Actas ROBOT 2011. 28-29 de Noviembre de 2011. Sevilla (España)

Partición de mapas para SLAM monocular en gran escala

E. Fernández-Moral, J. González-Jiménez y V. Arévalo

***Resumen*— En los últimos años, las soluciones de SLAM monocular basadas en *bundle adjustment* (BA) sobre keyframes han conseguido excelentes resultados en cuanto a precisión, eficiencia computacional y densidad del mapa. Pese a esto, su ámbito de aplicación está muy limitado, ya que su coste computacional se dispara cuando el mapa crece. El propósito de este trabajo es extender las soluciones basadas en BA a espacios de trabajo más grandes manteniendo la precisión y densidad de los mapas generados. Para lograr esto proponemos dividir estos mapas basándonos en el corte normalizado de un grafo en el que los nodos representan *keyframes* y el peso de cada arco viene dado por la medida SSO (*Sensed Space Overlap*) [1]. De esta forma se obtiene una representación métrico-topológica del espacio que, además, es útil para abordar problemas típicos de robótica móvil tales como el cierre de bucle o la planificación de tareas. Para validar nuestra propuesta hemos integrado nuestro algoritmo de partición en PTAM [2] y hemos realizado diversos experimentos en un escenario real integrado por varias habitaciones, obteniendo unos resultados muy prometedores en términos de eficiencia computacional y precisión del mapa.**

I. Introducción

El SLAM monocular es una técnica que consiste en construir un mapa de un entorno desconocido mientras el sistema se localiza con la información proporcionada por una única cámara. Este problema es complejo, ya que existen numerosos aspectos que dificultan crear un mapa preciso de la escena, por ejemplo, la posición 3D de las características no se puede determinar de forma directa; existe una ambigüedad en la escala inherente a la proyección perspectiva; y se produce deriva en la escala cuando el camino recorrido crece significativamente. Sin embargo, y a pesar de los inconvenientes mencionados, la simplicidad y el bajo coste de este tipo de sistemas hacen que la solución a este problema continúe siendo un importante campo de investigación.

Varios algoritmos de SLAM monocular han aparecido desde que, en 2003, se presentara el primero de estos sistemas funcionando en tiempo real [3]. Esta solución empleaba filtrado Bayesiano para actualizar recursivamente las posiciones de la cámara y de las *landmarks* del mapa junto con la matriz de covarianza de éstas [4]. El coste incremental de esta solución hace que sólo se puedan crear mapas de pequeños espacios compuestos por pocas *landmarks*. En la literatura se pueden encontrar soluciones que reducen la complejidad computacional del problema para hacerlo escalable ([5], [6], [7] y [8]) pero ninguna proporciona una representación densa de la escena, estando su aplicación restringida prácticamente a problemas puramente de localización.

En los últimos años se han propuesto técnicas de SLAM monocular basadas en *bundle adjustment* (*BA*) capaces de crear mapas densos y de gran precisión [2], [9], [10] y [11]. Éstas sólo funcionan adecuadamente en entornos de trabajo muy limitados (por ejemplo, un despacho pequeño), al ser incapaces de hacer frente, en tiempo real, el incremento de información que se produce cuando el escenario crece (por ejemplo, varias habitaciones). Para superar esta limitación proponemos partir el mapa en sub-mapas, de tal manera que las *landmarks* contenidas en ellos estén muy interrelacionadas y lo estén poco con las de otros. Así, el *BA* se limita a cada sub-mapa, haciéndose escalable el problema de SLAM y, por tanto, haciendo posible representar grandes entornos. Además, la estructura resultante permite etiquetar los sub-mapas como lugares topológicos que pueden ser utilizados para tareas de más alto nivel, como son el cierre de bucle o la planificación de tareas en el ámbito de la robótica móvil [12].

En este trabajo se presenta un procedimiento genérico de partición de mapas visuales que puede ser integrado en cualquiera de las técnicas de SLAM basadas en *BA*. Este procedimiento ha sido validado experimentalmente, creando mapas de escenarios compuestos por varias habitaciones. Para ello hemos integrado nuestra técnica de partición en PTAM (*Parallel Tracking and Mapping*) [2]. Los resultados arrojados por las pruebas realizadas demuestran que nuestra estrategia supone una solución factible para extender el rango de aplicación de los actuales algoritmos de SLAM monocular, como PTAM, a grandes espacios de trabajo, mejorando la eficiencia del sistema y la consistencia de los mapas creados.

