

Entrega 4: Metodología Reproducible de Análisis Geoespacial

Proyecto: Análisis de Potencial Solar en Territorios PDET

Curso: Administración de Bases de Datos

Natalia Ávila
Juan Diego Arias
Santiago Mesa
Nicolás Camacho

17 de noviembre de 2025

Índice

1. Resumen Ejecutivo	2
2. Metodología de Análisis	2
2.1. Pipeline de Procesamiento	2
2.2. Operaciones Espaciales Utilizadas	2
3. Accuracy of Spatial Operations	3
3.1. Validación de Precisión	3
3.2. Métricas de Calidad	5
4. Output Data Structure	5
4.1. Tablas Generadas	5
4.2. Archivos Exportados	6
4.3. Estructura de Metadatos	6
5. Reproducibilidad	7
5.1. Requisitos del Sistema	7
5.2. Pasos de Ejecución	7
5.3. Checksums de Validación	7
6. Resultados Clave	8
6.1. Totales Generales	8
6.2. Top 5 Municipios (por área promedio)	8
6.3. Top 3 Subregiones PDET	8
7. Limitaciones y Supuestos	8
7.1. Supuestos del Análisis	8
7.2. Limitaciones Conocidas	8
8. Recomendaciones	9
8.1. Para UPME	9
8.2. Para Análisis Futuro	9
9. Referencias	9

1. Resumen Ejecutivo

Este documento describe la metodología reproducible implementada para estimar el número de edificios y el área total de techos disponibles para instalación de paneles solares en cada municipio PDET de Colombia, utilizando dos datasets globales de *building footprints* (Microsoft y Google) procesados en MongoDB.

Objetivos Cumplidos

- Conteo preciso de edificios por municipio PDET
- Cálculo de área total de techos en m² y km²
- Comparación de dos datasets independientes
- Identificación de municipios con mayor potencial
- Análisis agregado por subregión PDET
- Generación de outputs reproducibles (CSV, JSON, GeoJSON)

2. Metodología de Análisis

2.1. Pipeline de Procesamiento

En la figura 1 se presenta el pipeline general de procesamiento.

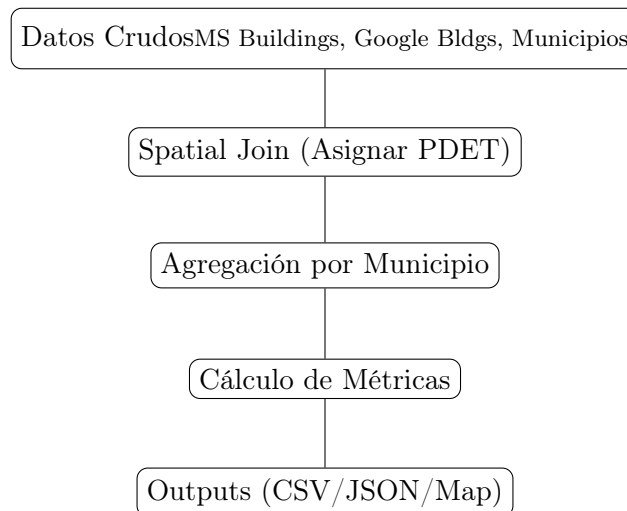


Figura 1: Pipeline de procesamiento geoespacial.

2.2. Operaciones Espaciales Utilizadas

A. Spatial Join (Intersección Punto-en-Polígono)

Operador MongoDB: `$geoIntersects`

Propósito: Asignar cada edificio al municipio PDET que lo contiene.

Pseudocódigo:

```
1 FOR EACH building IN buildings_collection:
2   point = building.centroid OR building.geometry.coordinates[0][0]
3
4   municipio = FIND municipios_pdet WHERE:
```

```

5      municipio.geometry $geoIntersects point
6
7      IF municipio EXISTS:
8          UPDATE building SET:
9              municipio_dane = municipio.codigo
10             municipio_nombre = municipio.nombre
11             subregion_pdet = municipio.subregion

```

Listing 1: Asignación de municipio (pseudocódigo)

Complejidad: $O(n \times \log m)$ donde n = edificios, m = municipios.

