Universidad Complutense de Madrid

Facultad de Informática

Doble Grado en Ingeniería Informática y Matemáticas

PERSONALIZACIÓN DE RUTAS EN SISTEMAS DE TURISMO Y OCIO

TRABAJO DE FIN DE GRADO

Jesús Aguirre Pemán
Tutor: María Belén Díaz Agudo
Tutor: Guillermo Jiménez Díaz

17 de junio de 2016

PERSONALIZACIÓN DE RUTAS EN SISTEMAS DE TURISMO Y OCIO

Autor: Jesús Aguirre Pemán
Tutor: María Belén Díaz Agudo
Tutor: Guillermo Jiménez Díaz

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

17 de junio de 2016

AUTORIZACIÓN PARA LA DIFUSIÓN DEL TRABAJO FIN DE GRADO Y SU DEPÓSITO EN EL REPOSITORIO INSTITUCIONAL E-PRINTS COMPLUTENSE

Los abajo firmantes, alumno/s y tutor/es del Trabajo Fin de Grado (TFG) en el Grado en ……de la Facultad de ……, autorizan a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el Trabajo Fin de Grado (TF) cuyos datos se detallan a continuación. Así mismo autorizan a la Universidad Complutense de Madrid a que sea depositado en acceso abierto en el repositorio institucional con el objeto de incrementar la difusión, uso e impacto del TFG en Internet y garantizar su preservación y acceso a largo plazo.

Periodo de embargo (opcional):

- ☐ 6 meses
- ☐ 12meses

TÍTULO del TFG: .....
……..
Curso académico: 20….. / 20…..

Nombre del Alumno/s:
……
……

Tutor/es del TFG y departamento al que pertenece:
……
……
……

Firma del alumno/s Firma del tutor/es

Abstract

Abstract —

The aim of this work is to provide a general and theoretical frame to personalize routes in tourism and leisure-time domains. We will also describe techniques that use Augmented Reality to improve user experience on the recommended routes.

To do this, we will introduce the necessary concepts of Graph Theory and path-finding algorithms. Afterwards, we will describe the problem of personalizing a route for individuals and groups, proposing three different approaches for its resolution: using A* algorithm, greedy algorithm, and case based reasoning, respectively. We will also analize the techniques and devices in Augmented Reality.

With the goal of seeing specific examples of the general frame exposed in this work, we will propose two case studies where we will apply the aforementioned personalization algorithms and Augmented Reality techniques. These case studies will be focused in the García Santesmases Museum of Computer Science and in the tourism domain in Madrid.

Key words — Personalization, graph, CBR, A*, greedy algorithms, Augmented Reality.

Resumen

Resumen —

El objetivo de este trabajo es proporcionar un marco general y teórico para la personalización de rutas en sistemas de turismo y ocio. Además, se verán técnicas que utilizan Realidad Aumentada para mejorar la experiencia del usuario que recorra esas rutas.

Para ello, se introducirán los conceptos necesarios de teoría de grafos, así como los algoritmos para recorrerlos. Posteriormente, describiremos el problema de personalización de rutas para individuos y grupos, proponiendo tres enfoques distintos para su resolución: utilizando el algoritmo A*, el algoritmo voraz, y el razonamiento basado en casos respectivamente. Además, se realizará un análisis sobre Realidad Aumentada, explicando los distintos dispositivos y técnicas que se utilizan en ella.

Con el objetivo de ver un ejemplo del marco general definido en este trabajo, estudiaremos dos casos en los que aplicaremos los algoritmos de personalización y técnicas de Realidad Aumentada explicadas anteriormente. Dichos casos de uso estarán centrados en el Museo de Informática García Santesmases y en el ámbito del turismo en Madrid.

Palabras clave — Personalización, grafo, CBR, A*, algoritmos voraces, Realidad Aumentada.

Índice general

Índice de figuras

1

Introducción y objetivos

1.1. Introducción

Cuando se planea una visita, ya sea a un museo, a una ciudad o a un centro comercial, las preferencias del usuario son importantes, y la ruta asociada a la visita puede personalizarse para satisfacer estas preferencias. Diferentes usuarios pueden presentar diferentes preferencias, y el problema será distinto si se considera una visita individual o una visita de grupo. Las preferencias del grupo son típicamente modeladas como la media entre las preferencias de cada uno de los miembros del grupo. Además, cuando se trata de grupos hay otros aspectos a considerar, como el número de personas, restricciones de espacio, edad media, etc.

En este trabajo vamos a describir una serie de sistemas de recomendación basados en distintos algoritmos que ayuden a planificar una ruta para una persona o un grupo, proponiendo un marco teórico reutilizable a varios dominios para el problema de planificación de rutas. Además, se utilizará Realidad Aumentada para mostrar información asociada a las recomendaciones. Trataremos dos casos de estudio, el primero centrado en el MIGS (Museo de Informática García Santesmases)[1], y el segundo en el ámbito de turismo en Madrid. Compararemos búsquedas heurísticas como el algoritmo A* o el voraz con un enfoque basado en casos (CBR, del inglés Case Based Reasoning) y discutiremos los beneficios de utilizar técnicas que usen conocimiento proporcionado por el experto del dominio.

[1]El museo, situado en la Facultad de Informática de la Universidad Complutense, honra al profesor José García Santesmases, que construyó la primera computadora en España entre 1953 y 1954.

1.2. Objetivos

Se pretende proponer un marco general y teórico que permita personalizar rutas en diferentes dominios. Para lograr este objetivo, deberemos completar varios objetivos intermedios.

1. Analizar las técnicas de personalización de rutas.
2. Analizar las posibilidades de la Realidad Aumentada y los diferentes entornos de la misma.
3. Proponer un modelo genérico reutilizable para diferentes casos de estudio.
4. Estudiar la adecuación de los algoritmos para encontrar la mejor ruta dado un grafo y las preferencias del usuario o del grupo.
5. Probar el modelo propuesto, aplicándolo a dos casos de estudio diferentes.
6. Realizar una comparativa experimental entre los algoritmos aplicados a los casos de estudio.
7. Aplicar técnicas de Realidad Aumentada a la representación visual de las rutas recomendadas.
8. Realizar un prototipo de aplicación móvil para cada caso de estudio que ayude al usuario a recorrer la ruta.

1.3. Organización del trabajo

En primer lugar estudiaremos las bases teóricas sobre las que construir los algoritmos de personalización de rutas. Haremos un análisis de la Teoría de Grafos y de los algoritmos para recorrerlos. También se definirá el concepto de Realidad Aumentada, y se indicará cómo se puede aplicar a la recomendación de ocio y turismo para mejorar la experiencia del usuario. Veremos las diferencias entre Realidad Aumentada en exteriores y en interiores, cómo se pueden solucionar los problemas de localización en interiores y los distintos entornos de desarrollo existentes.

En el Capítulo 3 propondremos un marco teórico para la recomendación de rutas personalizadas, que será adaptable a distintos casos de estudio concretos. En este capítulo formalizaremos el problema a resolver, y propondremos tres algoritmos de recomendación diferentes que permitirán utilizar búsquedas heurísticas y conocimiento experto para proporcionar las rutas personalizadas a los usuarios.

Posteriormente, estudiaremos dos casos de uso donde se han aplicado las soluciones explicadas en el capítulo anterior a problemas reales. En el primero, recomendaremos visitas personalizadas al Museo de Informática García Santesmases, utilizando técnicas

de Realidad Aumentada en interiores para proporcionar información adicional a los visitantes. En el segundo, proporcionaremos recomendaciones de ocio y turismo en Madrid, utilizando Realidad Aumentada en exteriores para guiar a los usuarios a los puntos de interés que formarán la ruta personalizada.

Finalmente, terminaremos con unas conclusiones que resuman y valoren los resultados obtenidos a lo largo del trabajo, y expondremos las líneas de trabajo futuro que permitirían mejorar las rutas personalizadas y las experiencias de Realidad Aumentada vistas en esta memoria.

2

Estado del arte

En este capítulo presentaremos la base teórica sobre la que construir los algoritmos de personalización de rutas, así como las bases de Realidad Aumentada necesarias para poder aplicar sus técnicas a las rutas propuestas. Para ello, comenzaremos con una introducción sobre Teoría de Grafos que presente las definiciones necesarias para formalizar el problema. Posteriormente veremos qué son los sistemas recomendadores y en qué categorías se subdividen. Por último, introduciremos el concepto de Realidad Aumentada y veremos qué posibilidades ofrece para mejorar la experiencia del usuario durante el recorrido de la ruta.

2.1. Personalización de rutas

En esta sección estudiaremos los conceptos de Teoría de Grafos necesarios para formalizar adecuadamente nuestro problema, así como una introducción teórica a los algoritmos de búsqueda heurística y a los algoritmos basados en conocimiento. Concluiremos con un análisis del trabajo relacionado con los ámbitos de ocio y turismo

2.1.1. Introducción a los grafos

Para poder definir formalmente el problema que queremos abordar, necesitaremos algunas definiciones previas sobre Teoría de Grafos [2].

Definición 1. Un *grafo simple (no) dirigido* $G = (V, E)$ consta de V, un conjunto no vacío de *vértices* o *nodos*, y de E, un conjunto de pares (no) ordenados de elementos distintos de V, llamados *aristas*.

Definición 2. Se dice que dos vértices u y v de un grafo no dirigido G son *adyacentes* en G si $\{u, v\}$ es una arista de G. También se dice que la arista e *conecta* u y v.

Definición 3. Sea n un entero no negativo y sea G un grafo no dirigido. Un *camino* de *longitud* n de u a v en G es una secuencia de n aristas $a_1, a_2, \ldots, a_n$ de G tal que $f(a_1) = \{x_0, x_1\}, f(a_2) = \{x_0, x_1\}, \ldots, f(a_n) = \{x_{n-1}, x_n\}$, donde $x_0 = u$ y $x_n = v$. Se dice que el camino *pasa por* los vértices $x_1, x_2, \ldots, x_{n-1}$ o también que *recorre* las aristas $a_1, a_2, \ldots, a_n$.

Ya podemos trazar caminos en grafos. Para ver las restricciones existentes sobre esos caminos, introduzcamos ahora los conceptos de *conexión* en grafos.

Definición 4. Se dice que un grafo no dirigido es *conexo* si hay un camino entre cada par de vértices distintos del grafo.

Para los grafos dirigidos, la definición equivalente es la siguiente:

Definición 5. Se dice que un grafo dirigido es *fuertemente conexo* si hay un camino de v_i a v_j y un camino de v_j a v_i para cualesquiera dos vértices v_i y v_j del grafo.

Teorema 1. *Siempre hay un camino entre cada par de vértices distintos de un grafo no dirigido conexo.*

Con las definiciones vistas, ya estamos en disposición de estudiar cómo recorrer los grafos mediante algoritmos de búsqueda.

2.1.2. Algoritmos de búsqueda en grafos

En primer lugar, la *búsqueda en el espacio de estados* es un proceso por el que se consideran sucesivos estados de una instancia con el objetivo de encontrar un *estado objetivo* con una propiedad buscada. Esta búsqueda puede ser *no informada*, como las búsquedas primero en anchura y primero en profundidad. La *búsqueda primero en anchura* es una estrategia en la que en primer lugar se *expande* (calculando sus sucesores) el nodo raíz, a continuación sus sucesores, después los sucesores de estos, y así iterativamente. Por su parte, la *búsqueda primero en profundidad* expande el nodo al nivel más profundo en el árbol de búsqueda [3].

Se denomina *búsqueda informada* a la que utiliza el conocimiento específico del problema más allá de la definición del problema en sí mismo [3]. A la aproximación general que consideraremos se le llamará *búsqueda primero el mejor*. La búsqueda primero el mejor es un caso particular del algoritmo general de búsqueda en grafos en el cual el nodo elegido para continuar con la búsqueda se basa en una función de evaluación $f(n)$.

El algoritmo A* es un algoritmo para encontrar el mejor camino en un grafo. Utiliza una función de evaluación $f(n) = g(n) + h(n)$, donde g calcula el coste del camino desde el nodo origen hasta el actual y h es la función heurística que estima el coste del camino desde el nodo actual hasta el final [4].

Este algoritmo combina las búsquedas primero en anchura y primero en profundidad [5]. La función h es la única componente utilizada en los algoritmos voraces, y g tiende a primero en anchura. De este modo, se cambia de camino de búsqueda cada vez que haya nodos más prometedores.

Al ser un algoritmo de búsqueda en anchura, A* es **completo**: en caso de existir una solución, siempre dará con ella. Para garantizar que la solución encontrada sea óptima, la función $h(n)$ debe ser una heurística *admisible*, es decir, que no sobrestime el coste real de alcanzar el nodo objetivo [4].

A* admite dos implementaciones diferentes. Nosotros utilizaremos la versión Tree-Search [6] en lugar de la Graph-Search. La diferencia entre las dos es que la Tree-Search obtiene siempre la solución óptima si la heurística elegida es admisible, mientras que para que Graph-Search llegue a la solución óptima, la heurística utilizada debe ser *consistente*, es decir, que para cada nodo v_i y su sucesor v_j se cumple $h(v_i) \leq b(v_i, v_j) + h(v_j)$. Toda heurística consistente también es admisible, de ahí que utilicemos el paradigma Tree-Search. Podemos ver el pseudocódigo de A* en el Algoritmo 1.

Algoritmo 1: A* según el paradigma Tree-Search

```
Function TreeSearch(Nodo raiz)
    PriorityQueue frontera = sucesores(raiz);
    while (|frontera| > 0) do
        nodo = extraer(frontera) ;        // elige el nodo de mejor coste estimado
        estado = estado_nodo;
        if (objetivo(estado)) then
            return solucion(nodo);
        end
        for cada sucesor de nodo do
            insertar(frontera, sucesor);
        end
    end
    return fallo;
```

Para poder aplicarlo a un problema, necesitaremos las siguientes definiciones:

- Cuál es el objetivo del problema.
- Cuáles son las acciones del problema.
- Cuál es el coste estimado de la solución.

En nuestro caso, el objetivo del problema es encontrar un camino que nos permita visitar el mayor número de nodos de acuerdo con nuestros gustos. Como hemos mencionado al introducir el algoritmo A*, el coste estimado de la solución es la función $f = g + h$. Las acciones del problema son añadir nodos al camino que estamos considerando como posible solución.

2.1.3. Sistemas recomendadores

Los sistemas recomendadores (RS, *recommender systems*) son herramientas software y técnicas que proporcionan sugerencias útiles de *items* para los usuarios [7]. El término "item" es de uso general, y se utiliza para indicar el tipo de contenido que recomienda el sistema. Los sistemas recomendadores se pueden dividir en seis categorías [8].

