4

Constraint Formulation

4.1 Preliminary

As detailed in previous chapters, solving BPP is a process of finding an optimal set of beacon coordinates that with a given set of constraints, optimizes location estimation for given indoor space. Following are few noteworthy observations regarding this process that will form the basis of formulating proposed optimization system in coming sections:

1. BPP is an ***a-priori*** step to live localization. As a precursor, it takes the information about obstacles, observation noise, connectivity and coverage in forms of constraints and delivers appropriate beacon locations.

2. An optimal beacon configuration is expected to provide maximum Line of Sight (LoS) coverage between the beacons and device. This must be ensured as a constraint in the form of direct visibility between CDLs and CBLs as shown in Figure 4.1.

3. The optimal beacon configuration must constrain the propagation of error present in the range observations. In order to be generalized, the optimization process should work with the amount of error rather than its source. For example, measurement noise from sources such as multipath, shadowing, interference is bound to vary with each indoor design and its modelling limits the applicability of the approach. Hence, the strategy is devised to limit the output error, as per the level of error propagation required by the user.

4. Selected beacon configuration must ensure the LoS k-connectivity for all the CDLs, with

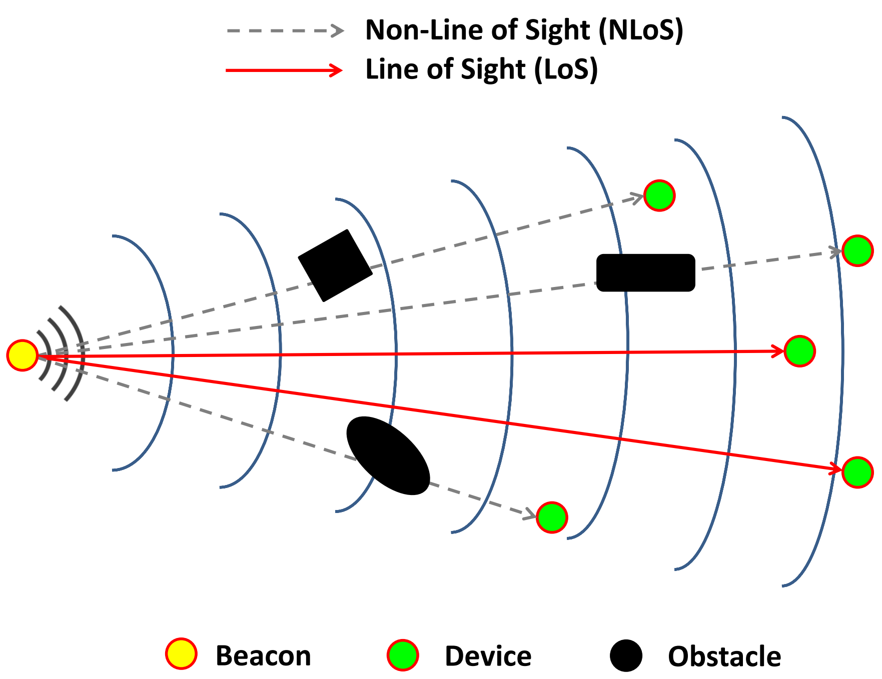

Figure 4.1: 2D depiction of LoS and Non-LoS situations from a beacon to surrounding devices in the presence of obstacles.

chosen threshold of sensing range (R). In other words, the optimal beacon configuration must deliver atleast k visible beacons within the distance R to all the CDLs.

Proceeding further, the next section describes the process of line of sight detection for an indoor 3D point cloud. This is essentially the first step in the optimization process.

4.2 Line of Sight Detection

The intersection of rays with surfaces and voxels have been widely utilized in computer graphics to trace the visibility between two points Zhong Ji et al. [2001]; Yun and Iskander [2015]. However, in order to calculate the node to node adjacency in Wireless Sensor Networks (WSN), ray-tracing algorithms are often modified to incorporate the radio propagation characteristics by Received Signal Strength (RSS) modelling. For real-time applications of RSS modelling such as three dimensional (3D) indoor localization, that requires atleast three observations from different sensors to a device; the availability of direct visibility is vital.

With reference to the point cloud classification described in Chapter 3, this section presents an algorithm that performs Projected Clipping and Orientation Calculation (PCOC) for identifying LoS visibility between beacons and devices. With this context, Section 4.2.1 briefs about the ray-tracing approaches and identifies suitable algorithm to be compared with PCOC. Section 4.2.2 demonstrates the point cloud classification and working of PCOC. Section 4.2.3 compares PCOC with a modified implementation Mena-Chalco [2019] of classical ray-voxel intersection algorithm Amanatides et al. [1987]. Section 4.2.4 discusses the applicability and limitations of PCOC.