El resto del artículo se organiza como sigue: la sección siguiente expone la problemática asociada a la partición de mapas; en la sección III se detalla nuestro método de partición y su integración en PTAM; la sección IV describe los experimentos llevados a cabo junto con sus resultados y, finalmente, la sección V expone las conclusiones y posibles líneas de investigación para el futuro.

Este trabajo ha sido financiado a través del proyecto DPI2008-03527 del Plan Nacional de Investigación.

Dpto. Ingeniería de Sistemas y Automática de la Universidad de Málaga. E.T.S. Ingeniería de Informática-Telecomunicación. Campus de Teatinos, 29071, Málaga. E-mail: eduardofernandez@isa.uma.es

II. PARTICIÓN DE MAPAS

A. Por qué dividir el mapa

Los mapas generados por sistemas de SLAM monocular basados en *BA*, como PTAM, presentan el inconveniente de estar limitados a pequeños espacios. Estos mapas son útiles para aplicaciones tales como realidad aumentada, videojuegos o la construcción de modelos muy precisos de pequeños entornos, pero no lo son, por ejemplo, para robótica móvil, donde se necesitan mapas de espacios más grandes.

En este trabajo se propone dividir el mapa en pequeños mapas locales con el objetivo de representar grandes espacios. Esta división en sub-mapas permite aproximar la optimización global con *BA* mediante la optimización individual de los diferentes sub-mapas, reduciendo así el coste computacional del proceso. Seguidamente se demuestra formalmente esto último.

El *bundle adjustment* minimiza el error de reproyección de las *landmarks* sobre los *keyframes* respecto a las posiciones en el espacio de ambos. El coste computacional de esta minimización es de orden cúbico con respecto al número de *keyframes* [2], de forma que éste crece rápidamente a medida que lo hace el mapa. Matemáticamente, el *BA* puede expresarse como:

$$\min_{\mathbf{a}_j,\,\mathbf{b}_i} \sum_{i=1,n} \sum_{j=1,m} v_{ij}\, d(\mathbf{Q}(\mathbf{a}_j,\,\mathbf{b}_i),\,\mathbf{x}_{ij})^2 \qquad (1)$$

donde n es el número de puntos 3D que forman el mapa; m es el número de *keyframes* desde los que estos puntos pueden ser observados; v_{ij} es una variable binaria que toma el valor 1 si el punto i es observado en el *keyframe* j y 0 en caso contrario; $\mathbf{a}_j$ es un vector que representa la posición (*pose*) desde la que ha sido capturado el *keyframe j;* $\mathbf{b}_i$ es un vector que representa la posición del punto i; $\mathbf{Q}(\mathbf{a}_j,\,\mathbf{b}_i)$ es la estimación de la proyección del punto i sobre la imagen j, y $\mathbf{x}_{ij}$ su observación en la imagen. $d(\mathbf{x},\mathbf{x'})$ es la distancia Euclidea entre los puntos representados por los vectores **x** e **x'**, esto es, el error de reproyección.

Si dividimos el mapa en sub-mapas locales caracterizados por estar "débilmente" interconectados, es decir, en los que v_{ij} es predominantemente 0 cuando el punto i pertenece a un sub-mapa diferente al del *keyframe j*, entonces podemos aproximar el *BA* mediante la optimización independiente de los diferentes sub-mapas, reduciendo considerablemente el coste computacional. Así, cuando el mapa consta de N sub-mapas, cada uno con n^k *landmarks* y m^l *keyframes*, (1) puede reescribirse como

$$\min_{\mathbf{a}_j^l,\,\mathbf{b}_i^k} \sum_{k=1,N} \sum_{l=1,N} \left(\sum_{i=1,n^k} \sum_{j=1,m^l} v_{ij}^{kl}\, d(\mathbf{Q}(\mathbf{a}_j^l,\,\mathbf{b}_i^k),\,\mathbf{x}_{ij}^{kl})^2 \right) \qquad (2)$$

donde el superíndice k sobre i indica que el punto i pertenece al *k-ésimo* sub-mapa y, de igual forma, l sobre j indica que el *keyframe j* pertenece al *l-ésimo* sub-mapa. Esta expresión se puede reescribir en función de la conectividad de los sub-mapas, quedando del siguiente modo

