CEA Comité Español de Automática

XLV JORNADAS AUTOMÁTICA MÁLAGA 2024

UNIVERSIDADE DA CORUÑA

Jornadas de Automática

Modelos a gran escala para mapeo semántico en robótica móvil

Moncada-Ramirez, J.*, Ruiz-Sarmiento, J.R., Matez-Bandera, J.L., Gonzalez-Jimenez, J.

Grupo de Percepción Artificial y Robótica Inteligente (MAPIR), Dept. de Ingeniería de Sistemas y Automática, Instituto Universitario en Ingeniería Mecatrónica y Sistemas Ciberfísicos (IMECH.UMA), Universidad de Málaga, Blvr. Louis Pasteur, 35, 29071 Málaga, España.

To cite this article: Moncada-Ramirez, J., Ruiz-Sarmiento, J.R., Matez-Bandera, J.L., Gonzalez-Jimenez, J. 2024. Large models for semantic mapping in mobile robotics. Jornadas de Automática, 45. https://doi.org/10.17979/ja-cea.2024.45.10940

Resumen

La aparición de los modelos a gran escala permite abordar algunas de las principales limitaciones que presentan las técnicas de mapeo semántico tradicional en robótica móvil. Sin embargo, estos modelos son propensos a generar respuestas incorrectas, incoherentes o incluso inventadas, pudiendo ocasionar comportamientos erróneos del robot. Para poder desplegarse en aplicaciones reales, por tanto, es crucial desarrollar mecanismos que permitan mitigar estas errores. En este trabajo se utiliza *ConceptGraphs*, un método del estado del arte basado en modelos a gran escala para construir mapas semánticos, sobre el que se plantean dos estrategias para reducir las respuestas erróneas. Primero, se propone adaptar el método para operar con modelos más recientes (por ejemplo, Gemini 1.5 y ChatGPT-4o). En segundo lugar, se incorpora una etapa de refinamiento de respuestas mediante la técnica denominada *Reflexión*, que permite al modelo autoevaluar y mejorar sus propias respuestas. Finalmente, se validan las propuestas mediante experimentos en entornos reales del conjunto de datos ScanNet.

Palabras clave: Robótica inteligente, Aprendizaje automático, Robots móviles autónomos, Construcción de mapas, Percepción y sensado

Large models for semantic mapping in mobile robotics

Abstract

The advent of large models enables us to overcome some of the main limitations of traditional semantic mapping techniques in mobile robotics. However, such models can produce incorrect, incoherent, or made-up responses, leading to undesirable robot behaviors. Consequently, providing mechanisms to mitigate such responses is crucial before these models can be deployed in real-world applications. In this work, we build upon ConceptGraphs, a state-of-the-art method based on large models to create semantic maps, on which we propose two approaches to reduce misleading responses. First, we propose to adapt the method to operate on more recent models (e.g., Gemini 1.5 and ChatGPT-4o). Second, we incorporate a refinement stage for responses using the Reflection technique, which allows the model to self-evaluate and improve its responses. Finally, we validate the proposals through experiments in real-world environments from the ScanNet dataset.

Keywords: Intelligent robotics, Machine Learning, Autonomous Mobile Robots, Map building, Perception and sensing

1. Introducción

Los robots móviles están siendo empleados en aplicaciones cada vez más diversas en campos como la asistencia en el hogar, la industria, la sanidad o la educación (Rubio et al., 2019). Común a estas aplicaciones es la necesidad de que el robot cuente con las capacidades cognitivas necesarias para interpretar su entorno y razonar sobre el mismo, requisito fundamental para poder desenvolverse y completar sus tareas con éxito. Una estrategia comúnmente empleada para alcanzar dichas capacidades es la utilización de mapas semánticos (Ruiz-

*Autor para correspondencia: jemonra@uma.es

Sarmiento et al., 2017a), modelos del entorno de trabajo donde se representan tanto los elementos del mismo (objetos, habitaciones, etc.) como su semántica (propiedades, funcionalidades, relaciones, etc.). Por ejemplo, empleando un mapa semántico, un robot podría gestionar un inventario organizando productos de limpieza, de baño, artículos de oficina, etc., teniendo en cuenta sus propiedades (cantidad restante, componentes, propiedades químicas, caducidades), sus funcionalidades, y relaciones. Estas relaciones podrían ser sus ubicaciones habituales (armarios, estantes, cajones) e incompatibilidades, permitiendo mantener las medidas de seguridad necesarias y emitiendo alertas en situaciones como violaciones de dichas medidas, productos agotándose, etc.