- Basado en contenido: identifican las características de los items que han recibido una valoración positiva un usuario y recomiendan nuevos items que tengan características similares a aquellos.
- Filtrado colaborativo: consiste en realizar recomendaciones basándose en las valoraciones de los usuarios. Cubre varias de las limitaciones del enfoque basado en contenido, puesto que algún item cuyo contenido sea difícil de expresar siempre podrá ser recomendado gracias a la valoración de los usuarios. Además, el filtrado colaborativo evita posibles problemas derivados del mal etiquetado de los objetos. A su vez, los sistemas de filtrado colaborativo pueden dividirse en 2 tipos [9].
  - Neighborhood: las valoraciones de los usuarios se utilizan directamente para predecir las valoraciones de los nuevos items.
  - Basado en modelo: utilizan las valoraciones de los usuarios para aprender un modelo predictivo. Este modelo es entrenado utilizando los datos disponibles y posteriormente es utilizado para predecir nuevas valoraciones.
- Demográfico: el sistema se basa en el perfil demográfico del usuario.
- Basado en conocimiento: la recomendación se basa en el conocimiento experto sobre cómo ciertos items satisfacen las preferencias de los usuarios.
- Basados en comunidad: este tipo de sistemas recomendadores se basan en las preferencias de los amigos del usuario, basándose en el hecho de que las personas confían más en las recomendaciones de sus amigos que en las de otros usuarios anónimos con gustos similares a los suyos.
- Híbridos: son combinación de las técnicas anteriores.

2.1.4. CBR

El Razonamiento Basado en Casos (CBR, Case Based Reasoning) consiste en resolver un problema utilizando para ello una situación pasada similar, de la que reutilizamos información y conocimiento [38]. Un ciclo CBR puede ser descrito por los siguientes cuatro procesos:

1. **Recuperar** de la base de casos el caso más similar a una consulta realizada por un usuario.

2. **Reutilizar** la información y conocimiento de ese caso para solucionar el problema.

3. **Revisar** la solución propuesta.

4. **Retener** las partes de la experiencia que puedan ser utilizadas para resolver futuros problemas.

2.1.5. Trabajo relacionado

El problema de búsqueda de rutas en el dominio del turismo ha sido abordado usando diferentes aproximaciones. Una de las mas clásicas y conocidas es el problema del camino más corto, que no tiene en cuenta preferencias de usuario. Sistemas como MacauMap [11] mezclan otros algoritmos que incluyen intereses del usuario para seleccionar los puntos de interés de la ruta recomendada, y genera la secuencia de viaje usando un algoritmo de búsqueda A*. Como en este trabajo, el recomendado considera el tiempo entre los puntos de interés y los tiempos de visita para proporcionar un recorrido óptimo.

Los sistemas recomendadores se han utilizado de forma creciente en el campo de turismo, recomendación de atracciones o puntos de interés, serviciones de viaje (restaurantes, hoteles, transportes,...), rutas y tours, o planes personalizados de varios días, entre otros [10].

Otras aproximaciones para recomendar itinerarios de turismo son categorizadas como el Problema de Satisfacción de Restricciones: el sistema recomendador genera una secuencia de puntos de interés a visitar, filtrando datos de acuerdo a las restricciones del usuario especificadas en la consulta. Este es el enfoque empleado por INTRIGUE [12], que recomienda itinerarios y destinos teniendo en cuenta las preferencias de un grupo de personas.

Varios sistemas recomendadores afrontan el problema como un problema de orientación con restricciones impuestas por los usuarios. PERSTOUR [13] recomienda rutas personalizadas utilizando la similitud entre puntos de interés y preferencias de los usuarios, inferidas de fotografías geolocalizadas. Recomienda un itinerario, con un límite de tiempo y un punto de partida y otro de salida especificados por el usuario, que maximiza la popularidad y el interés del usuario en los puntos recomendados, a la vez que se ciñe a una restricción de tiempo. Las mayores de las contribuciones de este trabajo son la personalización de la duración de la visita en cada punto, y la medición del nivel de interés del usuario en una categoría de puntos de interés basándose en el tiempo que el usuario pasa en estos puntos, relativo al usuario medio.

El interés del usuario y la recomendación de puntos de interés implican la existencia de descripción de conocimiento sobre estos puntos. El sistema T-Path [14] selecciona los itinerarios que mejor encajan con las preferencias de los usuarios clasificando los puntos de interés en categorías e infiriendo el vector de categorías utilizando una Red Bayesiana. Después, el sistema selecciona los itinerarios que son más similares al modelo del usuario, representado por un vector de categorías creado con la información coleccionada de las interacciones con el usuario.

El uso de aproximaciones CBR en la recomendación de itinerarios turísticos no es comúnmente empleado a pesar de que la planificación basada en casos es un campo que presenta resultados satisfactorios. Aunque TURAS [15] no proporciona recomendaciones de turismo, el trabajo sobresale por ser un sistema de planificación de rutas basado en casos que genera rutas personalizadas reutilizando partes relevantes de múltiples casos.

Otro sistema específico de recomendación de rutas turísticas basada en casos es TOURIST GUIDE-USAL [16], que genera la ruta adaptando casos previos al perfil del usuario descrito por características como el tipo de visita, el presupuesto o la duración. En este sistema, ls casos son rutas anteriores que incluyen los puntos de interés a vista, el tiempo a pasar en cada punto, el tiempo requerido para viajar de un punto a otro y etiquetas de las rutas (como ruta de museos, ruta de familia, ruta Románica, etc). Este trabajo inspiró nuestra aproximación a la planificación de rutas usando CBR.

2.2. Realidad Aumentada

La Realidad Aumentada (RA) es una técnica que permite mezclar elementos del mundo real con elementos digitales, integrando ambas partes en una sola vista [17]. Aunque sus inicios se remontan a la década de 1960, en los últimos años se ha producido una revolución en esta tecnología, motivada en gran parte por la aparición de los *smartphones*, que nos permiten visualizar la Realidad Aumentada sin necesidad de utilizar complejos sistemas como los dispositivos montados sobre la cabeza que se usaban en los comienzos de esta tecnología. El crecimiento de la RA ha permitido que el rango de sus aplicaciones se haya expandido notoriamente: anuncios, entretenimiento, educación, uso médico y un largo eccétera.

2.2.1. Definición

La Realidad Aumentada (RA) se define como una vista en tiempo real de un entorno real que ha sido *mejorado/aumentado* al añadirle información virtual generada por ordenador. El objetivo de la Realidad Aumentada es facilitar las tareas del usuario proporcionándole información [17].

Mientras que la Realidad Virtual sumerge por completo al usuario en el Entorno Virtual sin que pueda percibir el mundo real, la Realidad Aumentada *aumenta* el sentido de la realidad superponiendo objetos virtuales sobre el entorno real en tiempo real. En lugar de considerar la Realidad Aumentada y la Realidad Virtual como conceptos diametralmente opuestos, es conveniente verlos como los extremos de un continuo (*continuum*) [18], que podemos observar en la Figura 2.1.

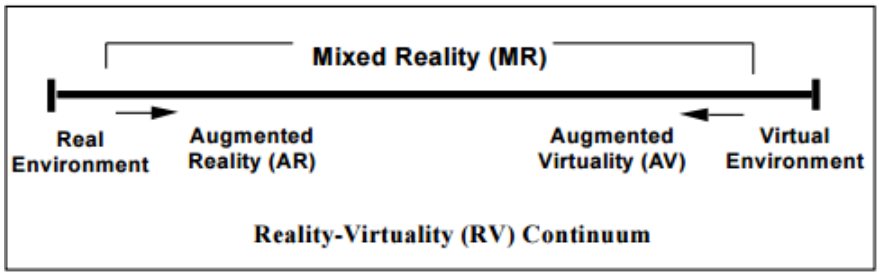

Figura 2.1: Continuo Realidad-Virtualidad

2.2.2. Historia

La primera aparición del concepto "Realidad Aumentada" se remonta a la década de 1950, cuando el cineasta Morton Heilig patentó el Sensorama, un simulador que proporcionaba una ilusión de realidad utilizando vídeo en 3D, olor, vibraciones en el asiento y viento, involucrando así la mayor parte de los sentidos [19]. En 1968, Ivan Sutherland creó un sistema de realidad Aumentada utilizando un dispositivo de visualización HMD (visualización montada en la cabeza), que coloca tanto las imágenes del mundo real como las del entorno virtual sobre la vista del usuario.

En los años 90, la Realidad Aumentada experimentó un gran crecimiento. En 1992 se presenta el primer paper sobre un prototipo de sistema RA (llamado KARMA), y dos años más tarde, Milgram y Kishino definirían el continuo realidad-virtualidad visto anteriormente. En 1997, Ronald Azuma escribe el primer estudio sobre Realidad Aumentada, donde describe los principales campos de aplicación de estas técnicas, así como los distintos tipos de RA y su funcionamiento [20]. Este estudio se convertiría en el texto más importante sobre Realidad Aumentada hasta la actualidad. En 1999 Hirokazu Kato desarrolla ARToolKit[1], el primer SDK de Realidad Aumentada, que posteriomente será publicado por la Universidad de Washington.

Durante la década de 2000, se realizan diversos proyectos de RA con ARToolKit . En 2003 se presenta un sistema de guía en interiores utilizando RA [21]. La aplicación utiliza un dispositivo portátil y ARToolKit para proporcionar una vista aumentada del entorno. En 2006, Reitmayr y Drummond presentan un sistema de *tracking* para RA en exteriores, utilizando un dispositivo portátil [22]. El sistema combina tracking basado en aristas, medidas del giroscopio, la gravedad y el campo mágnetico para proporcionar una localización precisa. Un año más tarde, Klein y Murray presentan un sistema capaz de realizar las tareas de tracking y mapping paralelamente, en dos hilos separados [23]. Es la primera aproximación a la técnica de SLAM, que describiremos posteriormente.

A finales de la década, coincidiendo con el *boom* de los *smartphones*, comienzan a aparecer más entornos de desarrollo. En 2008, la compañía Metaio presenta una guía de museo con RA. Este mismo año nace Wikitude, que en sus inicios combina los datos del GPS y la brújula con entradas de Wikipedia para proporcionar información obtenida de

[1] `https://www.artoolkit.org/`

esta última sobre las imágenes en tiempo real de la cámara. El siguiente año se produciría el lanzamiento de Layar, con similares funcionalidades que Wikitude. A partir de este momento, comienzan a multiplicarse las aplicaciones de Realidad Aumentada. En 2013 Google anuncia sus gafas de RA, llamadas Google Glasses. A su vez, Microsoft anuncia su proyecto de HoloLens en 2015, con lo que se prevén grandes avances en el campo de la Realidad Aumentada durante los próximos años.

2.2.3. Usos

Las tecnologías de Realidad Aumentada se pueden aplicar en una gran variedad de campos, entre los que destacan:

- **Medicina.** Los médicos pueden utilizar la Realidad Aumentada como medio para visualizar y recibir asistencia en operaciones quirúrgicas, como con Freehand Spect [24] (ver Figura 2.2). También se puede utilizar para entrenar a cirujanos noveles, donde las instrucciones virtuales les ayudarían con las tareas sin necesidad de apartarse del paciente para consultar un manual [20]. El problema radica en la falta de conocimiento sobre estas tecnologías por la mayoría de médicos, a pesar de que ya existan aplicaciones como *Evena's Eyes-On Glasses*[2], que permiten visualizar las venas de los pacientes en tiempo real, o *CamC*, que ofrece una tecnología basada en Realidad Aumentada que proporciona imágenes de rayos X sin realizar tracking adicional, reduciendo la exposición a la radiación. Por último, Mirracle es un espejo que utiliza Realidad Aumentada para crear la ilusión de que el usuario está viendo el interior de su cuerpo, pudiendo realizar gestos para ver secciones de los órganos [25].

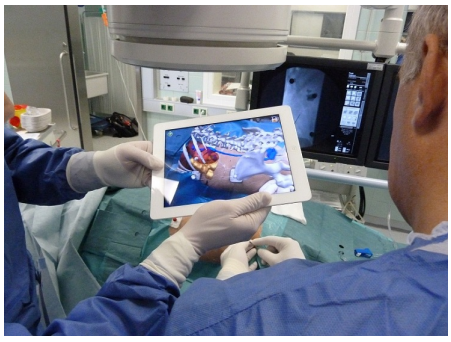

Figura 2.2: Aplicación de las técnicas de RA en medicina

- **Maquinaria.** La tecnología de RA también se puede utilizar para ayudar en el ensamblaje, mantenimiento y reparación de maquinaria. Sobreponer las instrucciones como dibujos 3D sobre la propia maquinaria facilita estas tareas, permitiendo al usuario centrarse en la pieza. Un ejemplo de esto es el sistema ARMAR, que mediante un HMD aumenta la vista del sistema, con información sobre

[2] `http://evenamed.com/eyes-on-glasses/`

los componentes, guías de mantenimiento paso a paso o advertencias de seguridad [26].

- **Entretenimiento.** Desde la aparición de ARQuake en 2001, el primer juego utilizando RA, uno de los campos de mayor presencia de la Realidad Aumentada es el del entretenimiento. Existen juegos como *Invizimals*[3] (para Play Station Portable), creado por la compañía española *Novarama* y lanzado en 2009. En este juego, el objetivo es capturar distintas especies de criaturas y luchar con ellas frente a otros jugadores. Para capturarlas, *Invizimals* utiliza la cámara de la consola y un dispositivo de forma cuadrada como marcador. Últimamente ha conseguido mucha popularidad la aplicación MSQRD[4], que reconoce el rostro del usuario y superpone efectos especiales sobre él.

- **Arquitectura.** Mediante RA se pueden crear fácilmente prototipos de los diseños arquitectónicos, e incluso ver el diseño sobre un edificio existente en tiempo real. En la Figura 2.3 se observa una fachada aumentada mediante la técnica SLAM (Simultaneous Localization And Mapping) que veremos posteriormente.

Figura 2.3: Aplicación de las técnicas SLAM en edificios (Wikitude)

- **Publicidad.** La Realidad Aumentada también ha sido utilizada en campañas de marketing. En 2008, la compañía de automóviles Mini lanzó un anuncio en revistas donde era posible visualizar un coche 3D utilizando la webcam del ordenador [27]. La aplicación que permite consultar el catálogo de muebles de IKEA[5] también incluye una funcionalidad que permite superponer un mueble en una habitación.