4.2.1 Related Works

Investigating the presence of obstacles for a given LoS is typically approached by identifying the intersection of ray as it propagates from it source through the nearby environment. This tracing of ray is generally accompanied by the approximation of surrounding space with bounding volumes or partition into voxel grids Amanatides et al. [1987]. Bounding volumes, which are based on k-d trees are efficient for traversal but reflect high computational complexity for updating and reconstruction Cosenza [2008]. On the other hand, region partitioning based approaches, as introduced in Glassner [1984], approximate 3D spaces with voxel grids. While voxel grids are inefficient for traversal in comparison to k-d trees, their creation time is significantly faster Cosenza [2008]. Even hardware acceleration techniques Yun and Iskander [2015]; Aguado Agelet et al. [2000] that are popular for supporting such massive data manipulation add only to their infeasibility for small scale applications. These storage and access complexities of ray-tracing approaches can easily amount to 60% of the total computation time Wald et al. [2002, 2003], making them uneconomical for WSN having limited memory and computation resources on sensor's circuitry.

As PCOC uses point clouds for representing sensor, device and beacon locations as 3D grid of coordinates, a comparative assessment with voxel grid based ray-tracing techniques stands suitable from systemic requirement of parameters. This will further be detailed in Section 4.2.2 and 4.2.3.

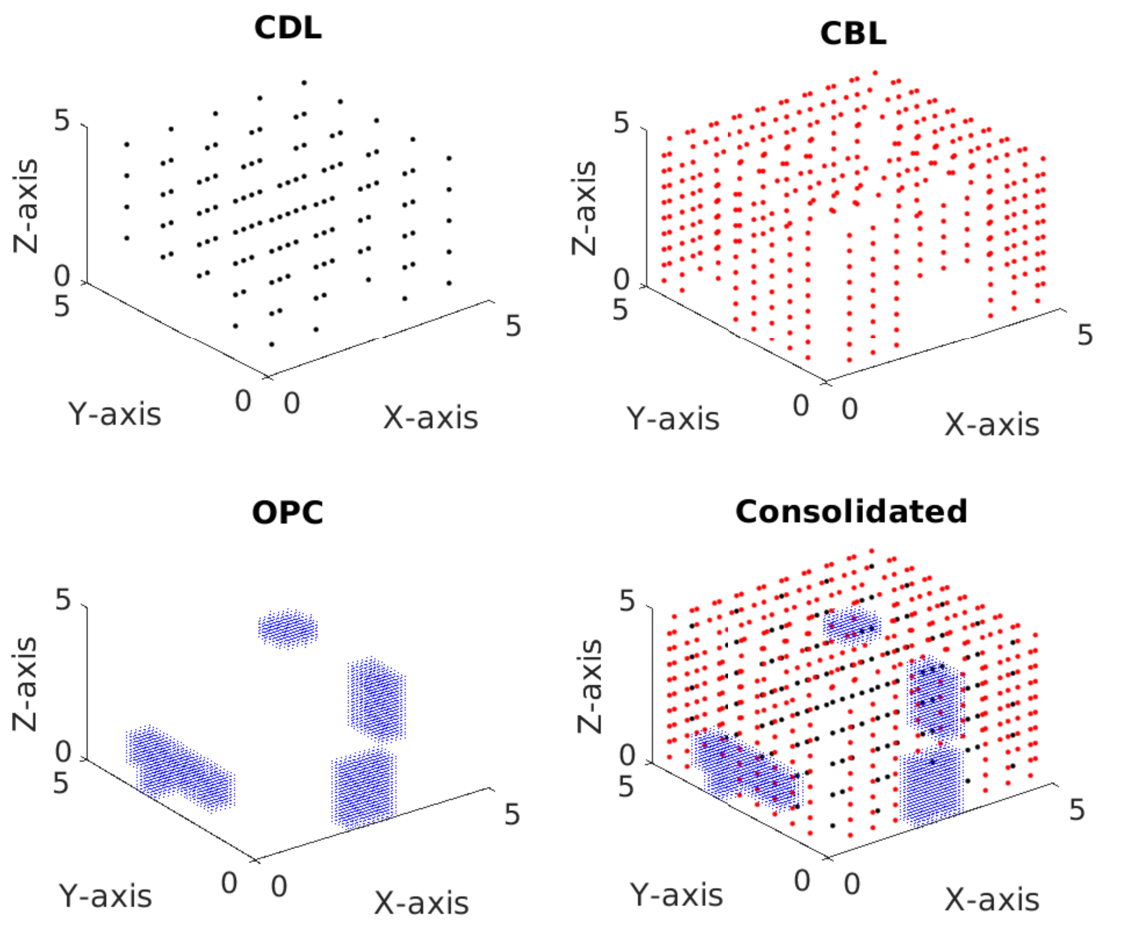

Figure 4.2: An example representation of (a) CDL, (b) CBL and (c) OPC coordinates as simulated in MATLAB

