Transmission Rate Maximization Under Power & Channel Limitations in Reliable VLP Systems

Prepared By

Student Name	Student ID
Ahmed Aly Gamal El-Din El-Ghannam	19015292
Ahmed Sherif Ahmed Mahmoud Ghanem	19015255
Hesham Haytham	19016850
Marwan Magdy Marzouk	19016614
Menna Mohamed Anwar Ramadan Mohamed	19016729
Yahia Walid Mohamad Eldakhakhny	19016891

Supervisor:	Dr. Mohammed Karmoose
Academic Year:	2023 – 2024
Department:	Communications and Electronics Engineering
Faculty:	Faculty of Engineering
University:	Alexandria University

Contents

Listings

1 Introduction

This report contains all the necessary mathematical deductions, equations, and explanations used to formulate the two optimization problems tackled in the project's presentation. The first and second problems were selected from the papers Localization Via Visible Light and Rate Maximization for Downlink Multiuser respectively.

CVX code, as well as both papers can be found in the project's Github repository.

2 Power

2.1 Understanding The Problem

The paper tackles two problems for power allocation of LEDs: one that approximates the maximum tolerable error in localization accuracy and represents it with the variable ϵ; and the other deals with robust power allocation where maximum tolerable error is represented in a more accurate way. The approximated problem was chosen because it was simpler to implement but by no means it was easy.

2.2 Objective Function

It is required to minimize the total power consumption of a VLP system while maintaining a certain level of accuracy. The objective function is as shown in equation 1

$$\text{minimize}_p \quad 1^T p \tag{1}$$

2.3 Constraints

The objective function is subject to multiple constraints depicted in equations 2, 3, and 4 respectively.

$$\text{subject to} \quad \text{trace}\{J^{-1}(p)\} \leq \epsilon \tag{2}$$

$$p_{\text{lb}} \leq p \leq p_{\text{ub}} \tag{3}$$

$$1^T p \leq P_T \tag{4}$$

Where p_{lb} and p_{ub} are the lower and upper bounds for electrical power consumption for each transmitter; P_T is the electrical power limit for all transmitters; $\text{trace}\{J^{-1}(p)\}$ is the trace of the inverse of the Fisher Information Matrix; and ϵ represents the maximum tolerable Cramér–Rao Lower Bound (CRLB) level for reliable and accurate localization.

2.4 Cramér–Rao Lower Bound

The Cramér-Rao Lower Bound (CRLB) is a fundamental result in the field of statistics and estimation theory, which provides a way of quantifying how much variance one can expect in an unbiased estimator if one is trying to infer a parameter from a set of data.

In the case of the proposed problem, the CRLB is defined as:

$$E\{\|\hat{l}_r - l_r\|^2\} \geq \text{trace}\{J^{-1}(p)\} \tag{5}$$

2.5 Fisher Information Matrix

The Fisher Information Matrix (FIM) denoted $J(p)$ is a fundamental concept in statistics and estimation theory, particularly in the context of maximum likelihood estimation. It measures the amount of information that an observable random variable carries about unknown parameters upon which the probability depends.

It is used to define the CRLB and is defined in the problem as:

$$J(p) = (I_3 \otimes p)^T \Gamma \tag{6}$$

Where p is the power vector, I_3 is an identity matrix, $\otimes$ is the Kronecker Product, and Γ is a matrix $\in \mathbb{R}^{3Lx3}$ that is unique to the proposed VLP system.

2.6 Kronecker Product

The Kronecker product is a matrix operation used to construct a large matrix from two smaller matrices. It works as follows:

1. If you have two matrices, A of size $m \times n$ and B of size $p \times q$, the Kronecker product $A \otimes B$ is a matrix of size $mp \times nq$.
2. To construct $A \otimes B$, you multiply each element a_{ij} of matrix A by the entire matrix B. This means replacing each element a_{ij} in A with the block $a_{ij}B$. Thus, each a_{ij} results in a $p \times q$ block within the resulting matrix.