Tradicionalmente, el flujo de trabajo o *pipeline* de las técnicas de mapeo semántico incluye la utilización de un método de detección de objetos para identificar los elementos del entorno e incluirlos en el mapa (Chaves et al., 2019; Fernandez-Chaves et al., 2021). Estos detectores suelen ajustarse empleando un cierto conjunto de datos en su etapa de entrenamiento (Ruiz-Sarmiento et al., 2017b), por lo que sólo serán capaces de detectar un conjunto cerrado de categorías de objetos fijado por dicho repositorio. Por ejemplo, el conjunto de datos Microsoft COCO (Lin et al., 2014) es ampliamente utilizado para el entrenamiento de detectores de objetos, como son las populares series YOLO desde su versión 3 (Terven and Cordova-Esparza, 2023), o Mask R-CNN (He et al., 2017), cuya reciente implementación en Detectron2 incorpora los populares *transformers*. COCO considera 80 categorías divididas entre objetos de exteriores (semáforos, árboles, etc.) e interiores (sofá, tv, etc.). Estas categorías incluyen, por ejemplo, tres tipos de fruta: banana, manzana y naranja, por lo que el resto de frutas no serán detectables y, por lo tanto, el robot no podrá operar con ellas. Esta limitación surge porque se considera un *vocabulario cerrado* de objetos detectables.

Una vez los objetos han sido identificados e introducidos en el mapa, lo que se conoce como información factual, esta se acompaña de información semántica codificada en una base de conocimiento. Esta base de conocimiento suele adoptar la forma de una ontología, donde se definen una serie de conceptos (p.ej., microondas, frigorífico, mesa, etc.), sus propiedades (p.ej., los microondas tienen forma de caja, son de tamaño medio, necesitan electricidad, etc.), sus funcionalidades (se emplean para calentar comida, pueden dar la hora) y sus relaciones (se suelen encontrar en cocinas encima de muebles como mesas). De este modo, los objetos presentes en el mapa se conectan con la información semántica asociada a ellos, permitiendo al robot considerarlos en la realización de sus tareas (Fernandez-Chaves et al., 2021; Monroy et al., 2018). La base de conocimiento se codifica a partir de información proporcionada por expertos del dominio en cuestión, un proceso tedioso que requiere definir con precisión conceptos y relaciones, resultando en una representación finita y posiblemente sesgada. Esto supone una limitación de *base de conocimiento cerrada* que, de manera similar al vocabulario cerrado, restringe la operación del robot a trabajar con la información semántica allí codificada.

Recientemente han aparecido artículos que se enfrentan a ambas limitaciones mediante la utilización de modelos a gran escala. Quizá el más popular sea el trabajo presentado por Gu et al. (2023), donde se introduce la representación denominada ConceptGraphs. Un ConceptGraph es un mapa semántico que captura el entorno de trabajo mediante un grafo de escena (del inglés *scene graph*) en el cual los nodos modelan los objetos identificados en el entorno, y los arcos definen las relaciones geométricas entre ellos (encima de, dentro de, etc.). Los autores consideran un vocabulario abierto de categorías de objetos mediante la utilización de un modelo de visión y lenguaje a gran escala (del inglés *Large Vision Language Model*, *LVLM*) que describe objetos segmentados en imágenes, y emplean un modelo de lenguaje a gran escala (del inglés *Large Language Model*, *LLM*) para obtener las relaciones geométricas entre objetos. Ante una cierta petición por parte del usuario, el grafo es aprovechado por el LLM como base de conocimiento abierta para generar la tarea a ejecutar por el robot para solventarla. Una de las limitaciones de este enfoque es la confianza que se deposita sobre los LLMs y sus respuestas, siendo bien sabido que estas pueden incluir información incorrecta, inexacta o inventada (Yao et al., 2023), resultando en un comportamiento erróneo del robot.

Este artículo describe un método para la creación y el aprovechamiento de mapas semánticos, basado en el propuesto por ConceptGraphs, que pretende mitigar dicha incertidumbre sobre la respuesta de los LLM. Para ello se proponen dos modificaciones sobre el método original. La primera consiste en su adaptación para trabajar con lenguajes a gran escala del estado del arte, en concreto Gemini 1.5 como LVLM y ChaptGPT-4o como LLM. Es conocido que la evolución de estos modelos es vertiginosa, por lo que la utilización de sus últimas versiones puede ser diferencial en cuanto a la calidad de sus respuestas. En segundo lugar, a la hora de explotar el mapa, se propone la inclusión de una nueva etapa de refinamiento de la respuesta del LLM mediante la técnica de Reflexión (Madaan et al., 2024). En concreto, este refinamiento se basa en un proceso de auto-reflexión donde el propio LLM se cuestiona sobre la validez de su resultado inicial, obteniendo un *feedback* que él mismo emplea para refinar su respuesta. En la definición de las entradas o *prompts* adecuados radica el éxito de esta etapa, permitiendo minimizar el número de respuestas incorrectas. La implementación del método con las mejoras propuestas se encuentra disponible públicamente en `https://github.com/jemoncadar/semantics-maps-moncada`. Para validar las propuestas se han realizado una serie de experimentos de construcción y aprovechamiento de mapas semánticos empleando el repositorio ScanNet, el cual contiene información capturada del mundo real, resultando en mapas correctamente creados y en una mejora de las respuestas para su aprovechamiento.