$$\min_{\mathbf{a}_j^l,\,\mathbf{b}_i^k} \sum_{k=1,N} \Bigg(\underbrace{\sum_{l=1,N,\; l\neq k} \sum_{i=1,n^k} \sum_{j=1,m^l} v_{ij}^{kl}\, d(\mathbf{Q}(\mathbf{a}_j^l,\,\mathbf{b}_i^k),\,\mathbf{x}_{ij}^{kl})^2}_{A} + \underbrace{\sum_{j=1,n^k} \sum_{j=1,m^k} v_{ij}^{kk}\, d(\mathbf{Q}(\mathbf{a}_j^k,\,\mathbf{b}_i^k),\,\mathbf{x}_{ij}^{kk})^2}_{B} \Bigg) \qquad (3)$$

donde el primer sumando corresponde a la interconexión de los diferentes sub-mapas (puntos observados en *keyframes* de sub-mapas diferentes), mientras que el segundo corresponde a su intraconexión (puntos observados en *keyframes* de los mismos sub-mapas).

Puesto que la división en sub-mapas es tal que estos están "débilmente" interconectados, y asumiendo que los errores de reproyección de los puntos son del mismo orden, entonces podemos afirmar que A es despreciable con respecto a B, en cuyo caso, la optimización del mapa global puede ser aproximada mediante la siguiente expresión:

$$\sum_{k=1,N} \left(\min_{\mathbf{a}_j^k,\,\mathbf{b}_i^k} \sum_{i=1,n^k} \sum_{j=1,m^k} v_{ij}\, d(\mathbf{Q}(\mathbf{a}_j,\,\mathbf{b}_i),\,\mathbf{x}_{ij})^2 \right) \qquad (4)$$

Esta aproximación equivale a optimizar cada sub-mapa de forma independiente, de modo que el coste computacional se reduce drásticamente cuando los sub-mapas son significativamente menores que el mapa global. De hecho, esta aproximación sería equivalente a la expresión original de no existir conexión entre los diferentes sub-mapas.

Finalmente, añadir que, gracias a la casi independencia entre sub-mapas, cuando la optimización se lleva a cabo de forma incremental, como es el caso que nos ocupa, no es necesario estar optimizando todos los sub-mapas continuamente, sino sólo el mapa actual.

B. Cómo dividir el mapa

La cuestión clave en la creación de un mapa métrico-topológico es cómo agrupar la información captada por la cámara en sub-mapas. Entre los métodos propuestos hasta el momento podemos destacar: el propuesto en [13] (*Atlas*), en el que un nuevo sub-mapa es generado cuando las estimaciones de localización bajan de un umbral de calidad; o el propuesto en [14], donde se crea un nuevo mapa cuando el número de *landmarks* del mapa previo alcanza un límite determinado. Sin embargo, ninguno de estos métodos proporciona una solución fundamentada matemáticamente o que no dependa de reglas heurísticas que deban ser ajustadas manualmente para cada entorno de trabajo. Otras soluciones emplean corte normalizado de un grafo para

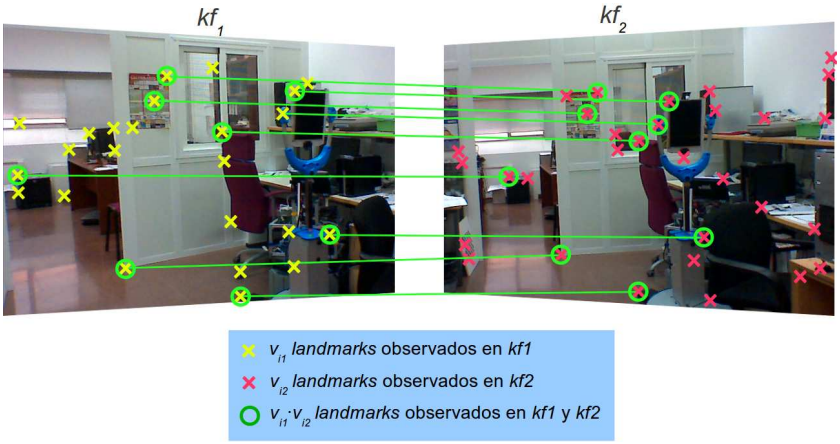

Fig. 1. Ejemplo de *Sensed Space Overlap* (SSO) entre dos *keyframes* (ver ecuación 5).