4.2.2 Algorithm

Preliminaries

Figure 4.2 shows an example of the three classes for devices, beacons and obstacles, simulated as coordinate clouds in MATLAB. For the simulation of device coordinates, a parameter set $(\Delta x_d, \Delta y_d, \Delta z_d)$ for Device Grid Size (DGS) is chosen that creates Candidate Device Locations (CDL) inside the periphery of indoor space. Similarly using $(\Delta x_b, \Delta y_b, \Delta z_b)$ as the Beacon Grid Size (BGS) parameters allows the creation of Candidate Beacon Locations (CBL) on the walls and ceiling of an indoor space. Finally, Obstacle Grid Size (OGS) values of $(\Delta x_o, \Delta y_o, \Delta z_o)$ are used to approximate the Obstacle Point Clouds (OPC) present in the indoor surrounding. For the present implementation, a uniform value for DGS, BGS and OGS is used to avoid inducing any design biases. Also, obstacles are assumed to be solid regular convex objects which is also suitable for voxel based approaches that approximates regions by bounding boxes.

PCOC

The main objective of the proposed algorithm remains with the identification of LoS from a CBL to a CDL. Hence, to explain its working, a set of 1 CDL $\{D : (x_d, y_d, z_d)\}$ and 1 CBL $\{B : (x_b, y_b, z_b)\}$ is chosen against the potential obstruction by 1 OPC $\{O : O^i = (x_o^j, y_o^j, z_o^j), \forall j = 1..s\}$ with s coordinates in it. This is collectively represented as $S = \{D, B, O\}$. Let the B to D LoS be denoted as L. Based on this, we proceed to defining the operations of projection, clipping and identification of obstacle's orientation with respect to L in the next subsections.

Projection and Clipping

As the basis of PCOC's computation, if in any one of the three $\{XY, YZ, ZX\}$ coordinate planes, entire O either falls on one side or stays invisible, there exists a LoS between $B - D$ in that plane. Projecting S onto the three coordinate planes results in corresponding subsets as: $S_{xy} = \{D_{xy}, B_{xy}, O_{xy}\}, S_{yz} = \{D_{yz}, B_{yz}, O_{yz}\}, S_{zx} = \{D_{zx}, B_{zx}, O_{zx}\}$. Here X_{ab} is a set containing the coordinates resulting by projecting X onto ab plane. These *projected* subsets S_{xy}, S_{yz} and S_{zx} form a 2D rectangular Region of View (RoV), which is *clipped* by B and D in respective projection planes, as shown in Figure 4.3. Each projected obstacle coordinate, falling beyond its RoV, is assigned a value 2 designating it to be invisible to L. A counter C_2 is provisioned in the algorithm to record the number of invisible coordinates in a projected and clipped OPC. Obstacle coordinates falling within the RoV are further classified for respective orientation with respect to L, as explained in the next subsection.

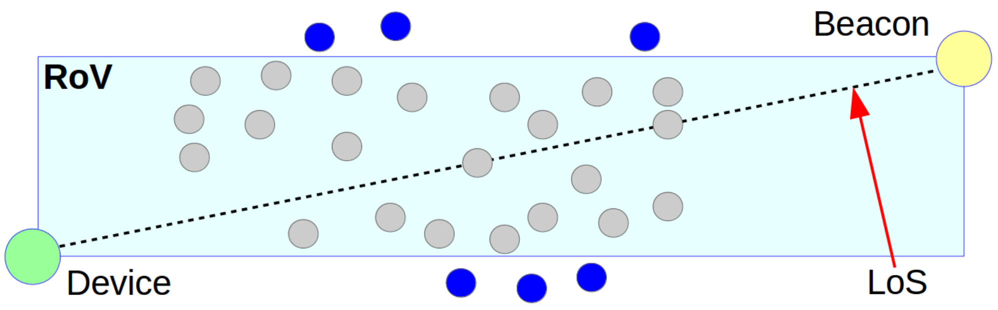

Figure 4.3: Representation of clipped rectangular RoV between beacon-device LoS that classifies contained coordinates (grey) and invisible coordinates (blue)

Orientation

In a 2D Cartesian plane, orientation of a coordinate P with respect to line L_{QR}, which joins Q and R, can be calculated using vector cross product. For two vectors $\vec{V_{QR}} = R - Q$ and $\vec{V_{QP}} = P - Q$, the sense of rotation of their cross products $M = \vec{V_{QR}} \times \vec{V_{QP}}$ reflects the orientation of P with respect to L_{QR}. In other words, the $sign(M)$ operation results in values -1, 0, 1 representing the left, over and right placement of P with respect to L_{QR}. Corresponding to these three numeric values, three counters C_{-1}, C_0, C_1 have been provisioned for recording the placement of clipped and projected OPC coordinates falling within a RoV. This process will be termed as the function SIDE in further discussions. We now present the complete working of PCOC.