- **Museos.** La mayoría de las aplicaciones de RA para museos sólo sirven para uno en concreto, puesto que dependen de la organización del museo para reconocer los ítems. Es decir, en este momento no es posible detectar las formas de los ítems, frecuentemente por su irregularidad, y hay que utilizar sistemas de posicionamiento en interiores (IPS) o marcadores. El museo Smithsonian de Historia Natural proporciona una aplicación llamada *Skin & Bones*[6] que superpone la piel del animal sobre los huesos expuestos en las salas del museo conocidas como *The Bone Hall*,

[3] `http://novarama.com/invizimals/`
[4] `http://msqrd.me/`
[5] `http://www.ikea.com/ms/es_ES/apps/mobileApps.html`
[6] `http://naturalhistory.si.edu/exhibits/bone-hall/`

donde se encuentran una gran variedad de esqueletos de la mayoría de especies vertebradas.

- **Moda.** Magic Mirror[7] es un sistema de RA que permite al usuario probarse virtualmente ropa antes de comprarla. Así, el usuario puede cambiar fácilmente detalles de las prendas, como el color o la talla.

- **Ocio y turismo.** Existen varios sitemas que utilizan RA para reconstruir virtualmente monumentos en ruinas, o para proporcionar información sobre el monumento en cuestión, como Archeoguide [28]. Podemos ver la aplicación de estas técnicas en la Figura 2.4. A la izquierda aparece el monumento en su estado actual, y a la derecha se superpone el estado del monumento en la Antigüedad. Recomendadores de turismo como Yelp[8] ofrecen la posibilidad al usuario de explorar la zona donde se encuentra proporcionándole carteles que mediante Realidad Aumentada le ofrecen información sobre localizaciones cercanas.

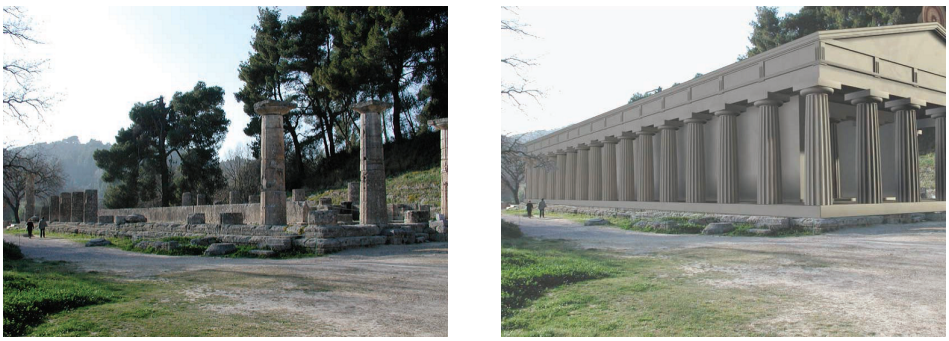

Figura 2.4: Archeoguide: Realidad Aumentada para reconstruir monumentos

2.2.4. Funcionamiento

Los métodos de Realidad Aumentada constan de 2 fases: rastreo(*tracking*) y reconstrucción. En primer lugar, la cámara detecta los marcadores, imágenes óptimas y puntos de interés. La operación de rastreo puede utilizar varias técnicas, como detección de bordes o de características. La etapa de reconstrucción utiliza los datos obtenidos durante el *tracking* para reconstruir un sistema de coordenadas en el mundo real.

En un sistema de Realidad Aumentada se realizan cuatro tareas para aumentar el entorno real, que son captación de escena, identificación de escena, mezclado de realidad y aumento, y por último, visualización. La captación de la escena se realiza mediante los dispositivos anteriormente mencionados. A continuación se exponen diferentes métodos para identificar la escena:

[7] `http://www.magicmirror.me/Apps/Virtual-Try-On`
[8] `http://www.yelp.com/`

Marcadores

Detectar la posición de la cámara atendiendo sólamente a los elementos que percibimos mediante ella es una tarea complicada. Por ello, se utilizan los *marcadores*, signos o imágenes que un sistema puede detectar de un vídeo utilizando procesado de imágenes, reconocimiento de patrones y técnicas de visión computerizada. Una vez detectado el marcador, el sistema determina la escala correcta de la escena y la posición de la cámara. Esta aproximación se denomina *tracking basado en marcadores* [17].

El procedimiento básico de detección de marcadores consiste en cinco pasos: adquisición de la imagen, preprocesamiento, detección de potenciales marcadores, identificación de los marcadores y cálculo de la posición del marcador.

Otras técnicas para el tracking visual son las basadas en *modelos* y las basadas en *características*. En las primeras, el sistemas tiene un modelo de parte de la escena, y compara las observaciones de la cámara con el modelo, averiguando así la posición de la cámara. En el seguimiento basado en características, el sistema detecta las características ópticas en las imágenes y aprender el entorno basándose en observaciones del movimiento entre *frames* en la cámara.

SLAM

SLAM (Simultaneous Localization And Mapping) es un proceso mediante el cual se puede construir un mapa del entorno y, al mismo tiempo, utilizar ese mapa para calcular su posición [29]. Sus aplicaciones a la Realidad Aumentada son de gran utilidad, puesto que podremos aumentar directamente las imágenes que recibimos.

Wikitude ha comenzado a implementar esta funcionalidad en su SDK, y aunque de momento sólo está disponible para pequeños entornos, pronto permitirá el *tracking* de grandes escenas que incluyan varias habitaciones. El SDK de Wikitude rastrea las escenas 3D identificando puntos característicos de los objetos y el entorno. Por tanto, cuanto más rica sea la escena a identificar, es decir, más puntos característicos contenga, mejor será el rastreo e identificación [30].

Sistema de Posicionamiento

En entornos exteriores es sencillo posicionar los puntos de interés mediante sus coordenadas GPS. Para realizar esta tarea en interiores, introduciremos el concepto de Sistema de Posicionamiento en Interiores (IPS, Indoor Positioning System). Estos sistemas permiten rastrear y obtener la posición en tiempo real de personas u objetos dentro de un edificio utilizando ondas de radio, campos magnéticos y otras señales captadas por los dispositivos móviles [31]. Para esta tarea también se pueden utilizar *beacons*, transmisores Bluetooth que son capaces de comunicarse con dispositivos en un rango cercano. De esta forma, la posición se determina en base a qué *beacons* son detectados por el dispositivo del usuario [32].

IndoorAtlas[9] ofrece un IPS capaz de obtener la localización en interiores utilizando para ello el sensor magnético de los *smartphones*. Permite crear localizaciones y rutas dentro de ellas que podemos usar para calibrar la localización, y utilizar así el posicionamiento en interiores en una aplicación Android o iOS. Teniendo ya la localización en interiores resuelta, podemos integrarla en nuestra aplicación y utilizar las tecnologías de RA basadas en Geolocalización vistas anteriormente. En la Figura 2.5 podemos ver un mapa de la cuarta planta del Museo García Santesmases calibrado mediante esta tecnología (sólo se muestra la parte correspondiente a las vitrinas del museo). Lamentablemente, la forma del museo imposibilita una detección adecuada de la posición del usuario.

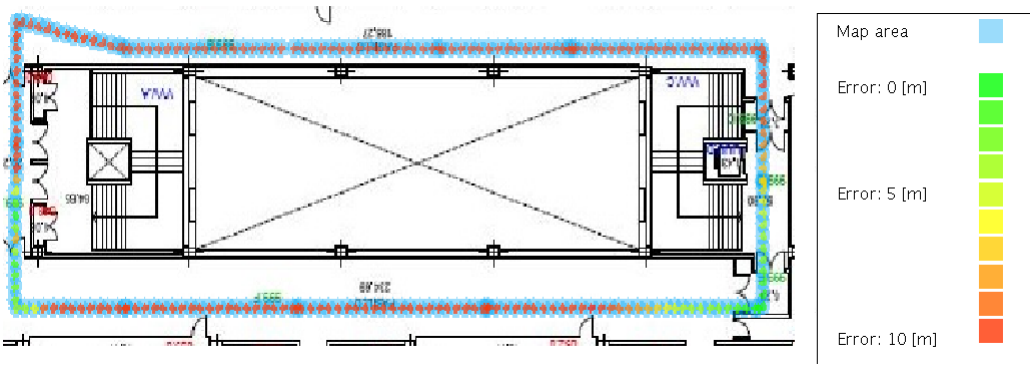

Figura 2.5: Mapa de la cuarta planta del museo utilizando IndoorAtlas

2.2.5. Entornos de desarrollo

Para llevar a cabo uno de los objetivos de este trabajo se han estudiado diferentes entornos para desarrollar aplicaciones de Realidad Aumentada.

Wikitude

El SDK de Wikitude[10] ofrece posibilidades en varios lenguajes de programación, entre los que destacan la API nativa para Android, la API de JavaScript (basada en HTML y JavaScript) y el plugin para Unity. Tras evaluar las características de cada uno, se optó por la API de JavaScript, pues era la que más funcionalidades ofrecía.

Una vez decidido el entorno de desarrollo, Wikitude ofrece varios ejemplos para probar las distintas herramientas de Realidad Aumentada. Entre las funcionalidades más destacadas se incluyen:

- Reconocimiento de imágenes.

[9] `https://www.indooratlas.com/`
[10] `http://www.wikitude.com/`

- Carga de modelos 3D, diseñados en Autodesk o Blender.
- Creación de marcadores en determinadas geolocalizaciones.
- Reconocimiento de objetos (en desarrollo).

Wikitude permite solicitar una licencia de uso educativo a través de Wikitude Academy[11]. Esto permite utilizar todas las funcionalidades de la SDK Pro de forma gratuita durante un periodo de hasta un año.

Vuforia

La plataforma Vuforia[12] permite reconocer imágenes, texto, marcadores fiduciales, formas geométricas como cubos o cilindros e incluso objetos como juguetes. Además, ofrece posibilidades interesantes como *Smart Terrain*, que permite crear un entorno tridimensional a partir del entorno real, donde los elementos virtuales interactúan con los reales. Se puede ver un ejemplo de esta tecnología en la Figura 2.6. También cuenta con la funcionalidad de *Extended Tracking*, que permite continuar con el rastreo aun cuando el marcador se encuentra fuera de la vista.

Figura 2.6: Smart Terrain de Vuforia

2.2.6. Realidad Aumentada en ocio y turismo

Una vez que hemos generado la recomendación, nuestro objetivo será hacer esa recomendación más atractiva para el usuario, a la vez que le facilitamos la tarea de recorrer la ruta que le hemos proporcionado. En este punto es donde entra en juego la Realidad Aumentada. Mientras que hace unos años la Realidad Aumentada requería de complejos dispositivos, hoy en día basta con un *smartphone* para poder disfrutar de sus contenidos. Esta forma de representar contenidos, que se superpone a la información del entorno real,

[11] `http://www.wikitude.com/wikitude-academy/`
[12] `https://www.vuforia.com/`

representa una ventaja sobre otras representaciones más convencionales como pueden ser listas de objetos o mapas.

Pocas aplicaciones combinan la recomendación de rutas con la Realidad Aumentada. MoreTourism [33] utiliza etiquetas y pesos para recomendar puntos de interés. La recomendación se realiza comparando la nube de etiquetas del usuario con la nube de etiquetas de los puntos. Este trabajo también propone recomendaciones para grupos creando una nube de etiquetas de grupo usando las nubes de etiquetas de sus miembros. La información se presenta integrando la información, imágenes y videos mediante servicios de Realidad Aumentada.

Otra aplicación que combina estas dos técnicas es RAMCAT [34], que presenta un sistema recomendador híbrido basado en contexto orientado a integrar Realidad Aumentada en dominios turísticos. La aplicación recomienda puntos de interés teniendo en cuenta preferencias personales y atributos contextuales. Existen varios módulos para estas recomendaciones: filtrado colaborativo, contextual, por contenido, etc. La información se puede presentar tanto aumentada como en un formato convencional de lista de puntos de interés. Además, el sistema se retroalimenta mediante las calificaciones de los turistas.

En [35], los autores presentan una herramienta de aprendizaje para el ámbito académico que recomienda contenido en función de los intereses del usuario, presentándolo mediante una interfaz basada en Realidad Aumentada. Los intereses del usuario se extran del comportamiento del usuario al interactuar con el contenido presentado en la interfaz.

Respecto a guiar a los usuarios hasta un punto, es interesante el enfoque de la aplicación Penguin NAVI[13], que utiliza a un grupo de pingüinos como guías para llevar a los usuarios hasta el Sunshine Aquarium de Tokio. El éxito de esta aplicación supuso un aumento de la afluencia del 150 % con respecto al mes anterior.

2.3. Conclusiones

En este capítulo hemos visto las principales nociones teóricas necesarias para la formalización del problema de la personalización de rutas. En primer lugar, se han introducido algunos conceptos sobre grafos que nos servirán para caracterizar el mapa del dominio, y sobre algoritmos de búsqueda para obtener caminos en el grafo. También hemos introducido los sistemas recomendadores y el razonamiento basado en casos. Por último, hemos estudiado las diferentes posibilidades que ofrecen las técnicas de Realidad Aumentada a a hora de mejorar la experiencia del usuario.

Esto nos permitirá guiar al usuario por el recorrido recomendado o ampliar la información disponible. Las técnicas vistas servirán para implementar elementos de Realidad Aumentada en las recomendaciones. Si se trata de un dominio cuyas rutas son en interiores, deberemos utilizar técnicas basadas en marcadores, o utilizar un Sistema de Posicionamiento en Interiores para la localización. Por el contrario, en un entorno al aire libre podremos hacer uso del GPS para localizar nuestra posición y la de los puntos

[13] `http://penguinnavi.erba-hd.com/`

que deberemos visitar, aunque también tengamos la posibilidad de utilizar marcadores. Veremos las implementaciones de estas tecnologías en los casos de estudio del Museo García Santesmases (Sección 4.1) y del recomendador de turismo en Madrid (Sección 4.2).

3

Marco teórico para la personalización de rutas

En este capítulo se propondrá un marco teórico para la recomendación de rutas personalizadas. Este marco, que será adaptable a cualquier caso de estudio concreto, admitirá tanto recomendaciones individuales como grupales, y será aplicable a dominios con muchos puntos de interés seleccionables o con tan sólo unos pocos. Por otra parte, las rutas podrán ser en entornos interiores, como un museo o un centro comercial, o exteriores como en una ciudad. Etiquetaremos los puntos de interés según sus características e introduciremos las preferencias del usuario, creando itinerarios en los que, respetando un límite de tiempo, se visiten los puntos de interés cuyas etiquetas mejor se adecúen a estos gustos.

3.1. Formalización del problema a resolver

El objetivo de nuestros algoritmos de planificación de rutas será encontrar una ruta para un individuo o un grupo de personas que recorra una secuencia de puntos de interés categorizados con diferentes etiquetas. La ruta debe maximizar la satisfacción de las preferencias del individuo o del grupo, descritas como un conjunto de pares de etiquetas y sus respectivos valores asociados. Además, la ruta ha de ajustarse al tiempo máximo proporcionado para la visita.

