

# Couverture Maximale - Positionnement de Caméras de Surveillance

---

## Projet de Recherche Opérationnelle - GL3 INSAT

### Introduction

Le **Problème de Couverture Maximale (Maximal Covering Location Problem - MCLP)** est un problème classique NP-difficile de recherche opérationnelle. Ce projet applique la **Programmation Linéaire en Nombres Entiers (PLNE)** avec le solveur **Gurobi** pour optimiser le positionnement de caméras de surveillance.

#### Objectifs:

- Maximiser la couverture pondérée des zones (priorité  $\times$  population)
  - Respecter les contraintes de budget et nombre de caméras
  - Encourager la redondance pour zones critiques (bonus 10%)
  - Éviter l'installation de caméras inutiles
- 

### Modélisation Mathématique

#### Ensembles et Paramètres

##### Ensembles:

- **I**: Emplacements potentiels de caméras
- **J**: Zones à surveiller

##### Paramètres:

- **c<sub>i</sub>**: Coût de la caméra i (€)
- **r<sub>i</sub>**: Portée de la caméra i (mètres)
- **p<sub>j</sub>**: Priorité de la zone j (1-10, où 7-10 = critique)
- **w<sub>j</sub>**: Population de la zone j
- **B**: Budget maximal (€)
- **K**: Nombre maximal de caméras
- **a<sub>ij</sub>**: Matrice de couverture ( $a_{ij} = 1$  si  $\text{distance}(i,j) \leq r_i$ , sinon 0)

#### Variables de Décision

- **x<sub>i</sub>**  $\in \{0,1\}$ : 1 si une caméra est installée à l'emplacement i, 0 sinon
- **y<sub>j</sub>**  $\in \{0,1\}$ : 1 si la zone j est couverte, 0 sinon

#### Variables de Décision

- **x<sub>i</sub>**  $\in \{0,1\}$ : Installation de caméra (1 = installée, 0 = non installée)
- **y<sub>j</sub>**  $\in \{0,1\}$ : Couverture de zone (1 = couverte, 0 = non couverte)



## Fonction Objectif

### Maximiser:

$$Z = \sum_{j \in J} p_j \times w_j \times y_j + 0.1 \times \sum_{j \in J, p_j \geq 7} \sum_{i \in I} p_j \times w_j \times a_{ij} \times x_i$$

### Composantes:

1. Couverture pondérée des zones (priorité × population)
2. Bonus de redondance (10%) pour zones critiques couvertes par plusieurs caméras

## Contraintes

### Modèle complet:

Maximiser:  $Z = \sum_{j \in J} p_j \times w_j \times y_j + 0.1 \times \sum_{j, p_j \geq 7} \sum_{i} p_j \times w_j \times a_{ij} \times x_i$

Sous contraintes:

(C1)  $\sum_{i \in I} c_i \times x_i \leq B$  [Budget]

(C2)  $\sum_{i \in I} x_i \leq K$  [Nombre de caméras]

## Matrice de Couverture

### Calcul de $a_{ij}$ :

$$a_{ij} = 1 \quad \text{si} \quad \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq r_i$$
$$a_{ij} = 0 \quad \text{sinon}$$

Une caméra  $i$  peut couvrir une zone  $j$  si la distance euclidienne entre elles est inférieure ou égale à la portée de la caméra.

## Complexité

- **Type:** PLNE (Programmation Linéaire en Nombres Entiers)
- **Classe:** NP-difficile
- **Exemple:** 20 zones × 15 caméras =  $2^{15} = 32,768$  combinaisons possibles
- **Solveur:** Gurobi (Branch-and-Bound, coupes, heuristiques, présolve)
- **Description:** Type de zone (commerciale, résidentielle, etc.)

## Caméras



## Structure du Projet

```
MaximalCoveringLocationProblem/  
├── main.py                # Point d'entrée  
├── requirements.txt       # Dépendances  
├── src/  
│   ├── optimization_model.py # Modèle Gurobi (450 lignes)  
│   ├── main_window.py      # Interface PyQt5 (710 lignes)  
│   └── visualization.py     # Visualisations (400 lignes)  
└── data/  
    └── example_data.json    # Données d'exemple
```

## Technologies

- **Python 3.8+**: Langage principal
- **Gurobi 10.0+**: Solveur PLNE (licence académique gratuite)
- **PyQt5 5.15+**: Interface graphique avec threading (QThread)
- **Matplotlib 3.5+**: Visualisations (carte couverture, heatmap, statistiques)
- **NumPy 1.21+**: Calcul de la matrice de couverture