Algorithm

With reference to the flow chart in Figure 4.4 and in accordance to Sections 4.2.2 and 4.2.2, the detailed working of PCOC is as the following:

- **Step-1**: Take S as input and initialize $i = 1$ as a counter for three projection planes. For the ease of indexed notation, an array P is used to represent the XY, YZ, ZX planes with respective indices of 1, 2 and 3.

- **Step-2**: Increment i by 1. If $i \leq 3$ move forward to Step-3. If i gets higher than 3, it concludes the unavailability of LoS for S as all the projection planes have been investigated, the execution stops.

- **Step-3**: Project S onto $P(i)$. Initialize counter j for iterative detection of orientation of all O^j coordinates in O. Initialize class counters $C_2^j, C_{-1}^j, C_0^j, C_1^j$ to 0.

- **Step-4**: Increment j by 1. If $j \leq s$, move to Step-5. Otherwise, verify the accumulated counters against the possibilities of (i) $C_{-1} + C_2 = s$ or (ii) $C_1 + C_2 = s$. If any of (i) and (ii) found true, declare the availability of LoS and stop execution. Otherwise, goto Step-2.

- **Step-5**: Calculate SIDE for O^j. If O^j is placed on the left or right to L, increment C_{-1} or C_1 respectively by 1 and move to step . Otherwise move forward to Step-6.

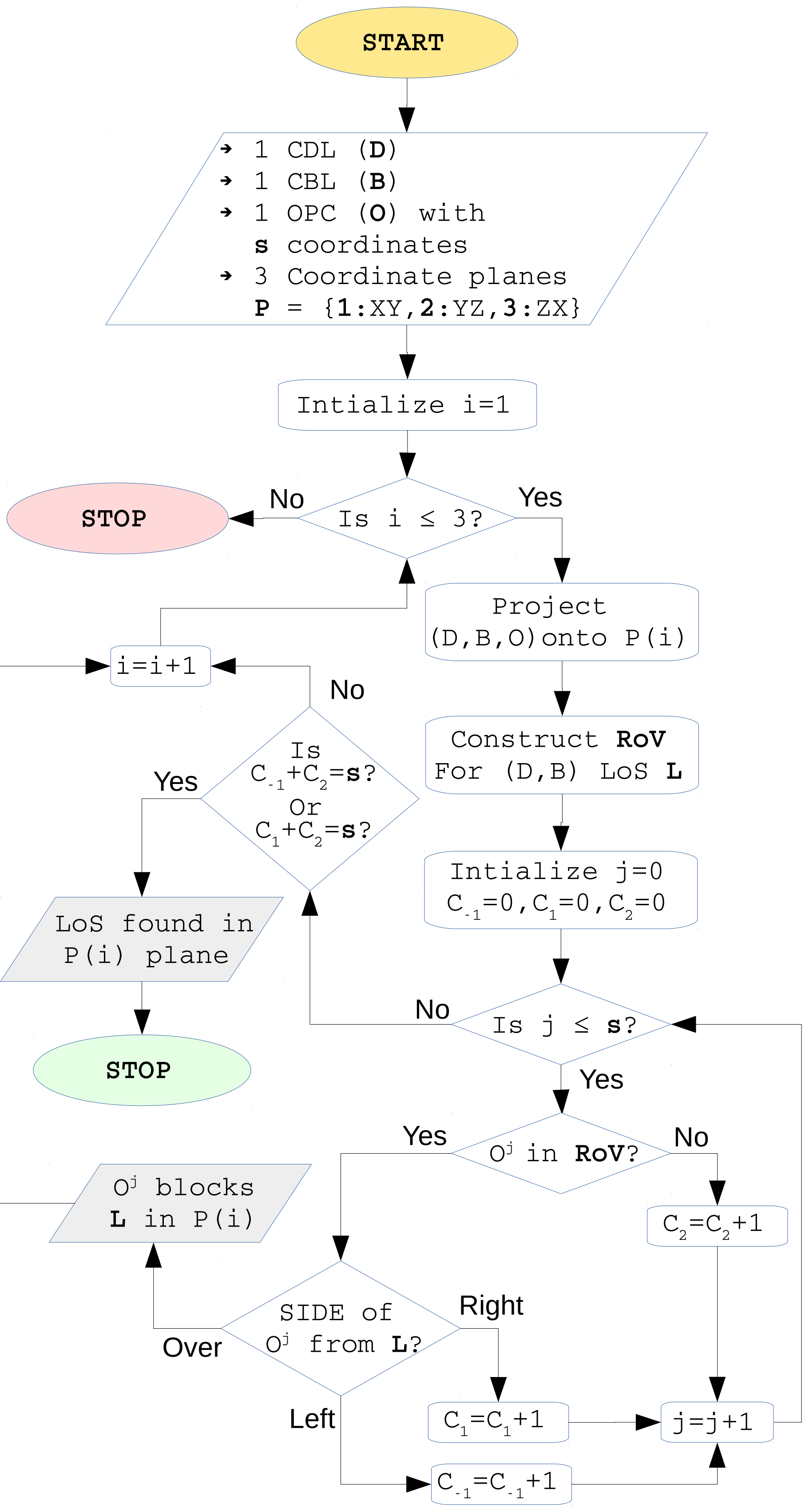
START
➔ 1 CDL (**D**)
➔ 1 CBL (**B**)
➔ 1 OPC (**O**) with **s** coordinates
➔ 3 Coordinate planes **P** = {**1**:XY,**2**:YZ,**3**:ZX}
Intialize i=1
Is i ≤ 3?
No
Yes
STOP
Project (D,B,O) onto P(i)
Construct **RoV** For (D,B) LoS **L**
Intialize j=0 C_{-1}=0, C_1=0, C_2=0
i=i+1
No
Is $C_{-1}+C_2$=**s**? Or C_1+C_2=**s**?
Yes
LoS found in P(i) plane
STOP
No
Is j ≤ **s**?
Yes
Yes
O^j in **RoV**?
No
$C_2=C_2+1$
O^j blocks **L** in P(i)
SIDE of O^j from **L**?
Over
Right
Left
$C_1=C_1+1$
j=j+1
$C_{-1}=C_{-1}+1$