2. Descripción del método para mapeo semántico

El método desarrollado, basado en ConceptGraphs, toma como **entrada** una secuencia de imágenes RGB-D, junto con sus poses, capturada en un entorno interior. A partir de dicha entrada, produce como **salida** un mapa semántico del entorno construido mediante el uso de modelos de gran escala. Este mapa se presenta como un grafo de escena, donde cada nodo representa un objeto y cada arista refleja la relación semántica entre dos objetos.

El ***pipeline*** de este método se divide en una serie de etapas que deben ser ejecutadas secuencialmente (ver Fig. 1). Ini-

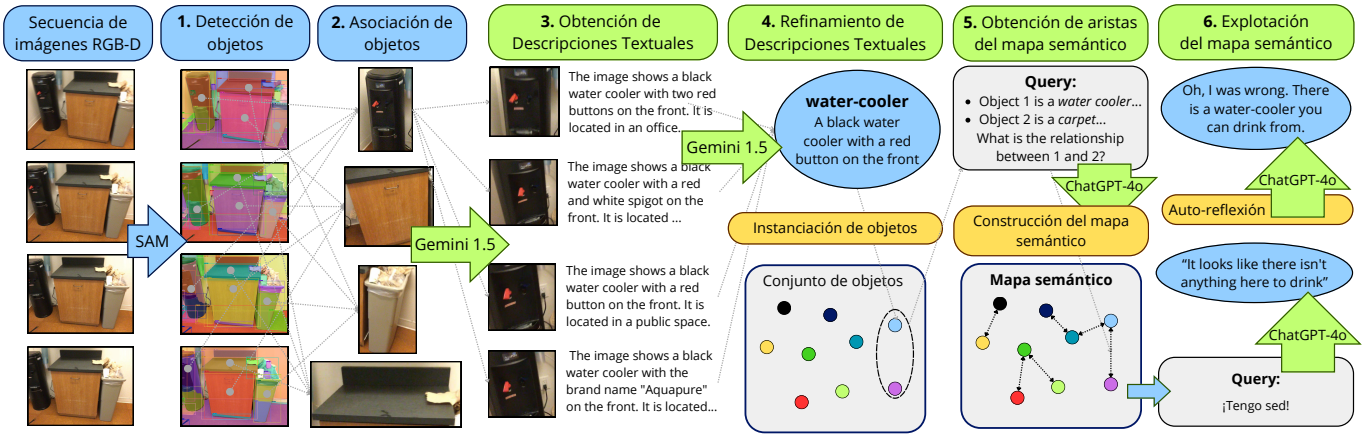

Figura 1: Visión general del método propuesto basado en ConceptGraphs, ilustrado sobre la secuencia `scene0003_02` del conjunto de datos ScanNet. Las etapas resaltadas en verde han sido modificadas y/o añadidas para incluir nuestras propuestas con respecto al trabajo original de ConceptGraphs.

cialmente, cada imagen de entrada se somete a un proceso de detección de objetos, capaz de identificar un conjunto abierto de categorías (ver Sec. 2.1). Esto resulta en múltiples vistas de cada objeto presente en el entorno. En un paso posterior, se realiza una etapa de asociación de datos, donde las detecciones que pertenezcan a los mismos objetos físicos se agrupan, utilizando un algoritmo que considera tanto la similitud geométrica como semántica, formando así los **nodos** del mapa semántico (Sec. 2.2).

Posteriormente, se generan **descripciones textuales** generadas por un LVLM para cada uno de los objetos (Sec. 2.3). Concretamente, estas descripciones se realizan sobre las N observaciones del objeto con un mayor valor de confianza en la detección dado por el detector de objetos. Estas descripciones individuales son luego procesadas mediante un LLM para generar una única descripción completa y coherente del objeto.

Para completar la construcción del mapa, las interacciones y relaciones entre los nodos adyacentes se deducen utilizando el LLM que, teniendo en cuenta el conocimiento general del mundo aprendido en el entrenamiento, establece las conexiones semánticas pertinentes entre ellos (Sec. 2.4). Estas conexiones son formalizadas como las **aristas** del mapa semántico, completando así el grafo semántico resultado del método.

Finalmente, el mapa construido puede ser aprovechado por el robot para completar sus tareas, labor que realiza el mismo LLM sobre el que se ha diseñado una etapa de refinamiento de la respuesta mediante la técnica de Reflexión, aumentando la robustez en la operativa del robot (Sec. 2.5).