2.7 Power Optimization using CVXPY

Code snippet 3 showcases the problem's solution on CVXPY.

```python
import cvxpy as cp
import numpy as np
import matplotlib.pyplot as plt
NL = 4
gamma = np.loadtxt("gamma_sys")
epsilon = np.arange(0.05, 0.76, 0.05)
iterations = len(epsilon)
minPowArray = np.empty(iterations)
allocatedPow_CRLB = np.empty(NL)
elecPow_up = 5
elecPow_low = 0.5
totalElecPow = 15
powConvEff = 1
p = cp.Variable(NL, nonneg=True)
def FisherInformationMatrix(p):
    p_matrix = cp.reshape(p, (NL, 1))  # reshape p to a column matrix
    return ( cp.kron(np.eye(3), p_matrix).T @ gamma )
```

Code Snippet 1: Power Optimization using CVXPY: Defined Parameters & Auxiliary Functions

```python
objective = cp.Minimize(cp.sum(p))
for i in range(iterations):
    constraints = [
        cp.trace(cp.inv_pos(FisherInformationMatrix(p))) <=
           epsilon[i],
        p >= 0,
        cp.sum(p) <= totalElecPow,
        p <= elecPow_up,
        p >= elecPow_low
    ]
    problem = cp.Problem(objective, constraints)
    problem.solve(solver=cp.SCS) # selected SCS as the solver
    minPowArray[i] = problem.value
    if epsilon[i] == 0.25:
        allocatedPow_CRLB = p.value
```

Code Snippet 2: Power Optimization using CVXPY: Problem Implementation

```python
# printing results
if problem.status == 'optimal':
    # print allocated electrical power @CRLB = 0.25
    print("Optimal Electrical Power Allocation:", allocatedPow_CRLB)

    # save power values in a text file to be used for rate
       optimization
    np.savetxt("kono_pawa_elec_w", allocatedPow_CRLB)

    # save power values in a text file to be used for rate
       optimization
    np.savetxt("kono_pawa_opt_w", allocatedPow_CRLB * powConvEff)

    # save power values in a text file to be used for rate
       optimization
    np.savetxt("kono_pawa_opt_dbm", 10 * np.log10(allocatedPow_CRLB *
       powConvEff * 1000))

    # plotting Min Power vs CRLB
    plt.plot(epsilon,minPowArray)
    plt.xlabel("Max CRLB")
    plt.ylabel("Min Power")
    plt.show()
else:
    print("yo, what the duck?!")
```

Code Snippet 3: Power Optimization using CVXPY: Printing Results

3 Rate

3.1 Transmitter Model

The transmitter is modeled using equation 7, where $\boldsymbol{z} \in \mathbb{R}^L$ and it represents the output of every transmitter. $\boldsymbol{w_k} \in \mathbb{R}^L$ and it represents the distribution of a symbol on the different transmitters. s_k is the symbol intended for user k.

$$\boldsymbol{z} = \sum_{k=1}^{K} \boldsymbol{w_k} s_k \tag{7}$$

3.2 Channel Model

Channel is modeled by the matrix $G \in \mathbb{R}^{L\times K}$ where the element $g_{l,k}$ represents the fading coefficient between transmitter number l and user number k. This matrix is highly dependent on the positions of the transmitters and receivers, as well as the properties of the transmitters. For our purposes, this matrix is randomly generated. Note that the k^{th} column in G is referred to as $\boldsymbol{g_k}$

3.3 Receiver Model

The receiver is modeled using equation 8 where y_k is the symbol received by the k^{th} user. The first term in equation 8 is the desired signal, the second term is the inter-user interference (IUI) and the third term is the noise seen by the k^{th} user.

$$y_k = \boldsymbol{g_k}^T \boldsymbol{w_k} s_k + \sum_{j=1, j\neq k}^{K} \boldsymbol{g_k}^T \boldsymbol{w_j} s_j + n_k \tag{8}$$

3.4 Rate Model

Here, we will derive the expression which will be used to represent the system rate, which is our objective function. The first equation of interest is Shannon's rate equation, where the Shannon rate serves as the upper bound for the rate received by the k^{th} user. And using the manipulations shown in equations 9 - . Note that the random variable T_k represents the sum of IUI and noise in the system.

