we hash out each time in the loop either night or day, also changing the imported excel file for each CCR. The plots are shown with respect to θ_{zen}. As you will notice, in some plots there are straight horizontal lines; these constant values could be due to specific characteristics or behaviors in the underlying calculations. The values of the probabilities are calculated based on the formulae mentioned before:

$$P_{PD} = 1 - e^{-N}$$

for the detection of the signal and noise and:

$$P_{SD} = (1 - P_{FA}) \times P_{PD}$$

for the detection of signal from noise, where N is the total number of detected photons (including noise). P_{PD} is the the probability of detecting both photons and noise, P_{SD} is the the probability of detecting just noise and P_{FA} is the probability of false alarm. When the photon count N becomes very large, the expression of the exponential approaches zero, causing the probability to approach 1.

So, the probability of detection becomes almost constant at 1, leading to a straight horizontal line. Conversely, if N is very small, the exponential is close to 1, making the probability close to 0. This also results in a constant value in the plot. The threshold line corresponds to 50% efficiency that is considered favorable. The night probabilities are generally be higher due to lower background noise, making it easier to distinguish the signal from the noise.

Then, the efficiency of each CCR was then calculated in two situations and we acknowledged two cases; the ideal, where the satellite is exactly over the ground station and the worst, where the satellite is just over the horizon. These two cases correspond to the minimum and maximum distances. In the ideal case, the zenith angle is at 90and the incident angle at 90°. In the worst case the zenith angle is 20° and the incident angle 60°. The matlab code and the corresponding plots can be found under the folder "best_worst".

4 Orekit

Orekit is a library for space dynamics written in Java. We use it as it provides basic elements (orbits, dates, attitude, frames) and various algorithms to handle them (conversions, propagations, pointing). In our analysis, we use it for orbit propagation, frame transformations, attitude handling, orbit determination, event detection and data handling. You can Download orekit from the Orekit official site. In order to use this library, we have to use a Python environment. This is usually done through the Anaconda interface. There, we create an environment containing both Python and Orekit channels, which you can name for example ”Orekit”.

In order to run the code in this environment, we use Jupyter Notebook. To access it, we first have to activate the environment we created with the channels mentioned; we do this by writing `activate Orekit` in the terminal. Then we have to access the interface through which we will use this environment and be able to write code; for this, we use Jupyter Notebook. To access it we now type `jupyter notebook` so a new browser window pops up. There you can create a new Notebook by clicking ”New” and then choosing the Python kernel. You can find the corresponding code in the file ”orekit.ipynb”.

After we import Orekit we run the code. The final result is an estimated orbit that best fits the observational data, as well as visualization of the results, showing how well the estimated orbit matches the observations and analyzes the errors in the estimated parameters. For reference, the steps we follow in the notebook are shown below in pseudocode format.

Algorithm .1: Satellite Orbit Determination

```
Data: Satellite data (TLE, CPF, range measurements, etc.)
Result: Enhanced orbit model, CPF file, validation plots
Step 1: Initialization and Setup;
Initialize environment and libraries (Orekit, Python tools);
Define frames (ECI, ECEF, LVLH) and propagators (SGP4, numerical);
Load satellite data (TLE, CPF, mass, drag coefficient);
Step 2: Data Preparation;
Process range measurement data from ground stations;
Set up observation model, estimator, and optimizer;
Step 3: Orbit Determination;
Build numerical propagator with refined parameters;
Perform optimization to fit range measurements to estimated orbit;
Retrieve optimized orbit parameters;
Step 4: Propagation and State Retrieval;
foreach time step in the propagation period do
    Propagate satellite state using refined propagator;
    Retrieve position and velocity vectors in ECI, ECEF, LVLH frames;
    Compute differences with CPF and TLE data;
    Store results;
end
Step 5: Visualization of Results;
Plot delta position between CPF and estimated orbit in LVLH frame;
Plot delta position between original TLE, fitted TLE, and estimation in LVLH
  frame;
Step 6: TLE Fitting and Enhancement;
Fit a new TLE to the refined orbit using optimization;
Adjust orbital elements and set BSTAR as a free parameter;
Step 7: Generate CPF File;
Propagate orbit for 7 days and save in CPF format;
Export CPF file for laser ranging stations;
Step 8: Validation and Comparison;
Re-propagate using fitted TLE;
Compare fitted TLE with initial TLE and refined orbit;
Visualize results to validate accuracy improvement;
```