2.1. *Detección de objetos*

El proceso de detección de objetos se realiza en dos etapas: i) segmentación de objetos y ii) descripción de objetos. En primer lugar, para segmentar los objetos presentes en cada imagen RGB $\mathcal{I}_t^{RGB}$ se utiliza Segment Anything (SAM) (Kirillov et al., 2023), un método del estado del arte que permite detectar objetos de manera agnóstica a sus categorías. Como salida, SAM ofrece una serie de **máscaras** $\{\mathbf{m}_{t,i}\}_{i=1\ldots M}$ correspondientes a cada uno de los objetos detectados. A continuación, se aplica cada máscara a la imagen $\mathcal{I}_t^{RGB}$ para extraer el fragmento de la imagen en la que aparece cada objeto, para posteriormente procesarlo con un extractor de características. Concretamente, en este caso se utiliza el modelo CLIP (Radford et al., 2021), el cuál devuelve un vector de características semánticas $\mathbf{f}_{t,i}$ que describe al fragmento de imagen procesado. Estos vectores de características se emplearán en una etapa posterior para realizar la asociación de datos entre múltiples observaciones de un mismo objeto físico.

2.2. *Asociación de objetos*

Hasta ahora, cada detección en una imagen viene caracterizada por una máscara $\mathbf{m}_{t,i}$ y un vector de características $\mathbf{f}_{t,i}$. Adicionalmente, cada detección es anotada con su nube de puntos $\mathbf{p}_{t,i}$, que se obtiene proyectando al espacio 3D los pixeles que caen dentro de $\mathbf{m}_{t,i}$, haciendo uso para ello de la matriz intrínseca de la cámara K y la imagen de profundidad $\mathcal{I}_t^D$. La nube de puntos obtenida es filtrada para eliminar puntos espurios, haciendo uso para ello de un método de agrupamiento basado en densidad, concretamente, DBSCAN (Ester et al., 1996).

Una vez que todas las detecciones obtenidas en $\mathcal{I}_t^{RGB}$ han sido caracterizadas con la máscara $\mathbf{m}_{t,i}$, el vector de características $\mathbf{f}_{t,i}$ y la correspondiente nube de puntos $\mathbf{p}_{t,i}$, se procede a realizar la etapa de asociación de datos para agrupar las diferentes observaciones de un mismo objeto físico en diferentes imágenes. Para ello, se calcula una similitud entre cada objeto detectado en la imagen actual $\langle \mathbf{p}_{t,i}, \mathbf{f}_{t,i} \rangle$ y los objetos ya existentes en el mapa $\langle \mathbf{p}_{\mathbf{o}_j}, \mathbf{m}_{\mathbf{o}_j} \rangle$ con los que se comparta algún solapamiento geométrico. La similitud total se calcula atendiendo a un doble criterio, teniendo en cuenta tanto la geometría como la semántica:

1. La **similitud geométrica** $s_{\text{geo}}(i,j) = \text{nnratio}(\mathbf{p}_{t,i}, \mathbf{p}_{\mathbf{o}_j})$ se calcula como la proporción de puntos en la nube de puntos del objeto detectado que tienen vecinos más cercanos en la nube de puntos del objeto en el mapa, dado un cierto umbral de distancia.
2. La **similitud semántica** $s_{\text{sem}}(i,j) = \mathbf{f}_{t,i}^T \mathbf{f}_{\mathbf{o}_j}/2 + 1/2$ se calcula como la distancia coseno normalizada entre los descriptores visuales (obtenidos en el paso anterior) correspondientes.

La **similitud total** $s(i,j) = s_{\text{geo}}(i,j) + s_{\text{sem}}(i,j)$ se calcula como la suma de la similitud geométrica y semántica. La

asociación a un determinado objeto se realiza siguiendo un algoritmo voraz, es decir, cada detección se asocia con el objeto existente con mayor similitud. Si dicha detección no tiene una similitud mínima δ_{sim} con ninguno de los objetos presentes, se instancia un nuevo objeto.

Al asociar una detección $\langle \mathbf{f}_{t,i}, \mathbf{p}_{t,i} \rangle$ con un objeto ya instanciado, es necesario actualizar tanto el vector de características como la nube de puntos del nuevo objeto. El vector de características se actualiza como $\mathbf{f}_{\mathbf{o}_j} = (n_{\mathbf{o}_j}\mathbf{f}_{\mathbf{o}_j} + \mathbf{f}_{t,i})/(n_{\mathbf{o}_j} + 1)$ donde $n_{\mathbf{o}_j}$ es el número de detecciones que se han asociado al objeto $\mathbf{o}_j$ hasta el momento. La nube de puntos se actualiza incluyendo todos los puntos procedentes de la nueva detección, es decir, $\mathbf{p}_{t,i} \cup \mathbf{p}_{\mathbf{o}_j}$.