dividir el mapa de acuerdo a una propiedad mesurable de las observaciones. Sobre esta base matemática, [15] aborda el problema de la construcción de un mapa jerárquico mediante imágenes; [1] genera un mapa métrico-topológico utilizando un sensor láser; y [8] divide el mapa generado en un ámbito de SLAM monocular basado en filtrado Bayesiano para reducir la complejidad del sistema. Éste último trabajo es el más cercano al nuestro, pero al emplear una estrategia de filtrado Bayesiano, crea un mapa con pocas características de escasa utilidad más allá del problema de localización. Por el contrario, nosotros hacemos uso del corte de grafo en un marco de SLAM visual basado en *keyframes*, que permite una representación mucho más detallada de la escena. Para ello empleamos un criterio para asignar el peso a los arcos (detallado en II.B) que es consistente con la división en sub-mapas débilmente conectados, permitiendo utilizar la aproximación descrita en II.A.

La solución propuesta consiste en dividir el mapa en sub-mapas en los que sus *keyframes* comparten un alto porcentaje de observaciones, realizando el corte entre los *keyframes* que compartan menos información. De esta forma se minimiza la pérdida de información si un sub-mapa se aísla de los demás. El enfoque propuesto se basa en considerar el mapa como un grafo, en el que sus nodos representan los *keyframes* y sus arcos constituyen una medida de cuanta información comparten. Para construir este modelo se ha de tener en cuenta dos aspectos importantes: primero, establecer cómo se asignaran los pesos de los arcos, y segundo, elegir el criterio para realizar la división.

En cuanto a lo primero, los pesos de los arcos se establecen de acuerdo al "*Sensed Space Overlap*" (*SSO*), siguiendo la estrategia empleada en [16]. Esta medida refleja el grado de información común entre dos *keyframes*, calculando la relación entre el número de *landmarks* comunes con respecto al número total de *landmarks*

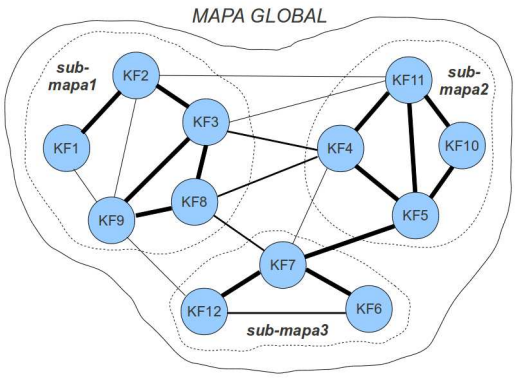

Fig. 2. Ejemplo de corte normalizado de un grafo. El grosor de los arcos indica el grado de conexión de los nodos.

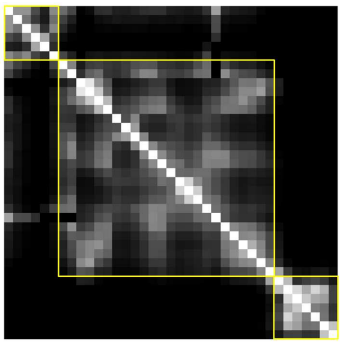

Fig. 3. Matriz de SSO donde cada *keyframe* se corresponde con una fila y una columna, y la luminosidad del elemento s_{ij} representa el SSO entre los *keyframes* i y j.

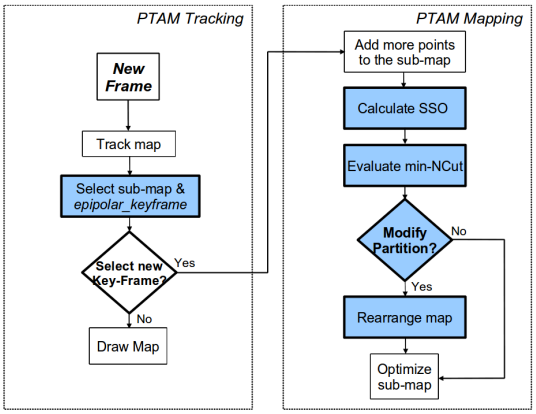

Fig. 4. Partición de mapas en PTAM. Los bloques azules se corresponden con etapas del proceso de partición del mapa.