Formalicemos ahora el problema a resolver. Sea $V = \{v_1, \ldots, v_p\}$ un conjunto no vacío de puntos de interés, y sea $K = \{k_1, \ldots, k_n\}$ un conjunto de etiquetas que describen tanto las características de los anteriores puntos de interés como las preferencias del grupo. Las etiquetas son dependientes del dominio. Cada punto de interés v_i está caracterizado por

un vector de preferencias $v_i.cov = \{\lambda_{k_1}, \ldots, \lambda_{k_n}\}$, con $\lambda_{k_j} \in [0,1]$, que representan el grado en que v_i cubre la etiqueta k_j; y un valor t_{v_i} que representa el tiempo recomendado para visitar el punto v_i.

Cada punto de interés tiene una localización. El mapa que representa las localizaciones de los puntos de interés está caracterizado como un grafo no dirigido $G = (V, E)$, donde E es un conjunto de aristas definidas como pares de puntos de interés $< v_i, v_j >$ donde $v_i, v_j \in V$. Cada arista tiene asociado un valor $t_{<v_i,v_j>}$ que representa el tiempo empleado en desplazarse desde el punto v_i al punto v_j. Para la implementación de los algoritmos podremos asumir que G es un grafo dirigido, ya que el coste de recorrer una arista en sentido directo o inverso es el mismo. G es un grafo fuertemente conexo ya que para cualquier par de vértices v_i y v_j existe un camino de v_i a v_j de v_j a v_i.

Un grupo de visitantes (la consulta en el algoritmo CBR) está caracterizado por una tupla $(size_q, age_q, pref_q, t_q)$, donde *size* es el número de personas en el grupo, *age* es la edad media de sus miembros, $pref_k = \{\lambda_{k_1} \ldots \lambda_{k_q}\}$ con $\lambda_{k_i} \in [0,1]$ es un conjunto de pesos que indican las preferencias del grupo para cada etiqueta y t_q es la restricción temporal que indica la duración máxima de la visita. En el caso de que se trate de un único visitante, estará caracterizado por la tupla $(pref_q, t_q, age_q)$. La solución del problema será una *ruta*, una secuencia ordenada de puntos de interés $R = \{v_1, .., v_m\}$ con m paradas, donde $v_i \in V$, y $t_R = \sum_{i=1}^{m} t_{v_i} + \sum_{i=1}^{m-1} t_{<v_i,v_{i+1}>}$, con $t_R \leq t_q$.

Ahora ya sabemos cómo relacionar los gustos del usuario con los puntos a visitar, pero necesitaremos una función que indique qué puntos son mejores basándose en estos gustos. El problema de cómo medir el grado en el que una ruta cubre las preferencias del usuario (definidas como un conjunto de etiquetas asociadas a unos pesos) no es trivial. De acuerdo con [1], definimos el valor de *cobertura* como el grado en que las preferencias del usuario son cubiertas por la ruta escogida.

Si tomamos $K = \{k_1, \ldots, k_n\}$ como el conjunto de etiquetas anteriormente definido y λ_{k_q} el valor asociado a la etiqueta k_q, la función de cobertura de un camino R es:

$$kc(R) = \sum\nolimits_{k_q \in K} \lambda_{k_q} \cdot cov(k_q, R)$$

siendo $cov(k_q, R)$ el grado en el que la etiqueta k_q está cubierta en el camino R, dado por:

$$cov(k_q, R) = 1 - \prod\nolimits_{v_i \in R} (1 - v_i.cov(k_q))$$

Esta función de cobertura cumple que es *submodular*, es decir, que

$$kc(R_1 \cup v_i) - kc(R_1) \geq kc(R_2 \cup v_i) - kc(R_2)$$

con $R_1 \subseteq R_2$ y $v_i \notin R_1$. Esta propiedad se utilizará para la demostración de la admisibilidad de h.

El valor $kc(R)$ usado es adecuado para rutas con un número reducido de puntos de

interés. No obstante, para grafos grandes la diferencia entre los valores de cobertura de las rutas correspondientes a visitar todos los objetos del museo frente a las rutas que visitan tan solo la mitad es muy pequeña. Por esta razón definimos una nueva función de cobertura:

$$kc\_sum(R) = \sum_{v_i \in R} v_i.cov(K) \tag{3.1}$$

donde $v_i.cov(K)$:

$$v_i.cov(K) = \sum_{k_j \in K} \lambda_j \cdot v_i.cov(k_j)$$

Esta función no tiene en cuenta la diversidad de las categorías visitadas en la ruta, pero como asumimos que visitamos una gran cantidad de puntos de interés, esta pérdida no supone un gran problema. Además, nos permitirá añadir puntos a la ruta con más facilidad, pues la cobertura de cada punto ya no depende del resto de la ruta.

Nótese que kc_sum también es una función submodular, ya que:

$$kc\_sum(R_1 \cup v_i) - kc\_sum(R_1) = v_i.cov(K)$$

3.2. Desarrollo del problema

Con las definiciones que hemos proporcionado, estamos en disposición de formular un algoritmo para planificación de rutas personalizadas en un grafo. Este algoritmo tendrá en cuenta el contenido de cada nodo, personalizando el camino elegido en función de las preferencias del usuario. Por tanto, además del grafo, formado por el conjunto de vértices V y el conjunto de aristas E, el algoritmo necesitará estas preferencias ($pref_q$).

3.2.1. Algoritmo A* modificado

El primer algoritmo que utilizaremos para encontrar la ruta óptima dentro del grafo es una modificación del Algoritmo A*, puesto que así obtendremos la solución óptima para la heurística que hayamos elegido, siempre y cuando esta sea admisible. Para ello, seguimos el algoritmo ORS-KC presentado en [1]. Para implementar el algoritmo A* necesitaremos definir la función de evaluación f, que será:

$$f^n(R_{s\to t}) = g^n(R_{s\to n}) + h^n(R_{n\to t}|R_{s\to n}),$$

donde g calcula el valor exacto de cobertura de la ruta $R_{s\to n}$, mientras que h estima el valor de cobertura del resto de la ruta condicionado a la parte ya recorrida. Pasemos ahora a definir la función heurística h y demostrar su admisibilidad.

Admisibilidad de la función heurística

Como la función kc es submodular, h es la cobertura del resto de la ruta $R_{s\to n}$, en lugar de la cobertura de la ruta $R_{n\to t}$. Así, en lugar de estimar $kc(R_{s\to n})$, calcularemos $kc(R_{s\to n}) - kc(R_{s\to n})$. Tal y como se demuestra en [1],

$$kc(R_{s\to n}) - kc(R_{s\to n}) \leq (\max_{k_q \in K} \prod_{vi \in R_{s\to n}} [1 - cov_{k_q}(v_i)]) \cdot kc(R_{n\to t}),$$

donde definimos $h^n(R_{n\to t}|R_{s\to n}) = (\max_{k_q \in K} \prod_{vi \in R_{s\to n}} [1 - cov_{k_q}(v_i)]) \cdot kc(R_{n\to t})$.

Para asegurarnos de la admisibilidad de h, calcularemos una cota superior de $kc(R_{n\to t})$ utilizando el algoritmo voraz, cuya implementación concreta veremos más adelante en el Algoritmo 3. Este algoritmo alcanza un factor de aproximación de $1 - \frac{1}{\sqrt{e}}$, es decir,

$$kc(L^{op}) \leq \frac{kc(L^v)}{1 - \frac{1}{\sqrt{e}}},$$

donde L^{op} es el conjunto óptimo de nodos mientras que L^v es el conjunto de nodos calculados por el algoritmo voraz.

Se puede encontrar en [1] la prueba de que $kc(R_{n\to t}) \leq \frac{kc(L^v)}{1-\frac{1}{\sqrt{e}}}$, con lo que la definición de h queda

$$h^n(R_{n\to t}|R_{s\to n}) = (\max_{k_q \in K} \prod_{vi \in R_{s\to n}} [1 - v_i.cov(K)]) \cdot \frac{kc(L^v)}{1 - \frac{1}{\sqrt{e}}},$$

que cumple la admisibilidad requerida por A*, pues hemos visto que no estima por encima del coste real.

Por tanto, finalmente f se calcula como

$$f^n(R_{s\to t}) = kc(R_{s\to n}) \cdot (\max_{k_q \in K} \prod_{vi \in R_{s\to n}} [1 - v_i.cov(K)]) \cdot \frac{kc(L^v)}{1 - \frac{1}{\sqrt{e}}}$$

Descripción del algoritmo

Una vez definida la función heurística h y demostrada su admisibilidad, podemos explicar el funcionamiento del Algoritmo 2. El algoritmo comienza creando una ruta vacía con valor de cobertura $-\infty$, donde almacenaremos la mejor ruta que vayamos calculando. También crearemos una cola de prioridad de rutas ordenada por valor decreciente de la función f. Mientras esta cola no esté vacía, tomaremos el primer elemento y comprobaremos que la estimación de su valor heurístico es mayor que el valor de cobertura de la mejor ruta hallada hasta el momento. Si no lo fuera, la ruta guardada sería la óptima, ya que al tratarse de una cola de prioridad, todas las rutas almacenadas en ella tendrán un valor heurístico igual o peor que la que hemos tomado.

Si el valor heurístico de R_i^k es mayor que la mejor cobertura encontrada hasta el momento($kcmax$), añadiremos un nuevo nodo a esa ruta, asegurándonos de no sobrepasar el tiempo máximo t_q del que disponemos para recorrer la ruta. Si el nodo que hemos añadido es el último vértice del grafo, comprobaremos si el valor heurístico de la ruta obtenida es mayor que el de la ruta que teníamos guardada como mejor, y en caso afirmativo la sustituiremos por la actual. Si la función heurística de la ruta actual proporciona un valor mayor que el de la ruta almacenada, la guardaremos como la mejor y si no hemos llegado al último nodo la almacenaremos en la cola de prioridad. Cuando no queden más nodos en la cola de prioridad Q habremos acabado, y devolveremos la ruta R almacenada.

Algoritmo 2: ORS-KC [1]

```
MaxPriorityQueue Q = ∅;
Route R = ∅;
double kcmax = −∞;
Route R_0^s = (v_s) ;
Q.push(R_0^s);
while Q is not empty do
    R_i^k = Q.get();
    if f(R_i^k) ≤ kcmax then
        break;
    end
    for cada arista (v_i, v_j) do
        R_j^l = R_i^k.add(v_j);
        if bs(R_j^l > t_q) then
            continue;
        end
        if (v_s = v_t) then
            if (kc(R_j^l) > kcmax) then
                R = R_j^l;
                kcmax = kc(Rjl);
            end
        else
            if f(R_j^l > kcmax) then
                Q.push(R_j^l);
            end
        end
    end
end
if (kcmax == −∞) then
    return "No hay ruta posible";
else
    return R;
end
```

Veamos un ejemplo que nos ayude a entender las definiciones y el algoritmo. Consideremos el grafo de la Figura 3.1, y supongamos las siguientes preferencias del usuario: *Historia*: 0,6; *entretenimiento*: 0,4; *museo*: 0,2; *monumento*: 0,1.

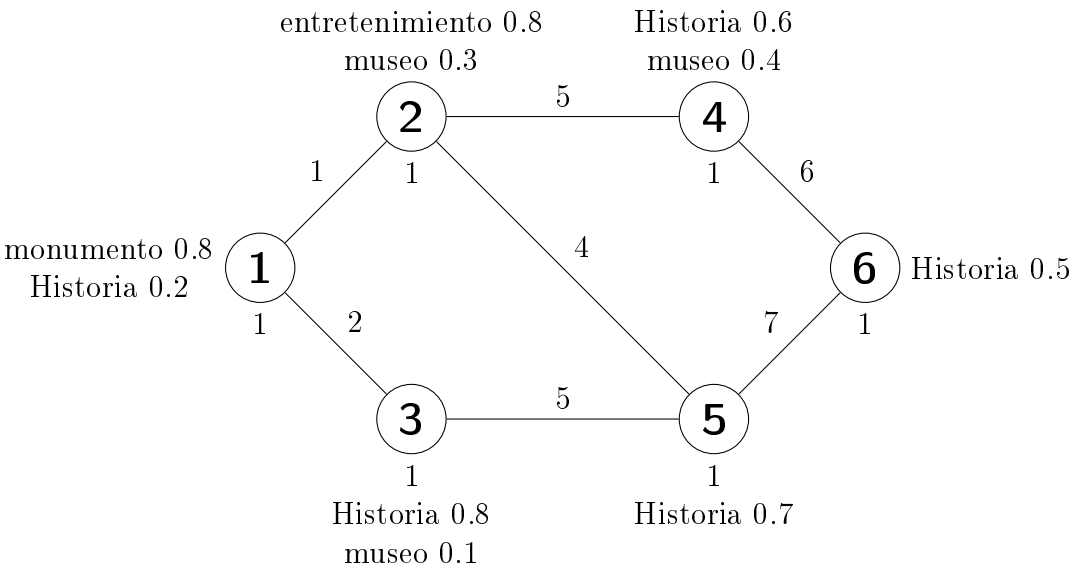

Figura 3.1: Grafo de ejemplo

Analicemos los caminos superior (vértices 1, 2, 4 y 6) e inferior (1, 3, 5 y 6). El camino inferior a priori nos proporcionaría mayor satisfacción, pues la etiqueta *historia* es la que mayor ponderación tiene entre los gustos del usuario. No obstante, como hemos comentado, el algoritmo A* tiene en cuenta la diversidad de las etiquetas de la ruta, por lo que escogerá el camino superior.

Al tratarse de una variante del algoritmo A*, que es completo, es comprensible que el tiempo de cálculo para obtener la solución sea grande en dominios con muchos puntos de interés. La ejecución del algoritmo A* tarda más de 10^4 segundos en un grafo de 22 vértices [36]. Esto puede ocasionar problemas en las aplicaciones prácticas del recomendador, como puede ser el uso en una aplicación móvil. Para ello, se han propuesto diferentes algoritmos que, sin ser tan exhaustivos como A*, siguen proporcionando buenas soluciones.

3.2.2. Algoritmo voraz

Recordemos que en el algoritmo A* necesitamos hacer una estimación del camino restante hasta alcanzar el último nodo. Para ello, en nuestra implementación hemos utilizado el algoritmo voraz presentado en [37]. Sin embargo, en el caso de un grafo fuertemente conexo, donde cada nodo está conectado a todos los demás, el algoritmo A* está considerando demasiadas restricciones que no son necesarias.

Presentemos teóricamente el algoritmo voraz. Cuando introducimos el algoritmo A*, vimos que constaba de 2 funciones, g y h que representaban el coste exacto del camino

recorrido y la estimación heurística del camino por recorrer. El algoritmo voraz sólo tiene en cuenta la función h, es decir, el coste estimado del camino más corto desde el nodo en el que nos encontramos hasta el nodo final [3].