2.3. Descripciones textuales de los objetos

Una vez obtenidos los nodos del mapa semántico (es decir, los objetos), el siguiente paso es evolucionar de una representación basada en detecciones visuales en imágenes a una definición de los objetos mediante **descripciones textuales**. Dichas descripciones textuales se generan en un primer lugar, de manera individual para cada observación del objeto, y posteriormente se combinan en una única descripción global del objeto otorgada de completitud y coherencia.

2.3.1. Generación de descripciones textuales individuales

Para la obtención de las descripciones textuales, por cada objeto, se seleccionan los 10 fragmentos de imagen donde se observe dicho objeto que hayan sido etiquetados con una mayor confianza en la detección por SAM. Dichos fragmentos se combinan con con el texto "*Describe with 2 or 3 sentences maximum the central element of the image*" para formar un *prompt*, el cuál se le proporciona un a LVLM, en concreto Gemini 1.5 Pro (`gemini-1.5-pro`). Como resultado se obtiene una descripción textual individual para cada uno de los fragmentos procesados.

2.3.2. Extracción de la descripción textual global

Una vez obtenidas las descripciones textuales individuales de un determinado objeto, estas descripciones se combinan en un único *prompt* con el objetivo de que un LVLM lo procese para obtener una única descripción global del objeto. En concreto, la instrucción principal de este *prompt* es *refinar* y *resumir* las descripciones dadas, extrayendo las características comunes de las descripciones individuales, y descartando aquellas que difieran radicalmente de la mayoría. Además, en dicho *prompt* se solicita al LVLM que genere una **etiqueta lingüística** semánticamente rica, compuesta de dos o tres palabras, que describa el objeto basándose en las descripciones textuales. En concreto, en este trabajo, el *prompt* completo es proporcionado a Gemini 1.5 Pro (`gemini-1.5-pro`) para obtener la descripción textual global del objeto.

2.4. Generación del mapa semántico

Hasta este punto, el *pipeline* se ha centrado en identificar los nodos del mapa semántico, que representan los objetos del entorno. A continuación, es necesario establecer las aristas, que representan las relaciones espaciales entre los objetos. Para ello, en primer lugar, se estima la conectividad entre todos los objetos teniendo en cuenta sus solapamientos espaciales. Esta estimación consiste en el cálculo de la intersección sobre la unión (IoU, por sus siglas en inglés, que significa *Intersection over Union*) entre los *bounding boxes* de cada par de objetos, obteniendo así una matriz de similitud. Sobre esta matriz de similitud se estima un árbol de expansión mínima, cuyas aristas unirán los nodos potencialmente conectados. A partir de cada par de objetos potencialmente conectados, se crea un *prompt* cuya instrucción principal pide describir la relación espacial entre los objetos en cuestión. Los objetos se representan en el *prompt* con su localización tridimensional (coordenadas y dimensiones de sus *bounding boxes*) y la descripción textual generada en el paso anterior, la cual es esencial para proporcionar al modelo de lenguaje contextualización sobre la naturaleza de los objetos. En este trabajo, este *prompt* se introduce a GPT-4o (`gpt-4o`), que responde identificando la relación entre los objetos y proporciona además una explicación detallada del razonamiento subyacente.

Para la descripción de la relación entre dos objetos, en el *prompt* se proporcionan varios ejemplos, como que un objeto esté dentro de otro (`object1` *in* `object2`) o que un objeto esté encima de otro (`object1` *on* `object2`). Sin embargo, también se permite al LLM expresar la relación usando términos alternativos o más detallados. Esta flexibilidad es esencial para capturar la diversidad y complejidad de las interacciones espaciales en escenarios del mundo real. Una vez completado el *pipeline*, el mapa semántico incluye los objetos, sus descripciones y las relaciones entre ellos. Para facilitar su uso posterior, es beneficioso expresar este mapa en un formato que otros modelos de lenguaje a gran escala puedan integrar y comprender fácilmente, como el aquí empleado, JSON.

2.5. Explotación del mapa semántico

Una vez construido el mapa semántico, se plantea su explotación en un escenario de interacción con el usuario. En particular, se emplea un LLM, con un *prompt* de sistema donde el mapa semántico es el elemento central, para responder a las peticiones de un usuario que solicita ayuda a un robot móvil para realizar una acción determinada. La instrucción de este *prompt* pide calcular hacia qué objeto del mapa semántico debe navegar el robot móvil para ayudar al usuario a completar la tarea. Adicionalmente, se solicita al LLM una lista de objetos potencialmente útiles ordenados por relevancia, una explicación de la consulta realizada por el usuario según su entendimiento, y una justificación de la decisión final.