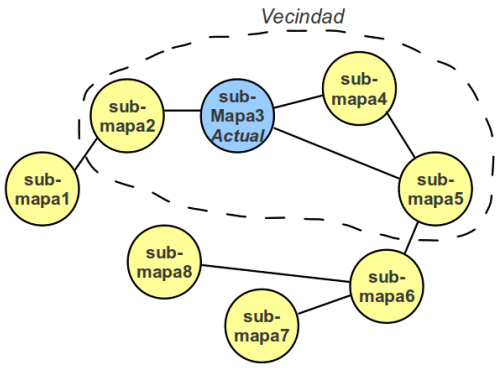

Fig. 5. Representación topológica de los diferentes sub-mapas de acuerdo a su conexión.

diferentes que son observadas en ambos *keyframes*. Matemáticamente

$$SSO(kf_1, kf_2) = \frac{\sum v_{i1} \cdot v_{i2}}{\sum v_{i1} + \sum v_{i2} - \sum v_{i1} \cdot v_{i2}} \quad (5)$$

donde v_{i1} y v_{i2} se refieren, al igual que en el apartado anterior, a las *landmarks i* observadas en los *keyframes* kf_1 y kf_2, respectivamente, y su producto se corresponde al conjunto de *landmarks* observadas en ambos (ver figura 1).

En cuanto al criterio para realizar la división, hemos utilizado la minimización del *normalized-cut* (*min-Ncut*), que fue introducido originalmente en [17][15] y ha sido usado previamente en construcción de mapas en el ámbito de SLAM en [1], [8] y [15]. La figura 2 ilustra este concepto para un mapa que es dividido en 3 sub-mapas, donde se observa cómo los nodos con conexiones más fuertes (representadas con líneas más gruesas) son agrupados juntos. En la figura 3 podemos observar la matriz simétrica de *SSO* correspondiente a un pequeño mapa, que se ha reordenado según el *min-NCut*, y que da lugar a tres agrupaciones de *keyframes* (o sub-mapas). Estos sub-mapas constituyen una estructura eficiente para el problema de SLAM monocular, puesto que las observaciones más alejadas del sub-mapa actual son descartadas a la hora de reproyectar las *landmarks* así como en la optimización del mapa. Esta técnica para dividir el mapa requiere establecer un umbral para el *min-NCut* bajo el cual el corte proporcionado es aceptado, afectando al número y tamaño de las divisiones resultantes. No obstante, la elección de dicho parámetro no depende del escenario de aplicación, y el funcionamiento de nuestro sistema muestra poca sensibilidad a la elección de este parámetro, obteniendo resultados de funcionamiento similares (a excepción del número de sub-mapas) para diferentes valores de dicho umbral.

III. Procedimiento de Partición e Integración en PTAM

En esta sección describimos la integración de nuestro algoritmo de partición con PTAM (*Parallel Tracking and Mapping*) [2]. PTAM es un algoritmo de SLAM monocular basado en *BA* que separa las tareas de localización y generación del mapa en procesos diferentes, permitiendo su ejecución eficiente en tiempo real. Esta técnica requiere la adquisición de un mapa inicial para poder funcionar de forma automática. Dicho mapa inicial se adquiere con un procedimiento de *Structure from Motion* (*SfM*) aplicado sobre dos *keyframes* seleccionados por el usuario.

Una vez creado el mapa inicial, el sistema analiza las imágenes que recibe de la cámara para localizarse. El sistema combina un modelo de movimiento para la cámara y un método de estimación de su rotación con la búsqueda activa de *landmarks* para determinar la *pose* (posición y orientación). Esta *pose* es refinada mediante la minimización del error de reproyección de los puntos del mapa en la imagen. La selección de *keyframes* se rige mediante unas reglas heurísticas "ad-hoc", en el proceso de localización. Mientras tanto, el proceso de creación del mapa lleva a cabo una optimización continua de éste, y lo incrementa añadiendo *landmarks* detectadas en los nuevos *keyframes*, que son extraídos mediante búsqueda epipolar entre cada nuevo *keyframe* y el *keyframe* del mapa más cercano a éste.