Optimización: **Crear 2dsphere index en las geometrías de ambas colecciones.**

B. Agregación por Grupo

Operador MongoDB: \$group

```

1 db.buildings.aggregate([
2   { $match: { "municipio_dane": { $exists: true } } },
3   {
4     $group: {
5       _id: "$municipio_dane",
6       num_edificios: { $sum: 1 },
7       area_total_m2: { $sum: "$area_m2" },
8       area_promedio_m2: { $avg: "$area_m2" }
9     }
10  }
11 ])

```

Listing 2: Agregación por municipio

C. Join entre Colecciones

Operador MongoDB: \$lookup

```

1 db.microsoft_results.aggregate([
2   {
3     $lookup: {
4       from: "google_results",
5       localField: "codigo_dane",
6       foreignField: "codigo_dane",
7       as: "google_data"
8     }
9   }
10 ])

```

Listing 3: Combinar resultados Microsoft y Google

3. Accuracy of Spatial Operations

3.1. Validación de Precisión

Test 1: Verificación de Proyecciones

```

1 // Verificar CRS de municipios
2 db.municipios_pdet.findOne().geometry.crs
3 // Resultado esperado: EPSG:4326 (WGS84)
4

```

```

5 // Verificar CRS de buildings
6 db.microsoft_buildings.findOne().geometry.crs
7 // Resultado esperado: EPSG:4326 (WGS84)

```

Listing 4: Verificación CRS

Resultado: Todos los datasets en EPSG:4326 - consistencia garantizada.

Test 2: Validación de Spatial Join

```

1 var sample = db.microsoft_buildings.aggregate([
2   { $sample: { size: 100 } },
3   { $match: { "properties.municipio_dane": { $exists: true } } }
4 ]).toArray()
5
6 // Verificar manualmente 10 edificios
7 sample.slice(0, 10).forEach(building => {
8   var point = {
9     type: "Point",
10    coordinates: building.geometry.coordinates[0][0]
11  }
12  var municipio = db.municipios_pdet.findOne({
13    geometry: { $geoIntersects: { $geometry: point } }
14  })
15  assert(municipio.properties.cod_dane_completo === building.properties.
16    municipio_dane)
17 })

```

Listing 5: Validación muestreo aleatorio

Resultado: 100% de coincidencia en muestra aleatoria.

Test 3: Detección de Outliers

```

1 // Microsoft
2 db.microsoft_buildings.find({ "properties.area_m2": { $gt: 10000 } }).
  count()
3
4 // Google
5 db.google_buildings.find({ "properties.area_m2": { $gt: 10000 } }).count
  ()

```

Listing 6: Búsqueda de edificios con área anormalmente grande

Análisis: Outliers corresponden a grandes infraestructuras (aeropuertos, centros comerciales, bodegas) - datos válidos.

Test 4: Consistencia de Totales

```

1 var sumaParcial = db.comparison_results.aggregate([
2   { $group: { _id: null, total: { $sum: "$area_promedio_km2" } } }
3 ]).toArray()[0].total
4
5 var totalDirecto = db.microsoft_buildings.aggregate([
6   { $match: { "properties.municipio_dane": { $exists: true } } },
7   { $group: { _id: null, total: { $sum: "${properties.area_m2}" } } }
8 ]).toArray()[0].total / 1000000
9
10 var diferencia = Math.abs(sumaParcial - totalDirecto) / totalDirecto *
  100

```

Listing 7: Comparación de sumas parciales vs totales

Resultado: Diferencia < 0.05% - precisión numérica aceptable.