Figure 4.4: Work flow of PCOC

- **Step-6**: If SIDE for O^j results in 0, as it is over L and blocks the LoS, goto Step-2. Otherwise, increment C_2 by 1 as O^j is invisible to $S_{P(i)}$ and goto Step-4.

The above procedure stops with either the declared availability of LoS in Step-4 or after exhausting the search through all the projection planes in Step-2. The next section presents a comparison of PCOC with voxel-based ray-tracing technique for their computational performances.

4.2.3 PCOC v/s Ray-Voxel Intersection

The primary source of complexity for both PCOC and voxel based approaches is the number of coordinates or voxels used for obstacle representation in respective cases. As discussed in Section 4.2.1 and 4.2.2, both the aforementioned approaches utilize grids to approximate obstacles by coordinate clouds and bounding volumes respectively. An intuitive advantage of PCOC over voxel based approaches is that, it starts calculations directly with the coordinates while ray-tracing techniques require an approximation of target obstacle region by bounding volumes, to begin with. We choose to implement a modified version Mena-Chalco [2019] of classical ray-voxel intersection algorithm Amanatides et al. [1987] accompanied with the bounding box calculation by Korsawe [2019]. The time consumptions of the two algorithms were recorded for a set of uniformly generated random point clouds, ranging from 10^1 to 10^8 in coordinate count. This range was further divided into 100 uniform logarithmic subdivisions. The average execution times of both the algorithms were calculated for 100 repetitions over each of the 100 coordinate counts. Simulations were performed using MATLAB on a 2.4 GHz Intel® Xeon® E5-2640 v4 CPU with 32 GB RAM.

The resulting performances of both PCOC and voxel methods have been show in Figure 4.5 with linear and logarithmic scales. As can be seen with linear scale, for higher number of coordinates in obstacle cloud, PCOC provides an approximately five fold improvement in time consumption over voxel-based approach. This is also demonstrated in the logarithmic scale as a consistent better asymptotic performance of PCOC against the voxel-method.