Con el objetivo de refinar y ajustar las respuestas obtenidas por el LLM en este proceso, se ha empleado la técnica de la **auto-reflexión** (Madaan et al., 2024) sobre la respuesta inicial (ver Fig. 1). Al reflexionar sobre la planificación ofrecida en primera instancia, el LLM puede proporcionar explicaciones más claras, e incluso corregir errores que hubieran dado lugar a una ejecución errónea de la tarea. En particular, esta auto-reflexión tiene lugar con un *prompt* especializado que incluye la primera respuesta y tiene como instrucción principal evaluar dicha respuesta en términos de corrección (la consulta inferida y los objetos relevantes están correctamente identificados), relevancia (los objetos identificados están realmente ordenados por relevancia y no se omite ninguno) y claridad (las explicaciones son sencillas y sin ambigüedades). En función de la evaluación anterior, se pide devolver una lista de sugerencias claras y concisas sobre modificaciones pertinentes para mejorar la respuesta inicial.

Por último, la reflexión generada en el paso anterior se materializa en una corrección de la respuesta. Esto se realiza utilizando un *prompt* con información sobre todas las respuestas anteriores, que pide devolver una respuesta similar a la original pero con las correcciones señaladas en el paso anterior.

3. Experimentos

Para validar las propuestas presentadas en este trabajo, se han llevado a cabo una serie de experimentos en dos conjuntos de datos distintos: Replica (Straub et al., 2019) y ScanNet (Dai et al., 2017). El primer conjunto de datos se ha utilizado para verificar y validar el correcto funcionamiento de ConceptGraphs, en concreto en la secuencia `room0`, dado que fue empleada por los autores en el trabajo original. Confirmado su correcto funcionamiento, los experimentos adicionales se realizaron con ScanNet, ya que es un conjunto de datos proveniente de entornos reales. En particular, de ScanNet se seleccionaron dos secuencias diferentes: `scene0000_01`, la cuál proviene de un apartamento con cocina y baño, representando así a una escena amplia con cierta complejidad, y `scene0003_02`, que representa una cocina de pequeñas dimensiones, lo que permite un control exhaustivo sobre el número y la disposición de los objetos presentes. De esta manera, el realizar experimentos en entornos reales con características diversas permite evaluar tanto las propuestas como el despliegue de estos métodos en escenarios reales.

Cabe señalar que la implementación del método descrito ha sido desplegada empleando una arquitectura basada en contenedores Linux que opera de manera distribuida (Ambrosio-Cestero et al., 2024). Esto ha permitido ejecutar las distintas etapas empleando, por ejemplo, dispositivos en el borde, sin comprometer los típicamente escasos recursos de un robot móvil.

3.1. Mapa semántico de ScanNet

Previamente a la ejecución del método descrito, conviene realizar una verificación preliminar de coherencia sobre el conjunto de datos a emplear, la cual consiste en aplicar un algoritmo de Localización y Mapeo Simultáneo (conocido como SLAM, por sus siglas en inglés, que significa *Simultaneous Localization and Mapping*) para reconstruir la escena 3D. En este caso, concretamente se ha utilizado el método GradSLAM (Krishna Murthy et al., 2020), y los resultados obtenidos se muestran en la Fig. 2. Dado que las reconstrucciones obtenidas son correctas, se puede asegurar la validez de los datos y continuar con las siguientes etapas del método.

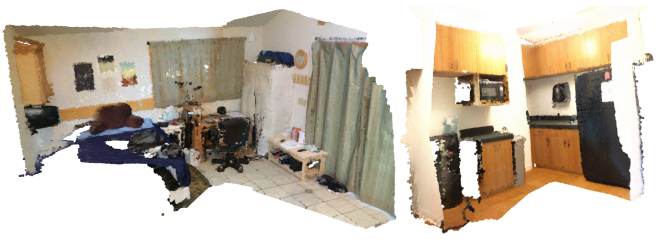

Figura 2: Resultado de la aplicación del método GradSLAM a partes de la secuencia `scene0000_01` (izquierda) y `scene0003_02` (derecha).

En el siguiente paso, que consiste en la detección de objetos en las imágenes, se segmentan las imágenes y se identifican los objetos presentes. Un ejemplo de esta fase se puede apreciar en la Fig. 1. Tras esto, para la asociación de datos sobre objetos se empleó un umbral de similitud de $\delta_{\text{sim}} = 1{,}0$, obteniéndose un total de 535 objetos para la secuencia `scene0000_01` y 43 para `scene0003_02`.

Una vez identificados los objetos y asociadas su observaciones en las diferentes imágenes, se procede a la obtención de las descripciones textuales individuales y su posterior refinamiento en un par <descripción global,etiqueta lingüística>. A modo ilustrativo, en la Fig. 1 se muestran cuatro fragmentos donde se ha observado un mismo objeto, así como las descripciones textuales individuales. También se muestra la descripción global refinada junto a la etiqueta lingüistica asignada.