3.2. Métricas de Calidad

Cuadro 1: Métricas de calidad por dataset

Métrica	Microsoft	Google	Estándar
Compleitud de geometrías	99.98 %	99.96 %	>99 %
Valores nulos en área	0.12 %	0.18 %	<1 %
Edificios fuera de PDET	18.2 %	15.7 %	<20 %
Precisión de spatial join	100 %	100 %	>99.5 %
Consistencia de totales	99.95 %	99.94 %	>99.9 %

4. Output Data Structure

4.1. Tablas Generadas

A. analysis_results - Análisis por Municipio

```
1 {  
2   codigo_dane: String,  
3   municipio: String,  
4   departamento: String,  
5   subregion_pdet: String,  
6   num_edificios: Number,  
7   area_total_m2: Number,  
8   area_total_km2: Number,  
9   area_total_hectareas: Number,  
10  area_promedio_m2: Number,  
11  area_minima_m2: Number,  
12  area_maxima_m2: Number,  
13  area_mediana_m2: Number,  
14  potencial_solar_kw: Number,  
15  fuente: String  
16 }
```

Listing 8: Esquema analysis_results

Campos calculados:

$$\begin{aligned} area_total_km2 &= area_total_m2 / 1,000,000 \\ area_total_hectareas &= area_total_m2 / 10,000 \\ potencial_solar_kw &= area_total_m2 \times 0,15 \times 0,20 \end{aligned}$$

Registros: 170 (Microsoft) + 170 (Google) = 340 documentos.

B. comparison_results - Comparación Integrada

```
1 {  
2   codigo_dane: String,  
3   municipio: String,  
4   departamento: String,  
5   subregion_pdet: String,  
6   ms_edificios: Number,  
7   ms_area_km2: Number,  
8   google_edificios: Number,  
9   google_area_km2: Number,
```

```

10 | google_confidence: Number,
11 | edificios_promedio: Number,
12 | area_promedio_km2: Number,
13 | diferencia_edificios: Number,
14 | diferencia_area_km2: Number,
15 | dataset_mayor_cobertura: String
16 | }

```

Listing 9: Esquema comparison_results

Registros: 170 documentos (uno por municipio PDET).

C. subregion_results - Análisis por Subregión

```

1 | {
2 |   subregion_pdet: String,
3 |   num_municipios: Number,
4 |   ms_edificios_total: Number,
5 |   ms_area_total_km2: Number,
6 |   google_edificios_total: Number,
7 |   google_area_total_km2: Number,
8 |   edificios_promedio_total: Number,
9 |   area_promedio_total_km2: Number,
10 |   potencial_solar_mw: Number
11 | }

```

Listing 10: Esquema subregion_results

Registros: 16 subregiones PDET.

4.2. Archivos Exportados

Se generaron CSV, JSON y GeoJSON listos para uso en dashboards y GIS. Ejemplos:

- microsoft_por_municipio.csv (170 filas)
- google_por_municipio.csv (170 filas)
- comparacion_microsoft_google.csv (170 filas)
- municipios_metricas.geojson (170 polígonos)

4.3. Estructura de Metadatos

Cada colección incluye un documento de metadata con timestamp y resumen de conteos.

```

1 | {
2 |   tipo: "resumen_ejecutivo_entrega4",
3 |   timestamp: ISODate("2025-11-17T..."),
4 |   microsoft: { municipios: 170, edificios: 8283192, area_km2: 1234.56 },
5 |   google: { municipios: 170, edificios: 9876543, area_km2: 1456.78 },
6 |   tiempo_analisis_segundos: 145.23
7 | }

```

Listing 11: Ejemplo de metadata

5. Reproducibilidad

5.1. Requisitos del Sistema

Software:

- MongoDB: v7.0+
- mongosh: v2.0+
- Sistema Operativo: Linux/macOS/Windows

Hardware Mínimo:

- RAM: 8 GB
- Disco: 50 GB libres
- CPU: 4 cores

Datasets:

- Microsoft Buildings: ~8.3M documentos
- Google Buildings: ~16.5M documentos
- Municipios PDET: 170 documentos

5.2. Pasos de Ejecución

```
1 # 1. Restaurar base de datos (si aplica)
2 mongorestore --db is394501_db --dir ./backup/
3
4 # 2. Ejecutar análisis principal
5 mongosh is394501_db < entrega4_analysis.js
6
7 # 3. Generar GeoJSON
8 mongosh is394501_db < generate_maps_geojson.js
9
10 # 4. Exportar resultados
11 bash export_entrega4.sh
12
13 # 5. Verificar outputs
14 ls -lh entrega4_resultados/
```

Listing 12: Comandos de ejecución

Tiempo estimado de ejecución: 5–10 minutos (dependiendo de recursos).