En la figura 4 se muestra una representación esquemática del proceso de partición propuesto sobre PTAM. Este proceso de partición comienza cuando se selecciona un nuevo *keyframe*, escogiendo también el sub-mapa al que pertenece y el *keyframe* con el que se hará la búsqueda epipolar de nuevas *landmarks* (denominado *epipolar_keyframe*). Posteriormente, en el proceso de construcción del mapa (*mapping*), se añaden nuevos puntos al mapa y se evalúa el *SSO* con todos los *keyframes* de la

Fig. 6. Sistema de visión utilizado en los experimentos.

vecindad, que incluye al conjunto de sub-mapas conectados directamente al sub-mapa actual (ver figura 5). Cuando la evaluación de la partición modifica la estructura actual del mapa, éste es reestructurado. Seguidamente, el sub-mapa actual es optimizado, siguiendo el funcionamiento normal de PTAM. Este sistema de partición afecta a los sub-mapas contenidos en la *vecindad* del sub-mapa actual, pudiendo crear nuevos sub-mapas así como combinar sub-mapas existentes. El coste computacional de reestructurar el mapa depende únicamente del número de modificaciones a realizar, y está acotado por el tamaño de la *vecindad*. Este proceso de reestructuración no afecta rendimiento del sistema, pues la localización sigue funcionando independientemente en otro hilo, y su carga computacional es insignificante en comparación con la optimización del mapa, con lo que tampoco afecta al proceso de mapeo.

IV. Experimentos y Resultados

Nuestro método ha sido validado con diferentes experimentos, en los que se demuestra su adaptabilidad a grandes espacios y se compara su eficiencia con la solución de un único mapa. A continuación describimos tres de ellos en los que hemos utilizado una webcam Philips SPC640NC, conectada mediante USB a un portátil Intel Core2 Duo a 2.4 GHz, 2Gb de memoria y tarjeta gráfica integrada nVidia GeForce-9400, con sistema operativo Linux (ver figura 6).

En un primer experimento se pretende verificar la eficiencia y fiabilidad del método de partición creando un mapa extenso. En esta prueba, un usuario mueve la cámara mientras se desplaza en un entorno compuesto por varias habitaciones, generando un mapa denso de la escena con 500 *keyframes* y 30.000 puntos 3D[1]. Nuestra solución da lugar a 8 sub-mapas y su rendimiento no se ve afectado por el crecimiento del mapa. Por el contrario, la implementación original de PTAM no es capaz de crear el mapa completo en el mismo experimento, puesto que la precisión del sistema empeora gradualmente hasta que éste es incapaz de continuar la creación del mapa.

[1] En el siguiente enlace http://babel.isa.uma.es/efernandez/partition se puede descargar un video de nuestro sistema construyendo un mapa de un laboratorio.

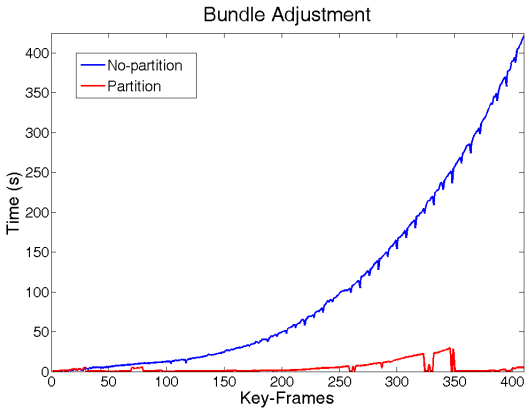

Fig. 7. Tiempos de cómputo de *bundle adjustment* con optimización completa (no en tiempo real) a lo largo de la creación del mapa con y sin aplicación de la técnica de partición.

El segundo experimento tiene por objeto demostrar la eficiencia del método propuesto. Para ello llevamos a cabo la optimización completa con *BA* (es decir, puro *SfM*) con cada nuevo *keyframe* que se añade al mapa para los casos con y sin partición, midiendo los tiempos en cada caso. Para esta prueba se graba un video que es procesado *offline* con un ordenador Intel Core-Quad Q9400 a 2.66 GHz, 4Gb de memoria y tarjeta gráfica integrada nVidia GeForce-9300. La figura 7 muestra los tiempos de optimización respecto al número de *keyframes* del mapa en ambos casos. En ella se puede apreciar una tendencia polinomial en el incremento del coste para el caso sin partición. Por el contrario, cuando se aplica partición, el coste está acotado por el tamaño del sub-mapa más grande, siendo este considerablemente menor (en nuestros experimentos el mayor sub-mapa nunca supera los 160 *keyframes*). En ambos casos se han creado mapas compuestos por alrededor de 400 *keyframes* y 27.000 puntos. La figura 8.a y 8.b muestran los mapas globales obtenidos en ambos experimentos. Visualmente se ha comprobado que ambos mapas representan fielmente el espacio de trabajo y que no hay diferencias apreciables entre ellos. En definitiva, se han obtenido los mismos mapas con tiempo de computo muchísimo menores. En la figura 7 se observa que para el caso "con partición" se produce un aumento gradual del coste entre los *keyframes* 250 y 350, acompañado de algunos escalones abruptos. Este comportamiento se debe a que un sub-mapa ha crecido significativamente más que sus vecinos, originando escalones cuando la cámara se desplaza y capta una zona de la escena cubierta por otro sub-mapa.