$$\begin{aligned} C_k &\geq I(X_k; Y_k) = h(Y_k) - h(Y_k|X_k) && (9)\\ &= h(X_k + T_k) - h(T_k) && (10)\\ &\geq \frac{1}{2}\log\left(e^{2h(X_k)} + e^{2h(T_k)}\right) - h(T_k) && (11)\\ &= \frac{1}{2}\log\left(1 + \frac{e^{2h(X_k)}}{e^{2h(T_k)}}\right) && (12) \end{aligned}$$

Assuming that $s_k \in \{-1, 0, 1\}$, $h(X_k)$ can be expressed as shown in equation 13

$$h(X_k) = \log(2|\boldsymbol{g_k}^T \boldsymbol{w_k}|) \tag{13}$$

So equation 12 can be rewritten as

$$C_k \geq \frac{1}{2}\log\left(1 + \frac{4|\boldsymbol{g_k}^T \boldsymbol{w_k}|^2}{e^{2h(T_k)}}\right) \tag{14}$$

Note that $h(T_k)$ must be upper bounded by the differential entropy of a Gaussian random variable as shown in equation 15, where σ_{tk}^2 is the variance of T_k.

$$h(T_k) \leq \frac{1}{2}\log(2\pi e\sigma_{tk}^2) \tag{15}$$

By substituting from equation 15 in equation 14 we get the following expression

$$C_k \geq \frac{1}{2}\log\left(1+\frac{2|\boldsymbol{g_k^T w_k}|^2}{\pi e\left(\sum_{j=1,j\neq k}^{K}\frac{1}{3}|\boldsymbol{g_k^T w_j}|^2+\sigma_{n_k}^2\right)}\right) = R_k \tag{16}$$

Equation 16 describes the maximum rate possible for the k^{th} user (R_k), so the maximum transmission rate of the whole system is described using equation 17

$$R = \sum_{k=1}^{K} R_k = \sum_{k=1}^{K}\frac{1}{2}\log\left(1+\frac{2|\boldsymbol{g_k^T w_k}|^2}{\pi e\left(\frac{1}{3}\sum_{j\neq k}|\boldsymbol{g_k^T w_j}|^2+\sigma_{n_k}^2\right)}\right) \tag{17}$$

Equation 17 is the objective function of our optimization problem.

3.5 Constraints Model

The only constraint we are concerned with is that of power. Thus, the power constraint can be expressed as follows

$$\sum_{k=1}^{K}\boldsymbol{w_k} \leq \boldsymbol{p} \tag{18}$$

where $\boldsymbol{p} \in \mathbb{R}^L$, and each element in $\boldsymbol{p}$ represents the maximum power of each transmitter.

3.6 Problem Formulation

By using equations 17 and 18, one can write the optimization problem at hand as follows

$$\text{maximize}_{\boldsymbol{w_k}} \sum_{k=1}^{K}\frac{1}{2}\log\left(1+\frac{2|\boldsymbol{g_k^T w_k}|^2}{\pi e\left(\frac{1}{3}\sum_{j\neq k}|\boldsymbol{g_k^T w_j}|^2+\sigma_{n_k}^2\right)}\right) \tag{19}$$

$$\text{subject to} \sum_{k=1}^{K}\boldsymbol{w_k} \leq \boldsymbol{p} \tag{20}$$

Note that the problem is non-convex, but it can be approximated by the convex optimization problem

$$\text{maximize}_{\boldsymbol{w_k},\boldsymbol{u_k},\boldsymbol{v_k}} \sum_{k=1}^{K}\frac{1}{2}\log\left(1+\frac{2u_k}{\pi e}\right) \tag{21}$$

$$\text{subject to } \operatorname{diag}(\boldsymbol{G}^T\boldsymbol{W}) \geq \frac{\boldsymbol{u}}{2\boldsymbol{\theta}^{(j)}} + \frac{\boldsymbol{v}^2\boldsymbol{\theta}^{(j)}}{2} \tag{22}$$

$$\|[\frac{\boldsymbol{g_k^T w_1}}{\sqrt{3}}, \ldots.., \frac{\boldsymbol{g_k^T w_{k-1}}}{\sqrt{3}}, \frac{\boldsymbol{g_k^T w_{k+1}}}{\sqrt{3}}, \ldots.., \frac{\boldsymbol{g_k^T w_K}}{\sqrt{3}}, \sigma_{nk}]\| \leq v_k, k = 1,2,...,K \tag{23}$$