Por último, la generación del mapa semántico resultó en objetos coherentes en consonancia con lo esperado. A continuación se puede observar un fragmento de los objetos del mapa semántico en formato JSON, que contiene únicamente dos objetos, de la secuencia `scene0003_02`:

```
[
  { "id": 1,
    "bbox_extent": [1.1, 0.6, 0.5],
    "bbox_center": [2.7, 0.9, 0.5],
    "possible_tags": ["water cooler", "spigot", "button", "cabinet",
        "office"],
    "object_tag": "water cooler",
    "caption": "a black water cooler with a red button on the
        front" }, ... ,
  { "id": 6,
    "bbox_extent": [0.9, 0.5, 0.3],
    "bbox_center": [1.4, 1.1, 0.5],
    "possible_tags": ["trash can", "garbage can", "plastic bag",
        "cat", "paper", "office"],
    "object_tag": "trash can",
    "caption": "a gray plastic trash can with a black plastic bag
        inside" }, ...
]
```

3.2. Explotación del mapa semántico de ScanNet

Por último, haciendo uso de los mapas semánticos generados, se llevó a cabo una serie de experimentos donde se realizaban ciertas consultas de interés acerca de la información incluida en el mapa que se le proporcionaba a un LLM (en concreto, ChatGPT-4o), simulando a un usuario que desea realizar diferentes tareas específicas en dicha escena. De entre los diferentes experimentos, es relevante señalar uno de ellos, donde se le planteó la consulta: *¡Necesito calentar mi comida!* (*I need to heat my food!*) sobre el mapa semántico de la escena `scene0003_02`. La respuesta ofrecida por el LLM fue la siguiente:

```
{
  "inferred_query": "Find objects related to heating food",
  "relevant_objects": [],
  "query_achievable": false,
  "final_relevant_objects": [],
  "explanation": "There are no objects in the scene explicitly
  related to heating food, such as a microwave or an oven.
  Therefore, the query cannot be achieved with the objects in
  the scene."
}
```

Esta respuesta es errónea dado que la escena incluye un microondas, un objeto que podría ser utilizado para completar la tarea requerida por el usuario. En estas circunstancias, se aplicó la técnica de auto-reflexión al LLM, incluyendo una etapa donde debe ser autocrítico con su respuesta y comprobar si era precisa, y en caso contrario, proceder con su corrección. El resultado de esta etapa fue el siguiente:

```
Correctness:
- The query "Find objects related to heating food" is accurate.
- The response incorrectly states that there are no objects
    related to heating food, while a "microwave oven" and a
    "refrigerator" are present.
- The query's achievable status is incorrectly marked as false due
    to this error.
- The explanation is not clear as it incorrectly claims no
    relevant objects are present.
Relevance:
- The response fails to identify all relevant objects related to
    heating food, such as the "microwave oven" and "refrigerator."
- Including and sorting these objects by relevance would improve
    accuracy.
Clarity:
- The response is clear in language and structure but may cause
    confusion by incorrectly asserting no relevant objects are
    present.
Suggestions for Improvement:
- Include "microwave oven" and "refrigerator" in the
    relevant_objects field.
- Revise the explanation to acknowledge the presence of relevant
    objects.
- Prioritize the most relevant objects in the final list.
- Provide more context on the relevance of these objects to
    heating food.
```

Entre otras correcciones, se señala que la respuesta original "afirma incorrectamente que no hay objetos relacionados con calentar comida, aunque en el mapa semántico hay un microondas y un refrigerador". Por último, en la última etapa de corrección de la respuesta original en función de la auto-reflexión se obtuvo la siguiente respuesta corregida y más acertada:

```
{
  "inferred_query": "Find objects related to heating food",
  "relevant_objects": ["11", "26", "31"],
  "query_achievable": true,
  "final_relevant_objects": ["11", "26", "31"],
  "explanation": "While not all objects in the scene are
    explicitly related to heating food, there are relevant
    objects like the 'microwave oven,' 'refrigerator,' and
    'paper note with magnets' that can be associated with food
    heating processes. The 'microwave oven' is used for heating
    food directly, the 'refrigerator' aids in preserving food
    that may be later heated, and the 'paper note with magnets'
    possibly contains information related to food heating
    instructions or recipes."
}
```

4. Conclusiones

El trabajo presentado surgió del estudio del uso de modelos a gran escala para la creación de mapas semánticos en robótica móvil, y de la necesidad de mejorar las respuestas de estos para alcanzar una operativa más robusta. Para ello, partiendo de la representación del estado del arte ConceptGraphs, se han propuesto dos estrategias que modifican el método original para mejorar su rendimiento: i) adaptar el método para que pueda operar con modelos más recientes y robustos, y ii) aplicar la técnica de Reflexión para que los modelos mejoren sus respuestas mediante la autoevaluación y autocrítica. Los experimentos realizados en entornos reales del conjunto de datos ScanNet han demostrado cualitativamente que estas estrategias son efectivas para mejorar el rendimiento global del método. Como trabajo futuro se plantea una evaluación cuantitativa de la propuesta.