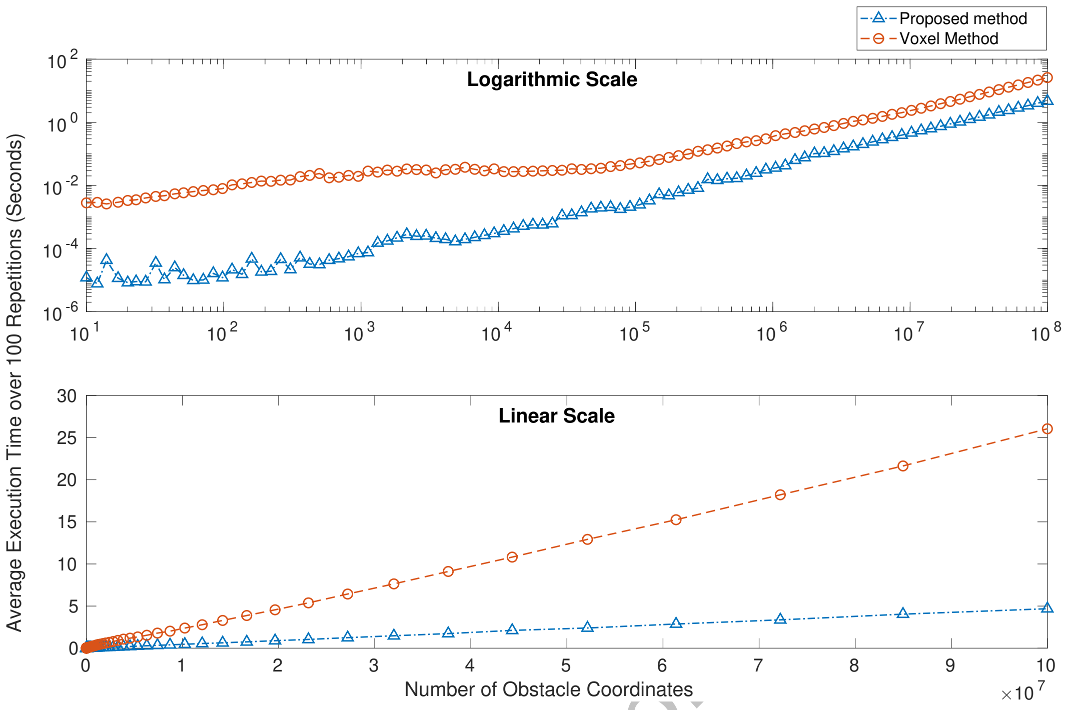

Figure 4.5: Comparison of computational performance of proposed PCOC against voxel based LoS detection technique.

4.2.4 Applicability and Limitations

The suitability of PCOC remains with the WSN applications which require an *a-priori* spatial knowledge of sensor-device configurations. The concept of approximating indoor space by point cloud is intended to allow the user to control the level of details for modelling physical elements. We propose the use of PCOC primarily in collecting the LoS coverage model for device sensors configurations which can further be utilized to optimize network resources such as hardware cost, sensing coverage and localization accuracy. In order to demonstrate the above idea, we implemented PCOC for the indoor design pattern as shown in Figure 4.2, having DGS=BGS=OGS=0.1 within the $5 \times 5 \times 5\, unit^3$ cubic space. A threshold sensing range of $3\, units$ was chosen for CBLs. Figure 4.6 shows the resulting LoS for a single CDL at the center of design, connected by red LoS lines to cyan CBL circles.

The situation shown in Figure 4.7 presents a limitation of the PCOC for obstacles with sharp edges. While approximating such obstacles with point clouds, if only one OPC coordinate falls on either side of the LoS and rest remain invisible, the PCOC wrongly concludes a clear LoS.

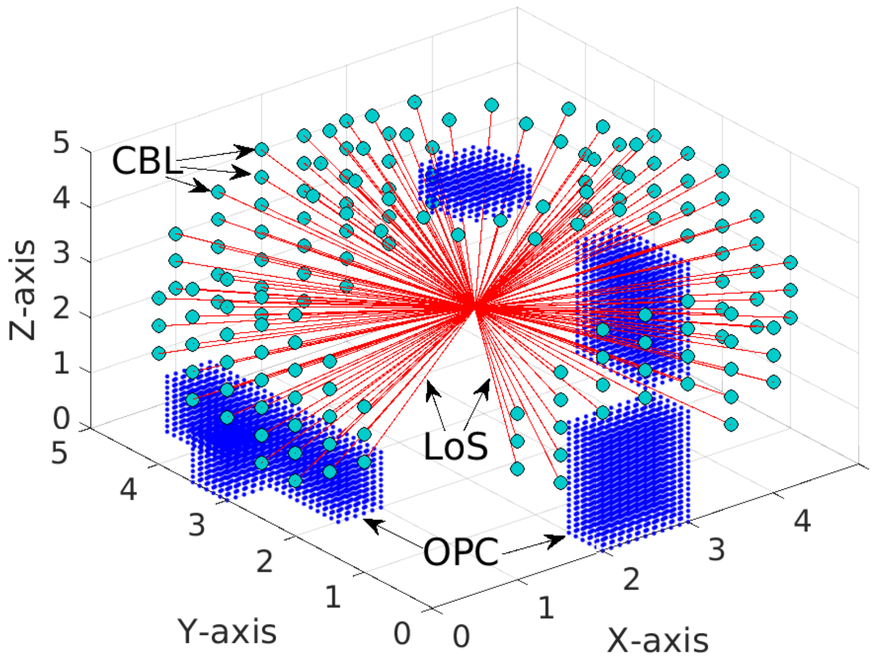

Figure 4.6: Visual representation of PCOC output for single CDL

For practical scenarios, such positioning of a single coordinate is highly unlikely. Yet, as a caveat to the application, use of sufficiently higher coordinate count i.e. a lower OGS is recommended for approximating obstacles.