$$\sum_{k=1}^{K}\boldsymbol{w_k} \leq \boldsymbol{p} \tag{24}$$

The previous convex problem can be solved iteratively, with each iteration it more closely approximates the original non-convex problem, where j is the iteration index, and $\boldsymbol{\theta}^{(j)} = \frac{\sqrt{\boldsymbol{u}}}{\boldsymbol{v}}$. $\boldsymbol{u}$ and $\boldsymbol{v}$ are two vectors of length K and are auxiliary variables used to simplify solving the problem. $\boldsymbol{u}$ and $\boldsymbol{v}$ are described by the equations

$$|\boldsymbol{g_k}^T\boldsymbol{w_k}|^2 \geq u_k v_k^2 \tag{25}$$

$$\frac{1}{3}\sum_{j=1,j\neq k}^{K} |\boldsymbol{g_k}^T\boldsymbol{w_j}|^2 + \sigma_{nk}^2 \leq v_k^2 \tag{26}$$

3.7 Rate Optimization Using CVXPY

The following code shows the python code used to represent the problem and solve it.

```python
import cvxpy as cp
import numpy as np
import matplotlib.pyplot as plt

# Function used in constraint 2
def c2(M: cp.Variable, v) -> cp.Variable:
    M_res = M
    M_res = (M_res - cp.diag(cp.diag(M_res)))/np.sqrt(3) + cp.diag(v)
    return cp.norm(M_res, axis=1)

# Function to calculate a weighted moving average
def calculate_ema(values, alpha):
    ema = values[0]  # start with the first value

    for value in values[1:]:
        ema = alpha * value + (1 - alpha) * ema

    return ema


# Problem constants
K = 3 # Number of users
L = 9 # Number of lamps
G = np.loadtxt("G") # Describe the channel
p = np.array([7.74903481e-07, 1.44155904e+01, 6.11280099e+00,
   3.00493992e+00, 1.81572075e-06, 6.07339084e+00, 1.07330140e-06,
   1.63758571e+01, 5.73877204e+00])

# sigma_max = 10**-7
sigma_n = np.ones(K)*10**-11

# Initial guess for theta
theta = np.ones(K)
iterations = 500
opt_rate = []
```

Code Snippet 4: Rate Optimization using CVXPY: Parameters and Functions

```python
# Iterate over the problem
for j in range(iterations):
    print(j)
    # Problem variables
    w = cp.Variable((L, K))
    u = cp.Variable(K)
    v = cp.Variable(K)

    # Problem statement
    objective = cp.Maximize(cp.sum(0.5*cp.log(1+2*u/(np.pi*np.e))))

    # g.T * w will be used a lot in the constraints so it is easier
       to define it here
    M = G.T @ w


    constraints = [
        cp.diag(M) >= u/(2*theta) + (0.5 * cp.multiply(v**2, theta)),
        c2(M, sigma_n) <= v,
        cp.sum(w, axis=1) <= p
    ]

    problem = cp.Problem(objective, constraints)
    try:
        problem.solve(solver=cp.SCS) #SCS
        # update value of theta
        uj = u.value
        vj = v.value
        theta = np.sqrt(uj)/vj

        # add this rate to opt_rate
        opt_rate.append(problem.value)
    except:
        break
# End of loop
```

Code Snippet 5: Rate Optimization using CVXPY: Iterating Over The Problem

```python
# Algorithm output
print("final optimal rate = ", opt_rate[-1])
# Plot output
n = np.arange(1, len(opt_rate)+1)
if len(n) > 20:
    np.savetxt("G", G)


plt.plot(n,opt_rate)
plt.xlabel("Iteration number")
plt.ylabel("Optimal rate")
plt.show()

# Take EMA and graph it with alpha
ema_values = []
alpha_values = np.arange(0, 1.1, 0.1)

for alpha in alpha_values:
    ema_values.append(calculate_ema(opt_rate, alpha))

plt.plot(alpha_values, ema_values)
plt.xlabel("Alpha")
plt.ylabel("EMA Rate")
plt.show()
```
Code Snippet 6: Rate Optimization using CVXPY: Printing Results