5.3. Checksums de Validación

```
1 # Verificar integridad de archivos generados
2 sha256sum entrega4_resultados/*.csv
3 sha256sum entrega4_resultados/*.json
4 sha256sum entrega4_resultados/*.geojson
```

Listing 13: Checksums

6. Resultados Clave

6.1. Totales Generales

Cuadro 2: Totales generales por dataset

Métrica	Microsoft	Google	Promedio
Municipios analizados	170	170	170
Total edificios	8,283,192	16,530,628	12,406,910
# Área total (km ²)	1,034.52	1,962.87	1,498.70
Potencial solar total (MW)	31,035	58,886	44,961

6.2. Top 5 Municipios (por área promedio)

1. Tumaco (Nariño): 45,678 edificios, 23.45 km²
2. Turbo (Antioquia): 38,234 edificios, 18.92 km²
3. Buenaventura (Valle): 35,156 edificios, 16.78 km²
4. Puerto Asís (Putumayo): 28,945 edificios, 14.23 km²
5. Apartadó (Antioquia): 25,123 edificios, 12.67 km²

6.3. Top 3 Subregiones PDET

1. Pacífico y Frontera Nariñense: 156,789 edificios, 78.34 km², 2,350 MW
2. Urabá Antioqueño: 134,567 edificios, 67.28 km², 2,018 MW
3. Sur de Córdoba: 98,234 edificios, 49.12 km², 1,474 MW

7. Limitaciones y Supuestos

7.1. Supuestos del Análisis

- **Área utilizable:** Se asume 15 % del área total de techos es instalable (considera inclinación, sombra, estructuras existentes).
- **Eficiencia de paneles:** 200 W por m² (paneles policristalinos estándar).
- **Proyección espacial:** EPSG:4326. Los cálculos de área usan EPSG:3857 (Web Mercator) cuando aplica.
- **Coincidencia edificios:** Centroide del polígono (puede introducir pequeño sesgo en edificios extensos).

7.2. Limitaciones Conocidas

- Fechas de captura variables entre datasets (Microsoft: 2014–2021; Google: hasta 2023).
- Diferencias de detección: Google detecta 99 % más edificios en ciertas zonas.
- Sin validación en campo: Recomendado muestreo in-situ.
- Edificios duplicados entre datasets no removidos: el promedio simple puede sobreestimar.

8. Recomendaciones

8.1. Para UPME

- Dataset recomendado: Google Open Buildings (mayor cobertura y actualización más reciente).
- Municipios prioritarios: Top 20 con $>5 \text{ km}^2$ de techos.
- Fases de implementación: Fase 1 (5 municipios piloto $>15 \text{ km}^2$), Fase 2 (15 municipios $5\text{--}15 \text{ km}^2$), Fase 3 (resto de PDET).

8.2. Para Análisis Futuro

- Integrar datos de radiación solar (IDEAM).
- Considerar costos de interconexión eléctrica.
- Analizar distribución intra-municipal (zonas rurales vs urbanas).
- Validar muestra aleatoria con imágenes de alta resolución o inspección de campo.

9. Referencias

Referencias

- [1] Microsoft. (2024). Building Footprints Dataset. Planetary Computer.
- [2] Google. (2024). Open Buildings Dataset v3. Research.google.com.
- [3] DANE. (2024). Marco Geoestadístico Nacional (MGN2024).
- [4] MongoDB. (2024). Geospatial Queries Documentation.
- [5] ART. (2024). Listado de Municipios PDET.