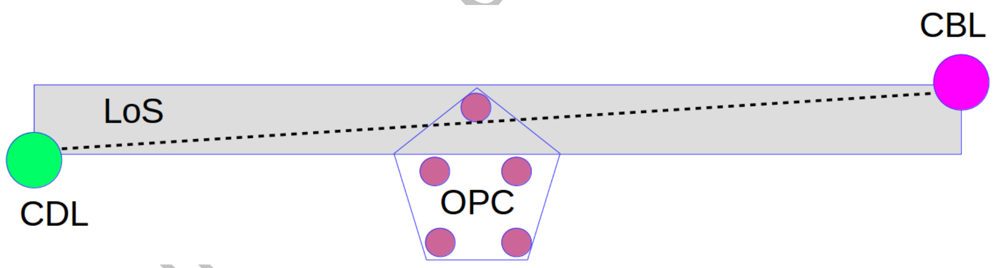

Figure 4.7: False LoS detection due to high OGS

Another limitation of the PCOC is the presence of an obstacle with concave inner surface in the CDL-CBL LoS. As shown in Figure 4.8, when projected according to Section 4.2.2, the method denies the possibility of a LoS in any of the projection planes, which otherwise exists. As a workaround to this issue, a carefully calculated minimum proximity distance between OPC and LoS can mark a sense of contact between them and infer correctly. Although, in practical modelling of sensor configurations for localization, the availability of LoS through concave objects should be disregarded due to its spatial dependency.

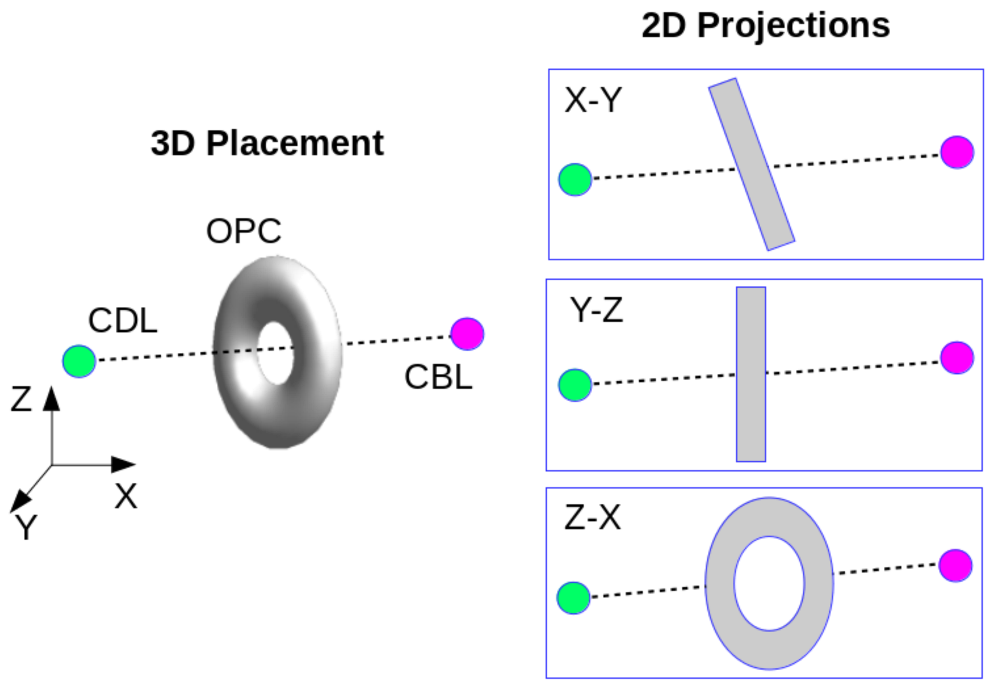

Figure 4.8: False deprecation of LoS due to concave obstacle

4.3 Assessment of Error Propagation

As introduced in Section 4.1, we next describe the basis of containing the propagation of error due to signal propagation losses. Among other measures of error propagation, Cramér-Rao lower bounds (CRLB) and Grometric Dilution of Precision (GDoP) have been frequently utilized in past with signal propagation modeling Li [2006]. Both of the metrics are derived from the variance-covariance matrix resulting from the least squares adjustment over input information [Dulman et al., 2008] and criticized for their effectiveness over different applications [Chaffee and Abel, 1994]. In addition to the observation noise, the derived expressions for both CRLB and GDoP contain direction cosines among the sensor device arrangement Yarlagadda et al. [2000]; Sharp et al. [2009]. This establishes the additional contribution of positioning geometry in error propagation which has been shown in Figure 4.9.