Agradecimientos

Este trabajo ha sido desarrollado en el contexto de los proyectos ARPEGGIO (PID2020-117057GB-I00) y Voxeland (JA.B1-09), financiados por el Ministerio de Ciencia e Innovación y la Universidad de Málaga, respectivamente.

Referencias

Ambrosio-Cestero, G., Matez-Bandera, J.-L., Ruiz-Sarmiento, J.-R., Gonzalez-Jimenez, J., 2024. Container based architecture for mobile robotics. XLV Jornadas de Automática.

Chaves, D., Ruiz-Sarmiento, J.-R., Petkov, N., Gonzalez-Jimenez, J., 2019. Integration of cnn into a robotic architecture to build semantic maps of indoor environments. In: IWANN 2019. Springer, pp. 313–324.
DOI: https://doi.org/10.1007/978-3-030-20518-8_27

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., Nießner, M., 2017. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: kdd. Vol. 96. pp. 226–231.

Fernandez-Chaves, D., Ruiz-Sarmiento, J., Petkov, N., Gonzalez-Jimenez, J., 2021. Vimantic, a distributed robotic architecture for semantic mapping in indoor environments. Knowledge-Based Systems 232, 107440.
DOI: https://doi.org/10.1016/j.knosys.2021.107440

Gu, Q., Kuwajerwala, A., Morin, S., Jatavallabhula, K., Sen, B., Agarwal, A., Rivera, C., Paul, W., Ellis, K., Chellappa, R., Gan, C., de Melo, C., Tenenbaum, J., Torralba, A., Shkurti, F., Paull, L., 2023. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. arXiv.
DOI: https://doi.org/10.48550/arXiv.2309.16650

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969.
DOI: https://doi.org/10.48550/arXiv.1703.06870

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., Girshick, R., 2023. Segment anything.
DOI: https://doi.org/10.48550/arXiv.2304.02643

Krishna Murthy, J., Saryazdi, S., Iyer, G., Paull, L., 2020. gradslam: Dense slam meets automatic differentiation. In: arXiv.
DOI: https://doi.org/10.48550/arXiv.1910.10672

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C. L., 2014. Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference. pp. 740–755.
DOI: https://doi.org/10.48550/arXiv.1405.0312

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., et al., 2024. Self-refine: Iterative refinement with self-feedback. Advances in Neural Information Processing Systems 36.
DOI: https://doi.org/10.48550/arXiv.2303.17651

Monroy, J., Ruiz-Sarmiento, J.-R., Moreno, F.-A., Melendez-Fernandez, F., Galindo, C., Gonzalez-Jimenez, J., 2018. A semantic-based gas source localization with a mobile robot combining vision and chemical sensing. Sensors 18 (12), 4174.
DOI: https://doi.org/10.3390/s18124174

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I., 2021. Learning transferable visual models from natural language supervision.
DOI: https://doi.org/10.48550/arXiv.2103.00020

Rubio, F., Valero, F., Llopis-Albert, C., 2019. A review of mobile robots: Concepts, methods, theoretical framework, and applications. International Journal of Advanced Robotic Systems 16 (2).

Ruiz-Sarmiento, J.-R., Galindo, C., Gonzalez-Jimenez, J., 2017a. Building multiversal semantic maps for mobile robot operation. Knowledge-Based Systems 119, 257–272.
DOI: https://doi.org/10.1016/j.knosys.2016.12.016

Ruiz-Sarmiento, J. R., Galindo, C., González-Jiménez, J., 2017b. Robot@home, a robotic dataset for semantic mapping of home environments. International Journal of Robotics Research.
DOI: https://doi.org/10.1177/0278364917695640

Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J. J., Mur-Artal, R., Ren, C., Verma, S., et al., 2019. The replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797.
DOI: https://doi.org/10.48550/arXiv.1906.05797

Terven, J., Cordova-Esparza, D., 2023. A comprehensive review of yolo: From yolov1 to yolov8 and beyond. arXiv preprint arXiv:2304.00501.
DOI: https://doi.org/10.48550/arXiv.2304.00501

Yao, J.-Y., Ning, K.-P., Liu, Z.-H., Ning, M.-N., Yuan, L., 2023. Llm lies: Hallucinations are not bugs, but features as adversarial examples. arXiv preprint arXiv:2310.01469.
DOI: https://doi.org/10.48550/arXiv.2310.01469