En un tercer experimento, comparamos la precisión del mapa generado con nuestro método (en tiempo real) con los

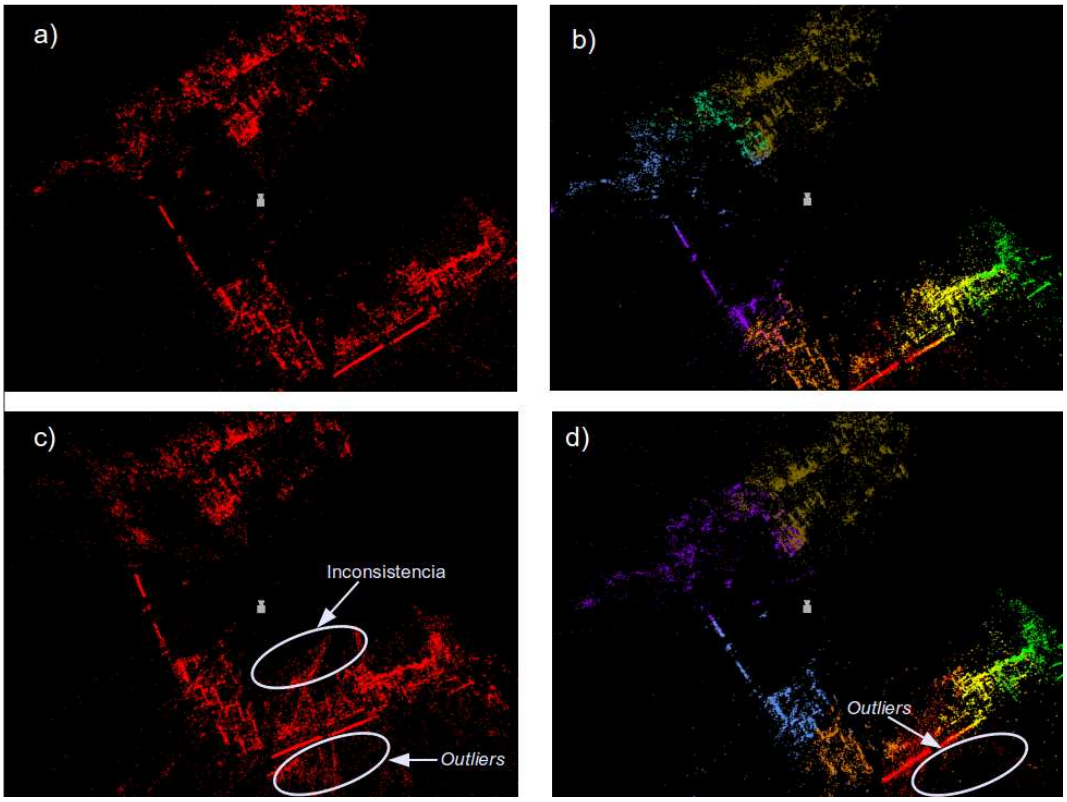

Fig. 8. Mapas generados con: a) optimización *offline* & sin partición; b) optimización *offline* & con partición; c) optimización *online* & sin partición; y d) optimización *online* & con partición.

obtenidos mediante *SfM* en el experimento anterior (figuras 8.a y 8.b), que se caracterizan por su alta precisión. Dicha comparación se realiza de forma visual, debido a la complejidad para obtener una métrica con la que evaluar la similitud entre dos mapas diferentes, y donde tampoco se dispone de un modelo de la escena con el que realizar dicha comparación. En la figura 8.d se aprecia que el mapa generado con nuestra técnica es muy similar a los mapas optimizados fuera de línea. Aun así, se observa un mayor número de espurios (*outliers*), por ejemplo, puntos detrás de paredes físicas del entorno. En el caso del mapa creado sin partición (figura 8.c), podemos ver que éste contiene inconsistencias, y además contiene significativamente más *outliers* que el generado con nuestra técnica. Esto lo atribuimos a que la optimización global *online* pierde efectividad a medida que el mapa crece, ya que dicha optimización se detiene frecuentemente antes de su finalización, lo que redunda en una pérdida de eficacia del sistema. En consecuencia, podemos afirmar que los sub-mapas generados con nuestra técnica son localmente más precisos.