Due to its computational advantages Li [2006]; Patwari et al. [2005] over CRLB, GDoP is utilized as a metric for the assessment of error propagation throughout the scope of this thesis. GDoP is promptly used with Global Navigation Satellite Systems (GNSS) such as Global Positioning System (GPS), to assess the achievable positioning accuracy from the satellite configura-

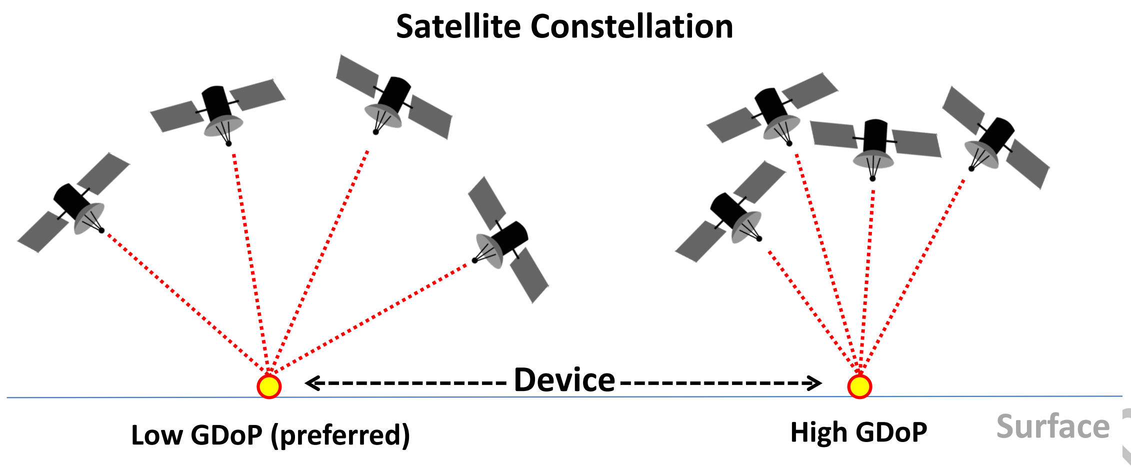

Figure 4.9: A visual representation of the effect of satellite geometry over the resulting GDoP via error propagation. An evenly spread satellite configuration (left) is preferred over a congested arrangement (right).

tion visible to a sensor at any time. For the ease of understanding, the simplest GDoP formulation results as a ratio of output positioning error to estimated range error in input observation, with the assumption of zero mean white Gaussian distribution of range observations with equal variance. This is shown in Equation 4.1.

$$GDoP(g) = \frac{\sqrt{\sigma_x^2 + \sigma_y^2 + \sigma_z^2}}{\sigma_r} \tag{4.1}$$

Here, σ_r is the standard deviation in range measurement while $\sigma_x, \sigma_y, \sigma_z$ represent positional deviations in the three axes respectively. A value of $1 \leq g \leq 5$ is assumed ideal for GDoP with 3D positioning Yarlagadda et al. [2000]; Sharp et al. [2009].

With Localization in WSN, use of GDoP is a prominent relative measure of estimating error propagation due to the noisy RSS measurements and CBL-CDL geometry. For the current implementation, the range estimates are assumed to be mutually independent with zero mean Gaussian noise. A combinatorial strategy for using GDoP as a constraint over the point clouds of CDLs and CBLs will be detailed further in the next chapter. Selecting low GDoP CDL-CBL geometry with maximum LoS as calculated from PCOC is the primary step for present MOO approach.

4.4 Optimization Objectives

As reviewed in Chapter 2, minimizing beacon deployment cost, maximizing the coverage of device connectivity and maximizing the accuracy of localization are primary objectives while optimizing a BPP. We elaborate on these objectives to pave the way for our choice and method of optimization.

1. **Cost**: In relation to BPP, cost refers to the number of beacons required for a localization setup. For optimization, it is utilized as a binary valued vector of decision variables indicating 1 as selection and 0 as rejection of particular beacon identified with respective index.

2. **Coverage**: Achieving a sufficient coverage of localization is in itself a threefold objective ensuring *sufficiency* and *reachability* of beacons within *LoS* to a device. *Sufficiency* is associated to the beacon density (k) as discussed in Chapter 3 and implemented as a lower bound constraint. On the other hand, *reachability* and *LoS* are formulated by estimating spatial adjacency between the nodes and included in the optimization as a coefficient matrix for CDLs and CBLs. This will be detailed in the next chapter.

3. **Accuracy**: In its absolute sense, it is the deviation of the estimated location from the actual value. In relation to the BPP, a beacon configuration is evaluated for its localization performance over a chosen set of CDLs. The accuracy is typically reported as average absolute value or Root Mean Squared Error (RMSE). The scope of this thesis remains with reporting RMSE to evaluate multiple beacon configurations.

Intuitively, it is understandable that, minimizing the beacon count decreases high coverage probability which in-turn is bound to reflect on the accuracy. On the other hand, allowing high accuracy configurations supports customization of higher beacon density for each CDL, resulting in a higher cost. This cohesion among the cost, coverage and accuracy denies the clear dominance of a single objective function and makes it critical to assign appropriate role to constraints and objectives in optimization. With this motivation, the next chapter describes our space exhaustive multi-objective approach for point cloud representation of indoor environment.