## Installation et Utilisation

```
# Installation  
pip install -r requirements.txt  
  
# Lancer l'application  
python main.py
```

**Note:** Installer Gurobi et obtenir une licence académique gratuite sur <https://www.gurobi.com/>

## Workflow de l'Interface

L'application PyQt5 comporte **3 onglets**:

1. **Configuration**: Saisie des données (zones, caméras, budget), génération aléatoire, import/export JSON
2. **Résolution**: Paramètres du solveur, lancement optimisation (thread non-bloquant), logs temps réel
3. **Résultats**: Résumé solution, 3 types de visualisations, export rapports
  - Configurer les paramètres du solveur (temps limite, gap)
  - Lancer l'optimisation (thread non-bloquant)
  - Observer le journal d'exécution en temps réel
4. **Résultats et Visualisation** (Onglet 3)
  - Consulter le résumé de la solution
  - Visualiser la carte de couverture



- Afficher la heatmap d'intensité
- Analyser les statistiques détaillées
- Exporter la solution (JSON ou rapport TXT)

## Visualisations

L'application offre plusieurs types de visualisations:

### 1. Carte de Couverture

- Zones couvertes (vert) vs non couvertes (rouge)
- Caméras installées avec cercles de portée
- Priorités représentées par intensité de couleur
- Annotations pour zones critiques

### 2. Heatmap d'Intensité

- Intensité de couverture en chaque point

---

## Données du Problème

Format JSON (data/example\_data.json)

```
{
  "max_cameras": 50,
  "max_budget": 1000000,
  "zones": [[x, y, priorité, population, "description"], ...],
  "cameras": [[x, y, coût, portée, angle, "type"], ...]
}
```

### Attributs des Zones

- **Position (x, y):** Coordonnées géographiques
- **Priorité (1-10):** 1-3 faible, 4-6 moyenne, 7-10 critique
- **Population:** Densité/nombre de personnes
- **Description:** Type de zone

### Attributs des Caméras

- **Position (x, y):** Emplacement potentiel
- **Coût (€):** Coût d'installation
- **Portée (m):** Distance maximale de surveillance
- **Angle (°):** 90°, 180°, 270°, 360°
- **Type:** Fixe, PTZ (Pan-Tilt-Zoom), Thermique

---

## Résultats et Interprétation



## Solution Optimale

### Variables de sortie:

- **$x_i$**  = 1: Caméra installée à l'emplacement  $i$
- **$y_j$**  = 1: Zone  $j$  couverte
- **Z**: Valeur de la couverture pondérée totale (plus Z est élevé, meilleure est la couverture)

### Métriques de Performance

1. **Taux de couverture**:  $(\text{Zones couvertes} / \text{Total zones}) \times 100\%$
2. **Utilisation du budget**:  $(\text{Coût total} / \text{Budget max}) \times 100\%$
3. **Redondance moyenne**: Nombre moyen de caméras par zone couverte
4. **Efficacité**: Zones couvertes par euro dépensé

### Exemple de Résultats

Configuration: 20 zones, 15 emplacements, budget 50K, max 10 caméras

### Solution obtenue:

- Fonction objectif: 41100.0
- 10/15 caméras installées (6 PTZ, 2 thermiques, 2 fixes)
- 11/20 zones couvertes (55%)
- Coût Total: 39500 (79.0% du budget)
- Temps de résolution: 0.01s

### Visualisations

1. **Carte de Couverture**: Zones (vert/rouge), caméras avec cercles de portée
2. **Heatmap**: Intensité de couverture, zones de redondance
3. **Statistiques**: Distribution types, taux couverture, coûts, redondance par priorité

---

## Validation et Tests

### Tests de Cohérence

- Budget insuffisant  $\rightarrow$  aucune caméra installée
- Budget/K suffisants  $\rightarrow$  toutes zones couvertes
- Zones isolées (hors portée)  $\rightarrow y_j = 0$
- Caméras inutiles  $\rightarrow x_i = 0$  (contrainte C4)

### Analyse de Sensibilité

Impact de la variation de:

- Budget B (contrainte C1)
- Nombre max K (contrainte C2)
- Priorités  $p_j$  (fonction objectif)
- Portées  $r_i$  (matrice de couverture  $a_{ij}$ )