La idea al utilizar este algoritmo es obtener una lista ordenada de los vértices que mayor relación guardan con los gustos del usuario. Después, basta con ordenar estos nodos para obtener un camino. Como el algoritmo A* proporcionaba caminos que partían del vértice inicial del grafo, debemos forzar que este vértice aparezca en la solución voraz, con lo que lo quitaremos de la entrada del algoritmo, añadiéndolo al ordenar la salida obtenida. El coste del algoritmo voraz es lineal respecto al tamaño de la entrada n, lo que lo hace mucho más rápido que el A*.

Para nuestra implementación deberemos definir cuándo es factible la solución, y cuál es el criterio del algoritmo voraz. La solución (es decir, el conjunto de nodos) será factible siempre y cuando la suma de sus tiempos no supere el límite t_q establecido en la formalización del problema. Para calcular los tiempos entre aristas, dado que se trata de un conjunto de nodos y no un camino, deberemos definir una función:

$$v_i.cost() = \min_{v_j \in G}\{t_{<v_j,v_i>}\}$$

donde el vértice inicial cumple $v_s.cost() = 0$. Por otra parte, el criterio según el cual seleccionará el algoritmo voraz deberá tener en cuenta las características de los nodos, los gustos del usuario y el tiempo empleado en visitar el nodo. Por tanto, se ha optado por el cociente $\frac{v_i.cov(K)}{v_i.cost()+t_{v_i}}$. La implementación final del algoritmo voraz se puede ver en el Algoritmo 3.

Algoritmo 3: Algoritmo Voraz

Function *ItemList CBRRouteFinding(Graph G=(V,E), double* t_q*)*
- ItemList U = V;
- Route R = {};
- double max = 0;
- int maxindex = -1;
- **while** $(|U| > 0)$ **do**
  - $max = \infty$;
  - **for** $(v_i \in U)$ **do**
    - **if** $(\frac{v_i.cov(K)}{v_i.cost()+t_{v_i}} > max)$ **then**
      - maxindex = i;
      - max = $\frac{v_i.cov(K)}{v_i.cost()+t_{v_i}}$;
    - **end**
  - **end**
  - **if** $(t_R + v_{max}.cost() + t_{v_{max}}) < t_q)$ **then**
    - $R = R \cup v_{max}$;
  - **end**
  - $U = U \setminus v_{max}$;
- **end**

3.2.3. Algoritmo CBR

Mientras que los dos algoritmos anteriores eran búsquedas heurísticas, ahora se pretende plantear un algoritmo basado en conocimiento experto, Para ello utilizaremos el Razonamiento Basado en casos (CBR, de sus siglas en inglés) visto en la Sección 2.1.4.

Tal y como se explico antes, un grupo (la consulta en el algoritmo CBR) estará caracterizado por una tupla $(size_q, age_q, pref_q, t_q)$, donde $size$ es el número de personas en el grupo, age es la edad media de sus miembros, $pref_k = \{\lambda_{k_1} \dots \lambda_{k_q}\}$ con $\lambda_{k_i} \in [0,1]$ es un conjuntos de pesos que indican las preferencias del grupo para cada categoría, y t_q es la restricción en tiempo, la máxima duración de la visita. Un individuo estará caracterizado por la tupla $(age_q, pref_q, t_q)$. Un caso $c = (size_c, age_c, pref_c, t_c, R_{sol_c})$ constará de la tupla de datos del individuo/grupo y el recorrido R_{sol_c} propuesto por un experto. Estos casos son obtenidos mediante Ingeniería del Conocimiento.

Para realizar una recomendación, compararemos la consulta introducida con los distintos casos CBR realizados por el experto en el dominio, y recuperaremos el caso que más se parezca a nuestros datos. Obtendremos este caso comparando cada uno de los campos de nuestra entrada con los casos de los que disponemos, como se detalla en el Algoritmo 4.

La función de similitud tiene en cuenta cada característica de la consulta, es decir, el tiempo disponible para la visita, el tamaño del grupo, la edad media y las preferencias medias del grupo. La distancia entre un caso c y la consulta q se computa de la siguiente forma:

$$\begin{aligned} distancia(c,q) &= |t_c - t_q| \cdot constTime \\ &+ |size_c - size_q| \cdot constSize \\ &+ |age_c - age_q| \cdot constAge \\ &+ \left(\sum_{k_i \in K} |pref_c(k_i) - pref_q(k_i)|\right) \cdot constLabel \end{aligned}$$

Los valores *constTime*, *constSize* y *constAge* pueden cambiar dependiendo del dominio, aunque se recomienda que los dos primeros sean mayores que el último. La función de estas constantes es ponderar los distintos atributos que forman la consulta.

La distancia es calculada en el intervalo $[0, \infty]$, pero necesitamos una medida de similitud en el intervalo $[0, 1]$, con valor 1 para casos idénticos, y 0 cuando la distancia tiende a infinito. Por esta razón, nuestra función de similitud se calcula de la siguiente manera:

$$Sim(c,q) = \frac{1}{e^{distancia(c,q)}} \tag{3.2}$$

Revisión de la solución

Para revisar la solución propuesta, correspondiente al tercer paso del ciclo CBR, se realizan dos estrategias de adaptación diferentes sobre la solución obtenida R_{sol_c}:

Algoritmo 4: Algoritmo CBR

```
Function ItemList CBRRouteFinding(List<CBRVisit>cbrlist, CBRVisit q)
    double mindistance = ∞ ;
    CBRVisit minvisit = null ;
    for todas las visitas c en cbrlist do
        double distance = 0 ;
        distance += |t_c − t_q| · constTime ;
        distance += |size_c − size_q| · constSize ;
        distance += |age_c − age_q| · constAge ;
        for todas las etiquetas i ∈ K do
            distance += |pref_c(k_i) − pref_q(k_i)| ;
        end
        if (distance <mindistance) then
            minvisit = cbrv ;
            mindistance = distance ;
        end
    end
    R_solc = route_minvisit;
    while (t_q > 1,1 · t_R_solc) do
        Eliminar el ítem v_i con peor cobertura ;
    end
    while (t_q > t_R_solc) do
        for todos los ítems v_i ∈ R_solc do
            Reducir el tiempo en v_i ;
        end
    end
    return R_solc;
```

- Reducción del tiempo pasado en los puntos de interés. Si el tiempo requerido para completar la visita recomendada es ligeramente mayor que el tiempo disponible para la visita, la visita recomendada se adapta reduciendo ligeramente los tiempos en los puntos de interés de la ruta.

- Eliminación de puntos de interés de la solución. Si el tiempo requerido para completar la visita recomendada es notablemente mayor que el tiempo disponible para la vista, la visita recomendada se adapta eliminando los puntos con peor valor de cobertura. Para conseguir esto, los vértices con menor valor $v_i.cov(K)$ son borrados de la solución.

3.3. Construcción del modelo para su adaptación a un caso concreto

A continuación se detallan los pasos requeridos para poder aplicar el marco teórico visto en este capítulo a problemas concretos.

1. **Representación del mapa como un grafo.** En primer lugar debemos identificar los objetos del dominio que formarán los puntos de interés. A continuación, deberemos adaptar el mapa de nuestro dominio a un grafo $G = (V, E)$ que sea dirigido y fuertemente conexo, y donde V sea el conjunto de puntos de interés. Las aristas del grafo $< v_i, v_j >$, con $v_i, v_j \in V$ deben tener asociado un valor $t_{<v_i,v_j>}$ que represente el tiempo necesario para desplazarse desde v_i hasta v_j, y los vértices tendrán un atributo t_{v_i} que indique el tiempo recomendado para visitar el punto v_i.

2. **Definición del conjunto de etiquetas.** Con ayuda de un experto del dominio se deben identificar las etiquetas $K = \{k_1, \ldots, k_n\}$ que caractericen a los puntos de interés elegidos. Posteriormente, a cada punto se le asignará un conjunto de valores $\{\lambda_{k_1}, \ldots, \lambda_{k_n}\}$, donde $\lambda_{k_j} \in [0, 1]$ representa el grado en que el punto v_i cubre la etiqueta k_j.

3. **Elección del algoritmo de recomendación.** Hasta lograr una base de casos suficientemente grande, se recomienda utilizar los algoritmos A* y voraz, siendo preferible A* para dominios pequeños de hasta 15 vértices y el voraz para dominios más grandes. Para lograr la base de casos, el mejor procedimiento es utilizar visitas reales, pero se puede realizar una entrevista con el experto en el dominio para que proporcione recomendaciones teóricas para distintos usuarios o grupos.

4. **Recomendación de rutas.** Para recomendar una ruta a un usuario o grupo, éste debe indicar tanto el tiempo del que dispone para realizar la visita como sus preferencias, que expresará asignando valores $\lambda_{k_j} \in [0, 1]$ a cada una de las etiquetas del dominio. Además, si se está utilizando el algoritmo CBR, deberá introducirse la edad en el caso de las visitas individuales, y los valores de edad media y tamaño del grupo para las visitas grupales.

Opcionalmente, podrán incluirse técnicas basadas en Realidad Aumentada que mejoren la experiencia del usuario durante el recorrido de la ruta, introducidos en la Sección 2.2 y cuya aplicación podrá verse en los casos de estudio del próximo capítulo.

3.4. Conclusiones

En este capítulo se ha propuesto un marco teórico para la personalización de rutas. Se han presentado tres algoritmos, con los que se pueden cubrir todo tipo de dominios, desde los que estén formados por un conjunto pequeño de puntos de interes hasta los que se

compongan de muchos puntos. En el Capítulo 4 veremos dos casos de estudio que llevarán a la práctica el marco teórico propuesto.

4

Casos de estudio

En este capítulo presentaremos dos casos de estudio donde aplicaremos el marco general y los algoritmos de recomendación vistos en el Capítulo 2.1.1. En el primer caso de estudio, proporcionaremos rutas personalizadas en el Museo García Santesmases, donde tendremos más de 50 objetos para visitar. Además, en esta sección se incluirá un estudio comparativo experimental de los algoritmos CBR y voraz. En el segundo caso de estudio recomendaremos rutas turísticas en el centro de Madrid. En este caso sólo tendremos 14 puntos de interés entre los que elegir, lo que nos permitirá aplicar el Algoritmo A*.

Para cada uno de los casos de estudio propuestos hemos creado un prototipo de aplicación móvil. En el caso del museo, no podremos utilizar posicionamiento en interiores para localizar los expositores, puesto que el error de precisión es mayor que las distancias entre los objetos. En lugar de esto, colocaremos marcadores en las vitrinas con los que, mediante Realidad Aumentada, los usuarios podrán acceder a contenidos relacionados con el objeto expuesto. Para el recomendador de turismo en Madrid, al estar localizado en exteriores, utilizaremos técnicas de geolocalización por GPS para proporcionar experiencias de Realidad Aumentada a los usuarios.

4.1. Caso de estudio: Museo de Informática García Santesmases

Ubicado en las plantas tercera y cuarta de la facultad de Informática, el Museo García Santesmases contiene más de 50 piezas relacionadas con la historia de la informática, y permite realizar una visita desde los orígenes de la misma en España, donde llegó de la mano del propio García Santesmases [39], hasta los últimos dispositivos utilizados en la

actualidad.

Estas piezas, aunque tienen en común su relación con el mundo de la informática, varían desde los primeros ordenadores digitales hasta los últimos modelos de servidores, pasando por procesadores, videoconsolas y diversos periféricos. El objetivo es personalizar las rutas que recorran los visitantes según las distintas inquietudes que tengan y el tiempo del que dispongan para la visita.

Durante el proceso de adquisición de conocimiento del dominio hemos observado que se pueden realizar rutas temáticas para diversos grupos, como pueden ser estudiantes de informática, interesados por categorías como *hardware*, *servidores* o *redes*, personas mayores, que prefieren contenidos relacionados con *historia*, *España* o *curiosidad*, cinéfilos, interesados por *cine* y *curiosidad*, niños que muestran preferencia por *videojuegos*, etc.

Además, queremos mejorar la experiencia de los visitantes utilizando Realidad Aumentada, de manera que se les presente la información de una forma novedosa que les haga interesarse por los contenidos del museo. Dado que el museo se encuentra en un edificio, deberemos utilizar tecnologías de Realidad Aumentada en interiores, como pueden ser Sistemas de Posicionamiento en Interiores o marcadores.

4.1.1. Resolución del problema

Para poder aplicar las técnicas vistas en el Capítulo 2.1.1 debemos formalizar el problema según se explicó en la Sección 3.1. En primer lugar definamos el grafo que recorreremos. Los nodos que lo forman son las distintas vitrinas del museo, que están compuestas por una o más piezas. La lista detallada de las vitrinas se puede encontrar en el Apéndice A. Dado que el museo consta de dos plantas, uniremos los subgrafos que formarían cada una de ellas en un sólo grafo, cuyos vértices se muestran en la Figura 4.1.

Además, cada nodo tendrá los siguientes atributos.

- **Identificador:** un número del 1 al 52 que indica de qué vitrina se trata.
- **Descripción:** breve descripción de la(s) pieza(s) que componen la vitrina.
- **Tiempo:** minutos que se recomienda tardar en examinar la vitrina.
- **Etiquetas:** conjunto de pares <Etiqueta, Valor> que categorizan la vitrina, asignando un valor entre 0 y 1 a las distintas etiquetas. En caso de que la vitrina cuente con más de una pieza, se ponderarán los valores de las etiquetas de las piezas que la compongan.

Las etiquetas del museo se han identificado en un proceso de ingeniería del conocimiento, y son: *almacenamiento*, *arte*, *ciencias*, *cine*, *curiosidad*, *docencia*, *España*, *Facultad de Informática*, *hardware*, *Historia*, *PCs*, *periféricos*, *redes*, *servidores* y *videojuegos*.