V. Conclusiones y Trabajo Futuro

En este artículo se presenta una metodología de partición de mapas que, combinada con un algoritmo de SLAM monocular basado en *BA*, permite construir mapas densos de grandes entornos. Los experimentos realizados demuestran que el sistema propuesto es capaz de operar en grandes espacios, mientras que las soluciones que crean un único mapa usando este tipo de técnica están limitadas a pequeños entornos de trabajo. Otro de los beneficios de usar nuestra técnica es la representación métrico-topológica de la escena que permite una gestión eficaz del espacio y mejora la precisión de los mapas locales. Esta representación híbrida, que no se ha abordado aquí, tiene la ventaja de facilitar, por ejemplo, el cierre de bucle y la planificación de tareas en el ámbito de robótica móvil.

Las posibles líneas de trabajo futuro incluyen la detección del cierre de bucle simultáneamente con la localización y creación del mapa. Esta mejora permitiría un incremento de precisión y la posibilidad de reducir la deriva en escala. Otro problema interesante en este tipo de sistemas es la detección de oclusiones, cuya resolución

ayudaría a mejorar la calidad y eficiencia de estos algoritmos. La utilización de características de alto nivel, como las basadas en regiones planas de la escena, supone otra línea de investigación interesante, ya que este tipo de características pueden añadir robustez al sistema, a la vez que proporcionan un mayor detalle sobre la geometría de la escena.

REFERENCES

[1] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. "Consistent observation grouping for generating metric-topological maps that improves robot localization". In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 818–823, 2006.

[2] G. Klein and D. W. Murray. "Parallel tracking and mapping for small AR workspaces". In *Proceedings of the International Symposium on Mixed and Augmented Reality*, 2007.

[3] A. J. Davison. "Real-time simultaneous localisation and mapping with a single camera," In *Proceedings of the International Conference on Computer Vision (ICCV),* 2003.

[4] R. Smith, M. Self and P. Cheeseman. "A Stochastic Map for Uncertain Spatial Relationships". *4th International Symposium in Robotics Research*, p. 467-474, 1988.

[5] E. Eade and T. Drummond. "Scalable monocular slam". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 469–476, 2006.

[6] L. A. Clemente, A. J. Davison, I. D. Reid, J. Neira and J. D. Tardós. "Mapping large loops with a single hand-held camera". In *Proceedings of Robotics: Science and Systems (RSS)*, p. 153, 2007.

[7] P. Piniés and J. D. Tardós. "Large scale slam building conditionally independent local maps: Application to monocular vision". *IEEE Transactions on Robotics (T-RO)*, vol. 24, no. 5, pp. 1094–1106, 2008.

[8] J. G. Rogers and H. I. Christensen. "Normalized graph cuts for visual slam". In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[9] G. Klein and D. Murray. "Improving the agility of keyframe-based SLAM". In *Proceedings. 10th European Conference on Computer Vision (ECCV'08)*, 2008.

[10] R. Newcombe and A. Davison. "Live dense reconstruction with a single moving camera". *IEEE Conference on Computer Vision andPattern Recognition*, 2010.

[11] H. Strasdat, J. M. M. Montiel and A. J. Davison, "Real-time monocular slam: Why filter?". In *IEEE International Conference on Robotics and Automation,* 2010.

[12] S. Thrun, W. Burgard and D. Fox. "Probabilistic robotics. Intelligent Robotics and Autonomous Agents". The MIT Press, 2005.

[13] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten and S. Teller. "An atlas framework for scalable mapping". In *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1899–1906, 2003.

[14] C. Estrada J. Neira, and J. Tardos. "Hierarchical SLAM: Real-time accurate mapping of large environments". *IEEE Transactions on Robotics*, vol. 21, no. 4, p. 588-596, 2005.

[15] Z. Zivkovic, B. Bakker and B. Krose. "Hierarchical map building using visual landmarks and geometric constraints". In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, p. 2480-2485, 2005.

[16] J.-L. Blanco. "Contributions to Localization, Mapping and Navigation in Mobile Robotics". *PhD dissertation*, pp. 230–235, 2009.

[17] J. Shi and J. Malik. "Normalized cuts and image segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, p. 888-905, 2000.