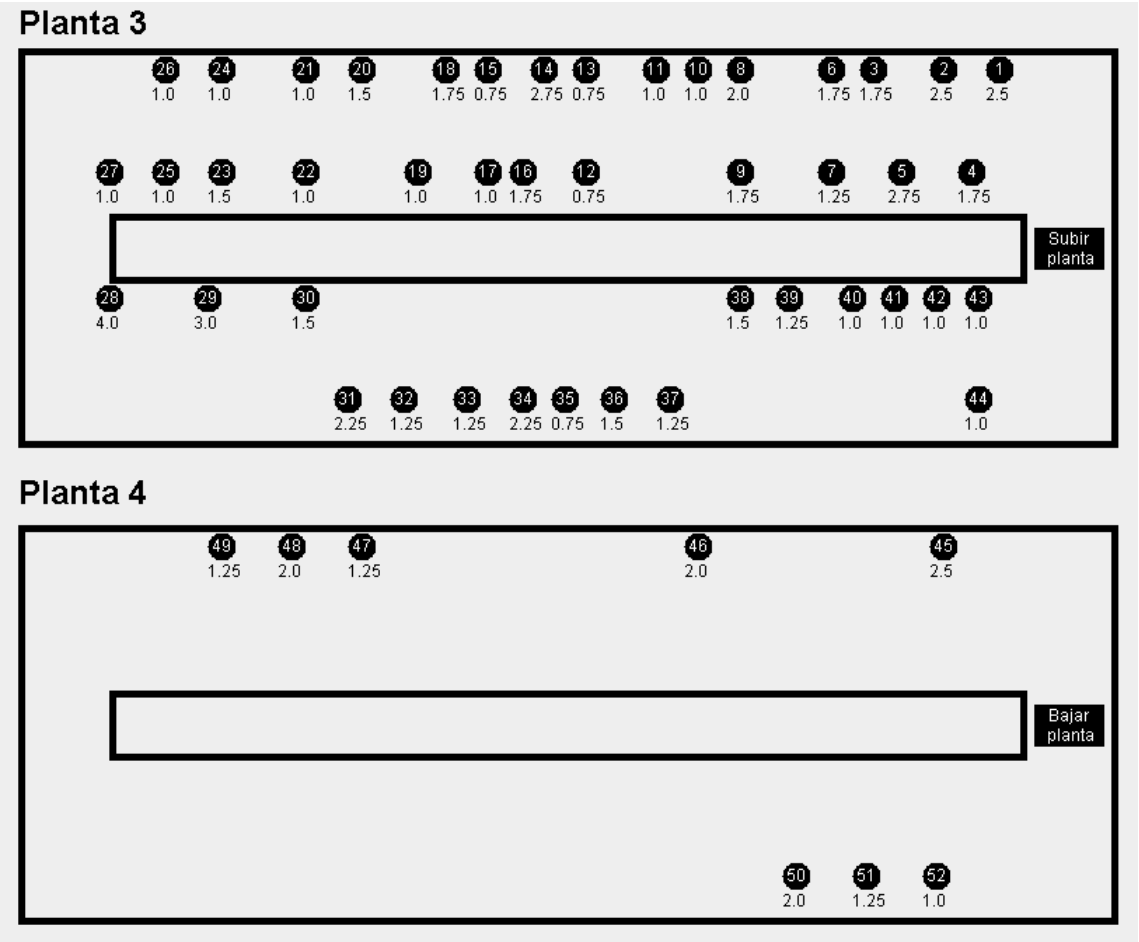

Figura 4.1: Nodos del Museo García Santesmases

Al ser un museo, donde nos podemos desplazar libremente, todos los nodos están conectados entre sí. Las aristas del grafo serán las distancias en tiempo entre los puntos, que en este caso concreto son casi nulas debido a la proximidad entre las distintas vitrinas. Aún así deberemos tenerlas en cuenta para el correcto funcionamiento del algoritmo. Para automatizar la generación de estas aristas, calculado la distancia entre dos nodos en función de la distancia geográfica en el grafo anterior, con alguna peculiaridad, que a continuación se explica.

El museo está dividido en 4 zonas: cada ala de cada planta es una zona diferente. Para calcular la distancia entre dos puntos, comprobaremos a qué zona pertenecen. Si ambos están en la misma zona, la distancia entre ellos será la distancia geográfica en el grafo multiplicada por una constante empírica que representa la velocidad, de tal forma que resulte la distancia en tiempo entre los dos puntos. Si los dos puntos pertenecen a distintas zonas, calcularemos la distancia desde el primer punto al punto de cambio de zona (si las dos zonas están en la misma planta será el lateral de la planta, en caso contrario las escaleras) y desde ahí al segundo punto.

Ahora que ya disponemos de los nodos del grafo y de las aristas que los unen, podemos

ejecutar los algoritmos. Para ello necesitaremos describir la visita. No podremos utilizar el algoritmo A*, pues el grafo consta de 52 nodos y no podría calcular la ruta personalizada en un tiempo razonable. El algoritmo voraz calcula una ruta que se adecúa a las preferencias introducidas por el usuario o el grupo, así como al tiempo del que dispone para la visita al museo. El algoritmo CBR además de las preferencias y el tiempo, recibirá como entrada la edad del usuario, o el tamaño del grupo y su edad media, en el caso de grupos. Una vez tenga todos los datos, el algoritmo devolverá la lista ordenada de nodos que deberá visitar, así como el tiempo que deberá gastar en cada uno de ellos.

Los casos del algoritmo CBR han sido adquiridos durante un proceso de observación de visitas reales al MIGS durante un mes, a los que se han sumado otros proporcionados por el experto del dominio (José Luis Vázquez-Poletti). El conjunto de casos consta de 28 casos prototípicos que cubren diferentes situaciones de visitas grupales, como grupos de varios tamaños de distintos alumnos de grados universitarios, másters o doctorados; visitas rápidas para diferentes grupos de las vitrinas más importantes del museo, visitas para niños, adultos y personas mayores, visitas temáticas con diferentes intereses (general, cine, videojuegos, historia). La Figura 4.2 visualiza el caso base como un grafo. Los nodos representan los casos y las aristas la similitud entre casos. El grosor de las aristas se ha calculado con la función de similitud .

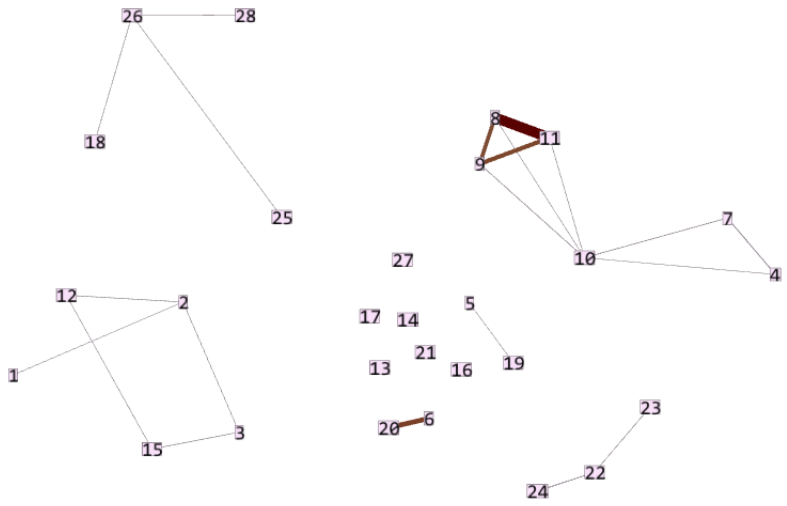

Figura 4.2: Base de casos representada como un grafo

Los valores correspondientes a la función de distancia definida en la Sección 3.2.3 son $constTime = 0.4$, $constSize = 0.2$, $constAge = 0.1$ y $constLabel$=0.5. Los pesos reflejan la importancia de las diferentes variables. La relevancia de la edad es menor puesto que se ha observado una dependencia entre edad y preferencias, es decir, algunas de las preferencias grupales pueden depender de la edad y son cubiertas por las distintas etiquetas en la consulta.

4.1.2. Recorridos de prueba

Para comparar los algoritmos voraz y CBR, estudiaremos una serie de recorridos personalizados para varios grupos de usuarios con gustos diferentes, correspondientes a visitas indicadas por el experto en el dominio. Los grupos seleccionados serán alumnos de informática, interesados en categorías como *FDI*, *redes* o *hardware*, alumnos de ciencias, con preferencia por vitrinas relacionadas con *ciencias* o *curiosidad* y pesonas mayores, que desean visitar objetos relacionados con las etiquetas *historia*, *España*, *curiosidad*.

Alumnos de Informática

Contemplemos primero la recomendación para un grupo de alumnos de la Facultad de Informática que disponen de una hora para visitar el museo. La tupla $(size_q, age_q, pref_q, t_q)$ que caracteriza las visitas en el algoritmo CBR será $(20, 20, pref_q, 60)$, donde $pref_q$ se corresponde con las preferencias que se exponen a continuación.

- Almacenamiento: 0,65
- Arte: 0,4
- Ciencias: 0,6
- Cine: 0,5
- Curiosidad: 0,7
- Docencia: 0,4
- España: 0,6
- FDI: 0,9
- Hardware: 0,65
- Historia: 0,6
- PCs: 0,9
- Periféricos: 0,6
- Redes: 0,6
- Servidores: 0,6
- Videojuegos: 0,9

Las rutas recomendadas para ellos pueden verse en la Figura 4.3. En ella aparecen en color verde los puntos de interés seleccionados por el algoritmo voraz, en azul los escogidos por el CBR, y en rojo los que han sido recomendados por ambos algoritmos. Vemos que tanto la ruta recomendada por el algoritmo voraz como la elegida por el CBR constan de 36 nodos, de los cuales 29 son comunes. Entre ellos se encuentran los elementos más representativos del museo, como el Supercomputador Cerebro, el primer minicomputador de España, el Friden EC-132 con el tubo de rayos catódicos, el simulador de centrales térmicas, los ordenadores Amstrad o las obleas de silicio.

Las vitrinas elegidas en cada ruta son:

- Voraz: 1, 2, 3, 4, 8, 10, 11, 13, 14, 15, 19, 20, 22, 23, 24, 26, 28, 29, 30, 31, 32, 33, 35, 36, 37, 38, 40, 43, 45, 46, 47, 48, 49, 51, 52.
- CBR: 1, 2, 3, 4, 6, 8, 10, 11, 13, 14, 15, 20, 23, 24, 26, 29, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 48, 49, 50, 51, 52.

- Comunes: 1, 2, 3, 4, 8, 10, 11, 13, 14, 15, 20, 23, 24, 26, 29, 31, 33, 35, 36, 37, 38, 39, 40, 43, 45, 48, 49, 51, 52.
- Sólo voraz: 19, 22, 28, 30, 39, 46, 47.
- Sólo CBR: 6, 34, 39, 41, 42, 44, 50.

Con respecto a las pequeñas diferencias entre ambas rutas, el algoritmo voraz escoge los nodos 28 y 30, correspondientes a los ordenadores Amstrad y con una alta valoración en la etiqueta *videojuegos*. Por contra, el algoritmo CBR rescata la visita de alumnos de informática en clase, que cuenta con unas preferencias muy similares a nuestra visita, pero con menor valor en el campo de videojuegos. Por tanto, en lugar de elegir esos dos nodos, se prefiere profundizar en las vitrinas que abarcan desde el punto 38 al 43. En este caso, además de tratar contenidos más relacionados con el temario de las asignaturas de la carrera, se puede desarrollar una explicación conjunta de todas las vitrinas, conectando unas con otras.

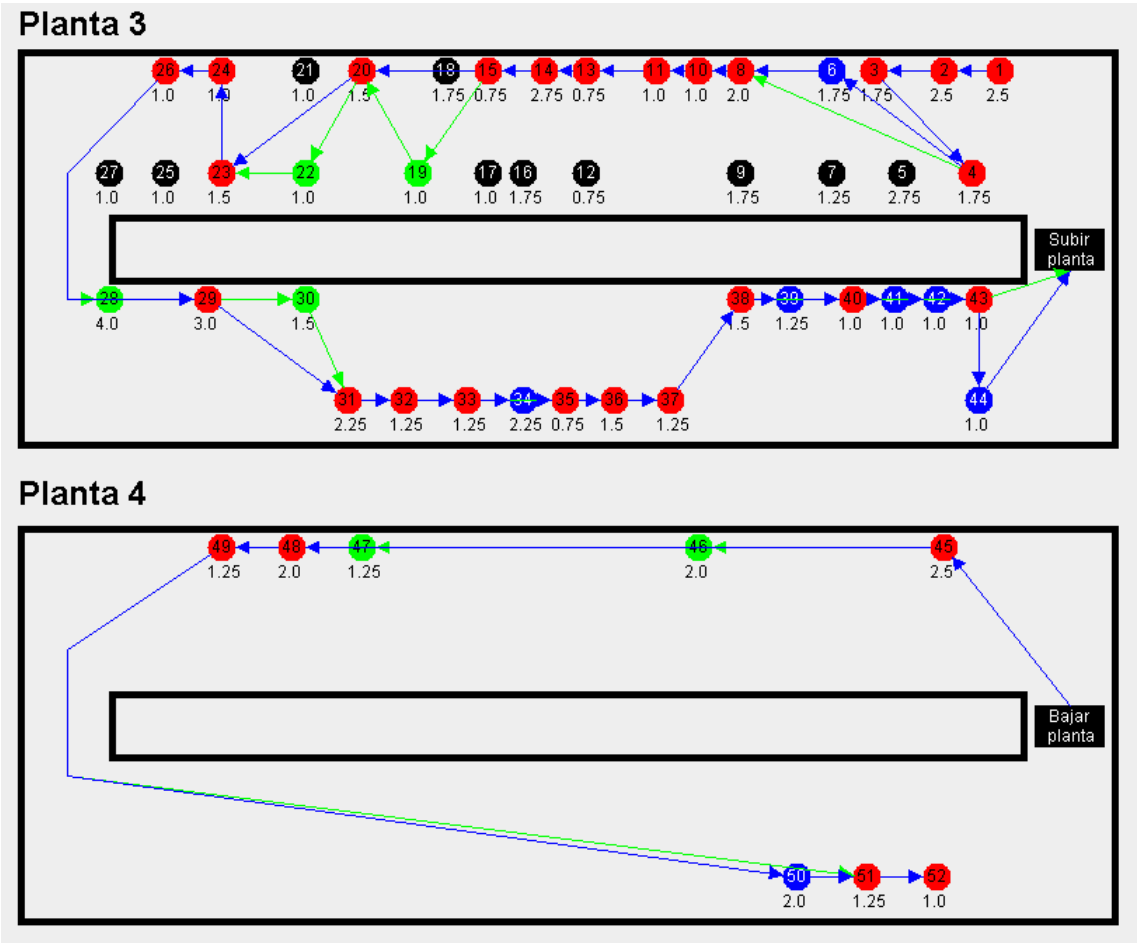

Figura 4.3: Ruta recomendada para alumnos de Informática

Alumnos de carreras de ciencias

Continuemos ahora con el caso de un grupo de alumnos de un máster en ciencias que dispongan de una hora para realizar una visita al museo. En este caso, la tupla que caracterizará las visita en el algoritmo CBR será $(30, 25, pref_q, 60)$, donde las preferencias $pref_q$ corresponden los gustos que se presentan a continuación.

- Almacenamiento: 0,3
- Arte: 0,5
- Ciencias: 0,9
- Cine: 0,4
- Curiosidad: 0,9
- Docencia: 0,5
- España: 0,7
- FDI: 0,4
- Hardware: 0,2
- Historia: 0,7
- PCs: 0,5
- Periféricos: 0,2
- Redes: 0,3
- Servidores: 0,2
- Videojuegos: 0,6

Las rutas recomendadas para ellos pueden verse en la Figura 4.4. Las vitrinas elegidas en cada ruta son:

- Voraz: 1, 2, 3, 4, 7, 8, 10, 11, 12, 14, 15, 16, 17, 19, 20, 22, 23, 24, 26, 28, 29, 30, 31, 33, 34, 35, 36, 37, 38, 45, 47, 48, 49, 51.
- CBR: 1, 2, 3, 4, 6, 9, 11, 12, 14, 15, 18, 19, 23, 28, 29, 30, 31, 34, 35, 36, 38, 40, 43, 45, 46, 48, 50, 51, 52.
- Comunes: 1, 2, 3, 4, 11, 12, 14, 15, 19, 23, 28, 29, 30, 31, 34, 35, 36, 38, 45, 48, 51.
- Sólo voraz: 7, 8, 10, 16, 17, 20, 22, 24, 26, 33, 37, 47, 49.
- Sólo CBR: 6, 9, 18, 40, 43, 46, 50, 52.

De nuevo vemos en color verde los puntos de interés elegidos por el algoritmo voraz, en azul los que ha recomendado el algoritmo CBR y en rojo los comunes a ambos algoritmos. Al igual que en la visita anterior, elementos representativos del museo como el primer minicomputador de España, los ordenadores Amstrad o las obleas de silicio han sido escogidos por ambos algoritmos.

En este caso la visita recuperada era de alumnos del máster en Bioinformática, que aunque comparten unas preferencias similares con los alumnos de ciencias muestran más interés por la informática, por lo que la visita se centra más en la parte de *PCs* y *servidores* (por ejemplo visitando los discos duros y el CRAY Y-MP EL). Por su parte, el algoritmo voraz se centra más en los objetos de la primera parte del museo, de carácter más general, como pueden ser los teléfonos móviles o el perro robótico Sony AIBO ERS-7.

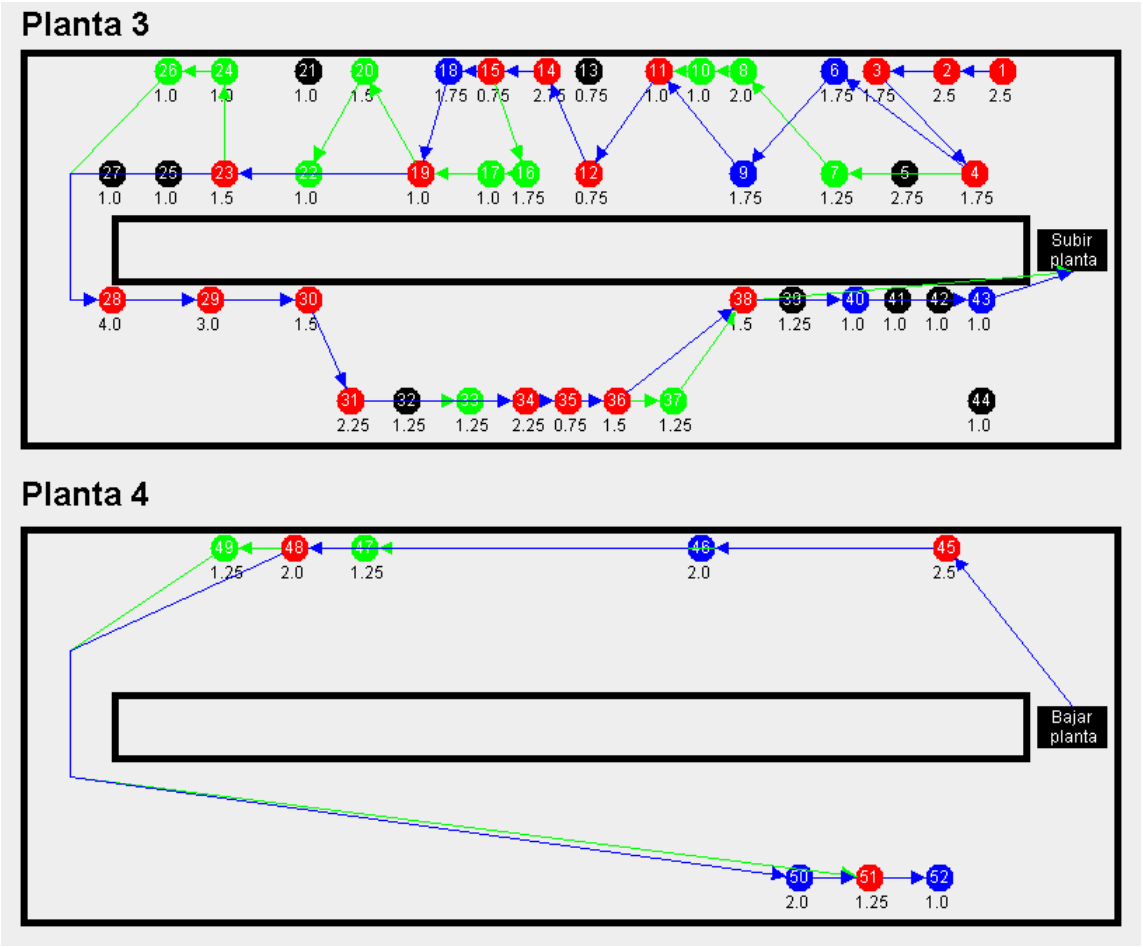

Figura 4.4: Ruta recomendada para alumnos de ciencias

Personas mayores

Por último, consideremos un grupo de personas mayores que se disponen a realizar una visita al museo, para la que disponen de 40 minutos. La tupla correspondiente a esta visita será $(30, 65, pref_q, 60)$, donde las preferencias $pref_q$ han sido modelizadas como se expone a continuación.

- Almacenamiento: 0,2
- Arte: 0,6
- Ciencias: 0,5
- Cine: 0,6
- Curiosidad: 0,9
- Docencia: 0,6
- España: 0,8
- FDI: 0,3
- Hardware: 0,3
- Historia: 0,9
- PCs: 0,4
- Periféricos: 0,3
- Redes: 0,2
- Servidores: 0,2

- Videojuegos: 0,1

La ruta recomendada para ellos puede verse en la Figura 4.5. Las vitrinas elegidas en cada ruta son:

- Voraz: 1, 2, 3, 4, 7, 8, 10, 11, 14, 15, 19, 20, 22, 23, 28, 30, 31, 36, 38, 40, 48.
- CBR: 1, 2, 4, 6, 8, 10, 14, 15, 20, 22, 28, 29, 30, 31, 34, 36, 37, 45, 46, 48, 51.
- Comunes: 1, 2, 4, 8, 10, 14, 15, 20, 22, 28, 30, 31, 36, 48.
- Sólo voraz: 3, 7, 11, 19, 23, 38, 40.
- Sólo CBR: 6, 29, 34, 37, 45, 46, 51.

En primer lugar observamos que tanto el agoritmo voraz (verde) como el CBR (azul) recomiendan una ruta de 21 nodos, de los cuales 14 son comunes a ambos algoritmos (un 66 % del total). Entre los nodos que no lo son, se puede ver cómo el algoritmo voraz recomienda ítems de categorías variadas, pero que interesan al grupo, mientras que el CBR realiza alguna recomendación que no se ajusta a los gustos de los usuarios (por ejemplo tanto el IBM 7090 como las consolas Atari y Nintendo están etiquetados con la categoría *videojuegos*, que no interesa a los visitantes. Esto se debe a que el caso que ha recuperado el algoritmo (una visita de padres de alumnos de la facultad) presenta ligeras diferencias en los valores de las preferencias de los usuarios.

Conclusiones

Con los tres casos tratados anteriormente podemos concluir que ambos algoritmos proporcionan recomendaciones acertadas de rutas en el museo. La diferencia entre las rutas propuestas entre ambos es baja siempre que se cuente con recomendaciones para grupos similares en la base de casos. A continuación presentaremos una comparativa entre ambos algoritmos donde se han realizado una gran cantidad de consultas a ambos recomendadores aleatorizando las variables que conforman dichas consultas.

4.1.3. Comparación del algoritmo voraz con el algoritmo CBR

El propósito de la evaluación es comparar los valores de la función de cobertura $kc\_sum(R)$ descrita anteriormente, de acuerdo a los atributos de tamaño, tiempo disponible para la visita, edad media y preferencias del grupo.

El experimento consistirá en una serie de tests que evalúen el impacto de cada atributo en la consulta de forma independiente, y las diferencias entre los resultados obtenidos por los algoritmos voraz y CBR. Para ello, cada atributo evaluado es fijado a diferentes valores, y los demás atributos se generan de manera aleatoria. El atributo correspondiente a las

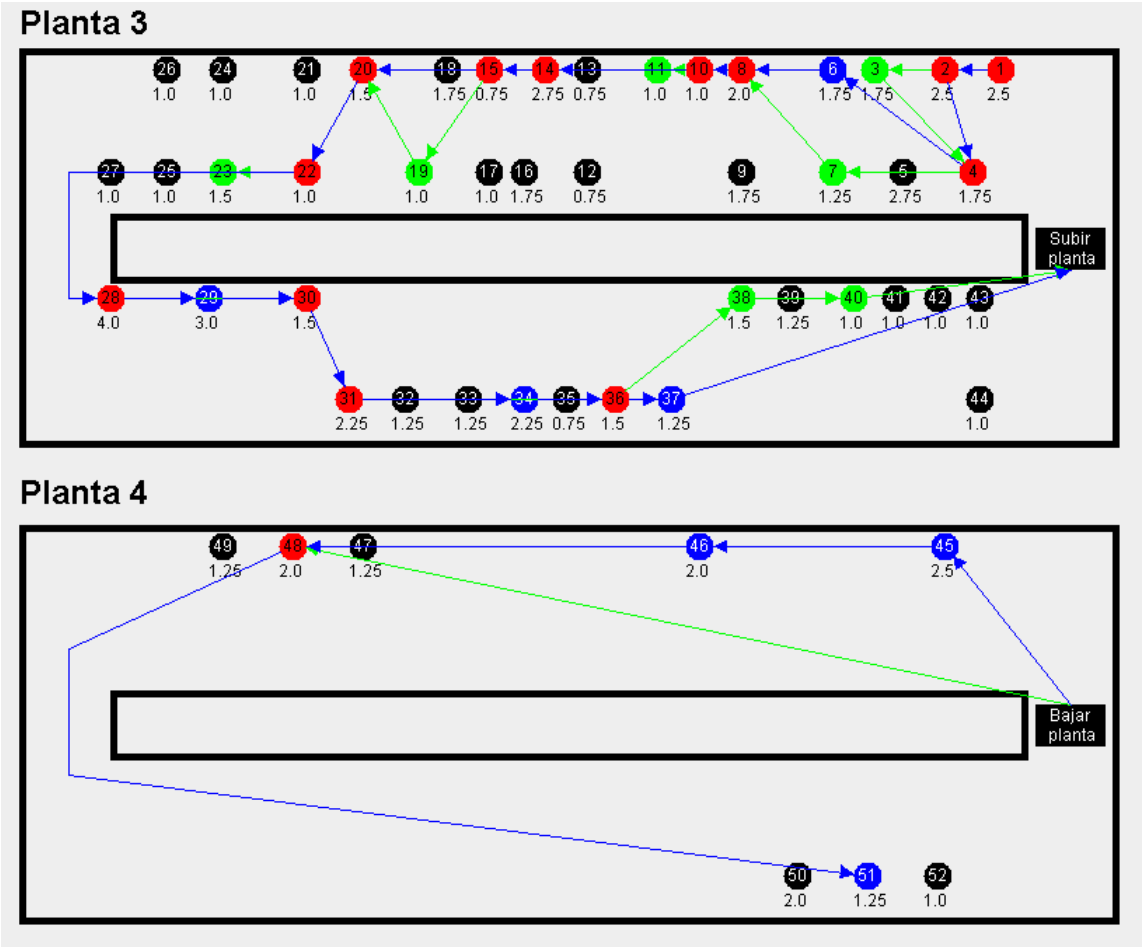

Figura 4.5: Ruta recomendada para mayores

preferencias del grupo se ha fijado a tres conjuntos que representan tres visitas prototípicas (intereses neutros, estudiantes de Ingeniería Informática y personas mayores). En cada test se han generado 400 consultas para cada valor fijado.

Para cada experimento hemos medido el valor de la función de cobertura $kc\_sum(\mathrm{R})$ y hemos normalizado el resultado obtenido con respecto a una consulta equivalente que representa una visita que recorre todos los puntos de interés. Finalmente hemos calculado el valor medio para las 400 consultas, definiendo un porcentaje medio de cobertura.

Resultados Experimentales

Como es esperable, el algoritmo voraz ofrece mejores resultados en términos de cobertura, aunque vemos que el Algoritmo CBR también muestra buenos resultados con un tiempo de respuesta muy eficiente.

La Figura 4.6 muestra algunos de los resultados experimentales obtenidos al fijar la longitud de la visita (t_q) en la consulta. Los resultados muestran, como era de esperar,

que la cobertura de la ruta aumenta al aumentar el tiempo disponible para la visita. 90 minutos dan suficiente tiempo a los visitantes como para visitar todos los puntos de interés del museo, por lo que las preferencias del grupo no afectan a la ruta.

No obstante, podemos observar que la cobertura obtenida por el algoritmo CBR es menor en los grupos grandes incluso en una visita de 90 minutos donde todos los nodos son elegidos en la solución obtenida por el algoritmo voraz. Esto se debe a las restricciones de espacio que han sido capturadas como conocimiento en los casos. Los casos nos permiten observar que los grupos grandes nunca se paran en las vitrinas pequeñas. Eso también explica por qué la cobertura es menor en el caso del CBR aun cuando hay suficiente tiempo como para visitar todos los nodos. El algoritmo voraz no tiene en cuenta el conocimiento experto y por tanto genera rutas que no son adecuadas para grupos grandes.

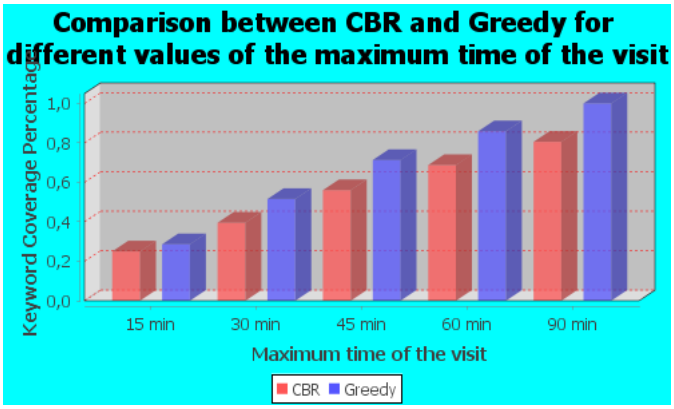

Figura 4.6: Comparación para diferentes valores de tiempo disponible

En la Figura 4.7 observamos los valores de la cobertura para grupos de diferente tamaño ($size_q$). Vemos cómo la cobertura apenas varía con los distintos tamaños de grupo analizados. Como hemos mencionado anteriormente, la cobertura es menor en grupos grandes debido a las restricciones de espacio en el museo, ya que el algoritmo voraz proporciona visitas que nunca se podrían llevar a cabo con grupos grandes.

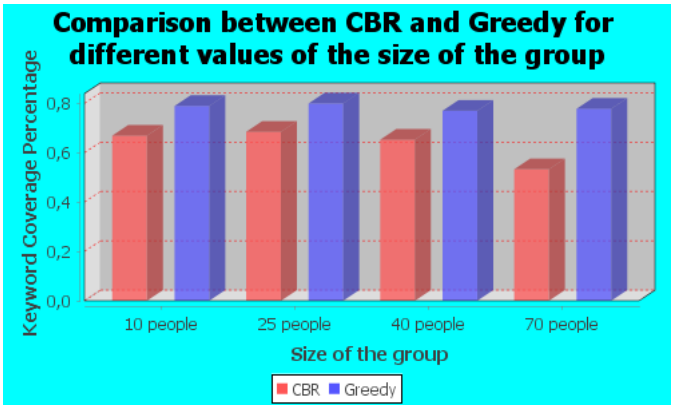

Figura 4.7: Comparación para diferentes valores de tamaño de grupo

Aunque la edad (age_q) es considerada como un atributo que caracteriza la visita del grupo, su importancia es menor que la de otros atributos. Por ello, al haber considerado una baja ponderación en el algoritmo CBR, la influencia en la cobertura es menor (Figura 4.8.

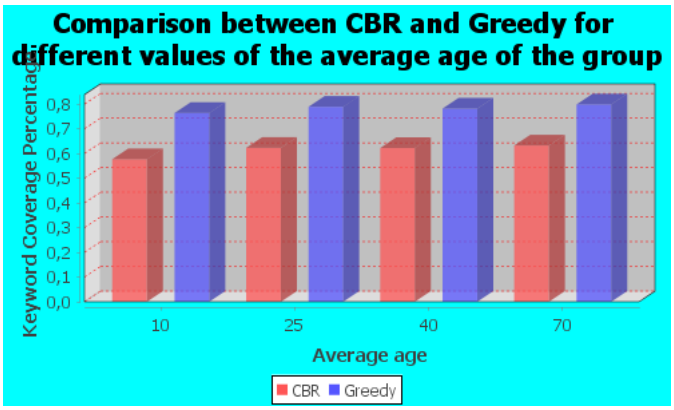

Figura 4.8: Comparación para diferentes valores de edad media del grupo

La Figura 4.9 compara los resultados con la consultas que representan preferencias prototípicas de grupos para visitás temáticas ($pref_q$), El primer grupo (*neutral*) muestra un interés medio en cada etiqueta, el segundo grupo (*mayores*) prefiere los objetos relacionados con las categorías *historia*, *españa* y *curiosidad*, mostrando poco interés por categorías como *hardware*, *servidores*, *redes* o *videojuegos*. Finalmente los *estudiantes de informática* están más interesados en las categorías específicas como *servidores* y *redes*, mostrando un alto interés general en los contenidos exhibidos en el museo.

Como el grupo de *mayores* apenas muestra interés por la mitad de las etiquetas del museo, es razonable que la cobertura de los recorridos recomendados para ellos sea menor que la de los recorridos proporcionados al grupo *neutral*. Por otra parte, dado que los *estudiantes de informática* forman un grupo muy interesado en casi todas las categorías, la cobertura asociada a las rutas que les sean recomendadas será notablemente mayor que las del grupo *neutral*. Sin embargo, ya que consideramos el porcentaje de cobertura sobre el máximo posible de ese grupo y no el valor total, vemos que los tres grupos presentan valores similares.

Al incluir más etiquetas en la consulta, la cobertura de las recomendaciones del algoritmo voraz aumenta, puesto que está más informado. Como para el algoritmo CBR las preferencias de los usuarios son sólo parte de su entrada, la diferencia apenas es apreciable (Figura 4.10). Para este experimento hemos delimitado el tiempo disponible a visitas entre 15 y 30 minutos, ya que para valores mayores las diferencias apenas serían apreciadas, pues ambos algoritmos seleccionarían casi todos los puntos de interés del museo.

Los resultados de la cobertura en la aproximación CBR dependen fuertemente de la calidad del caso base recuperado. En esta evaluación preliminar, los resultados del algoritmo CBR son prometedores con una base de casos relativamente pequeña (28 visitas) y un tiempo de respuesta inmediato. No alcanzar resultados óptimos en la cobertura no resulta un problema en visitas de grupos cuyas preferencias son obtenidas como una

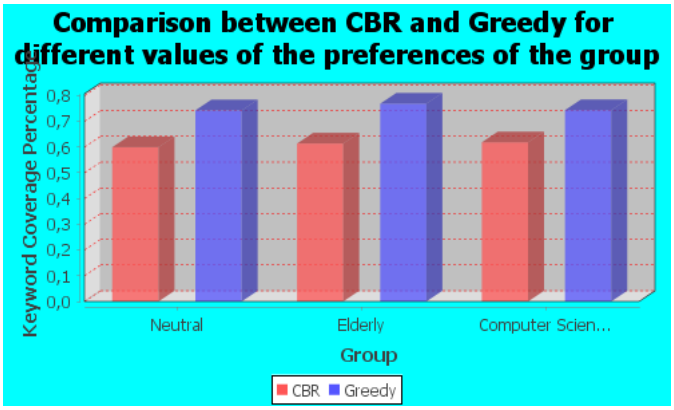

Figura 4.9: Comparación para diferentes valores de las preferencias del grupo

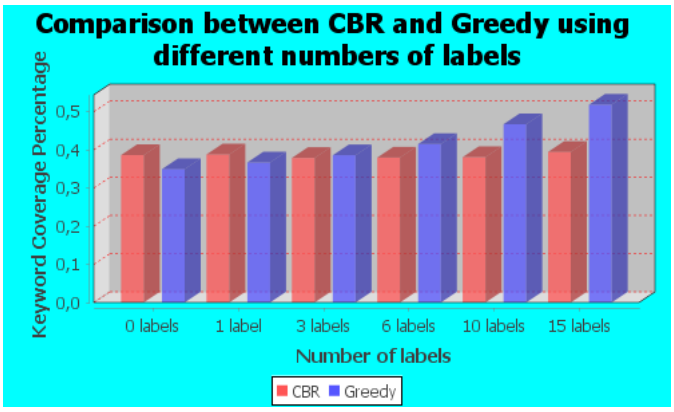

Figura 4.10: Comparación para diferente número de etiquetas consideradas

media entre las de los miembros del grupo. La búsqueda heurística optimiza la cobertura, pero depende de un proceso detallado de definición de los valores de las preferencias de los usuarios. La aproximación CBR es aplicable para consultas cuyas preferencias son aproximaciones con valores prototípicos o incluso no están definidos.

Como hemos comentado anteriormente, algunas rutas calculadas por el algoritmo de búsqueda heurística presentan problemas al aplicarlas a casos reales (como ocurre con las restricciones de espacio en las pruebas anteriores). Es posible que otras medidas de evaluación, como la satisfacción del usuario, puedan beneficiar a la aproximación CBR ya que capturan situaciones reales, y otros aspectos sutiles, como dependencias entre nodos, que ayudan a dar coherencia a la guía narrativa que se sigue al describir los objetos.

4.1.4. Prototipo de aplicación con Realidad Aumentada

Para poder aplicar directamente las recomendaciones en el museo a visitas reales, se ha realizado una aplicación móvil en Android que permite al usuario introducir sus preferencias y el tiempo disponible para la visita, recibiendo la ruta personalizada.

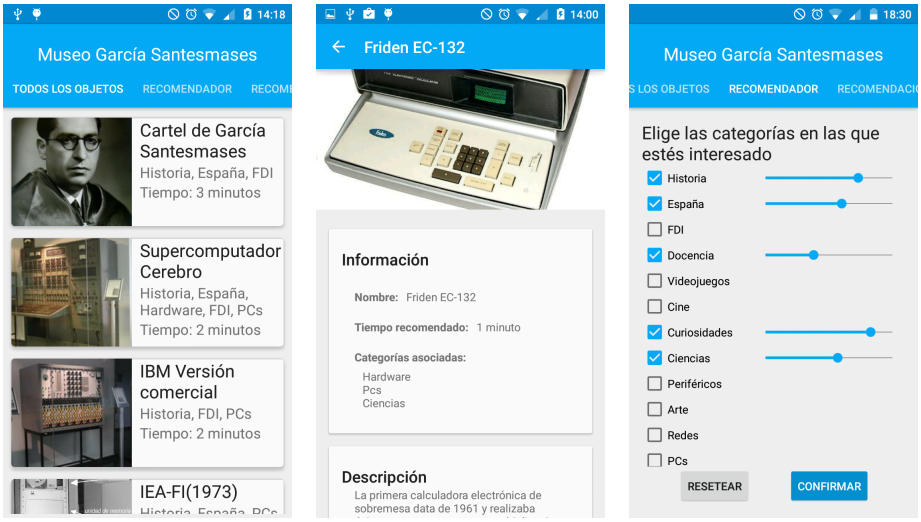

Figura 4.11: App móvil MIGS

En la Figura 4.11 (izquierda) podemos ver la lista de todos los objetos exhibidos en el museo. En cada objeto podemos ver su imagen, su nombre, las etiquetas asociadas a la pieza y el tiempo recomendado para visitarla. En la Figura 4.11(centro) observamos la vista en detalle de un objeto, a la que accedemos desde la lista de objetos anterior. En esta vista podemos ver la imagen, etiquetas y tiempo, así como una descripción de la pieza, y enlaces a páginas web que proporcionan más información sobre el objeto en cuestión.

Por último, en la Figura 4.11 (derecha) vemos la interfaz del recomendador, donde podemos introducir las categorías en las que estamos interesados seleccionándolas en las checkboxes de la izquierda, y ajustando el grado de interés en la categoría mediante la barra deslizante que aparecerá. Cuando hayamos introducido las preferencias, pulsaremos el botón de continuar para introducir el tiempo disponible para la visita y recibir la ruta personalizada, cuyos objetos aparecerían en la pestaña de "Recomendaciones". Para la recomendación de las rutas se ha utilizado el algoritmo voraz, que accede a la base de datos de los objetos alojada en un servidor de Internet. De esta forma, los datos de los objetos pueden ser modificados cómodamente sin tener que actualizar la aplicación.

Realidad Aumentada

Debido a la peculiar distribución del museo, y a la escasa distancia entre las vitrinas que se exponen en él, no ha sido posible la aplicación de tecnologías de localización en interiores para guiar al usuario a través del museo. No obstante, se ha optado por un enfoque basado en marcadores que mejorará la experiencia del usuario durante la exploración del museo. Estos marcadores, colocados en distintas vitrinas, permiten al usuario acceder a contenidos multimedia relacionados con la vitrina, como pueden ser enlaces a páginas web, imágenes, o vídeos.

La aplicación está realizada utilizando el entorno de desarrollo de Wikitude Javascript, y los marcadores se alojan en el servidor de Wikitude, que permite descargar una copia de este conjunto de marcadores para utilizarlo en la aplicación. El usuario sólo tendrá que apuntar con la cámara de su *smartphone* al marcador para disfrutar de la experiencia de Realidad Aumentada.

En la Figura 4.12 podemos ver uno de estos marcadores, correspondiente a la vitrina 17, que contiene el perro robótico Sony AIBO ERS-7. A la derecha, la aplicación reconoce el marcador y muestra una imagen correspondiente a la vitrina en la que se encuentra, y un enlace con el que se accede al vídeo promocional lanzado en su estreno.

Figura 4.12: Realidad Aumentada en el MIGS

El código correspondiente a la aplicación puede encontrarse en los archivos adjuntos a esta memoria y en GitHub[1].

4.2. Caso de estudio: Recomendador de ocio en Madrid

Las aplicaciones de recomendación de turismo y ocio son muy frecuentes hoy en día, y muy útiles de cara a descubrir planes en ciudades o zonas que no conocemos. La motivación para abordar este caso de estudio era presentar un dominio que tuviese un conjunto reducido de puntos para poder ejecutar el algoritmo A* visto en el Capítulo 2.1.1 y obtener los resultados en un intervalo razonable de tiempo. Además, al tratarse de un dominio en exteriores, podremos incluir elementos de Realidad Aumentada mediante localización por GPS.

Nuestro objetivo será conseguir una buena recomendación en función de los gustos del usuario, utilizando para ello los algoritmos A* y voraz. Además de esto, se pretende enriquecer la experiencia con ayuda de técnicas de Realidad Aumentada, que permitan al usuario guiarse hacia los distintos puntos de interés recomendados. Para ello, disponemos de varias localizaciones en Madrid que atienden al espectro de gustos de una gran variedad

[1] `https://github.com/jaguirrepeman/MIGS`

de usuarios: desde monumentos como la Puerta del Sol hasta zonas de bares (La Latina), pasando por museos (El Prado) o iglesias (La Almudena).

4.2.1. Resolución del problema

Para realizar la recomendación procederemos de modo similar al anterior caso de estudio. En primer lugar, definiremos el grafo. Los 14 nodos que lo forman pueden verse en la Figura 4.13, y su lista detallada se encuentra en el Apéndice B. Al igual que en la Sección 4.1, cada nodo tendrá como atributos un *identificador* que indique de qué punto de interés se trata, una *descripción* del punto, un *tiempo* recomendado para visitar el punto y unas *etiquetas* que categorizan el punto.

En este caso no podemos obviar la distancia entre los distintos puntos de interés, ya que las distancias más cortas son de 400 metros, llegando incluso a varios kilómetros. Para calcular el tiempo correspondiente a cada arista hemos consultado las rutas en Google Maps. Recorreremos andando las distancias cortas (menos de 15 minutos) y en transporte público (autobús o metro) los recorridos que excedan esos 15 minutos.

Las distintas etiquetas que categorizarán los nodos del grafo y los gustos del usuario son las siguientes: *arte*, *cultura*, *entretenimiento*, *gastronomía*, *Historia*, *iglesia*, *infantil*, *monumento* y *museo*.

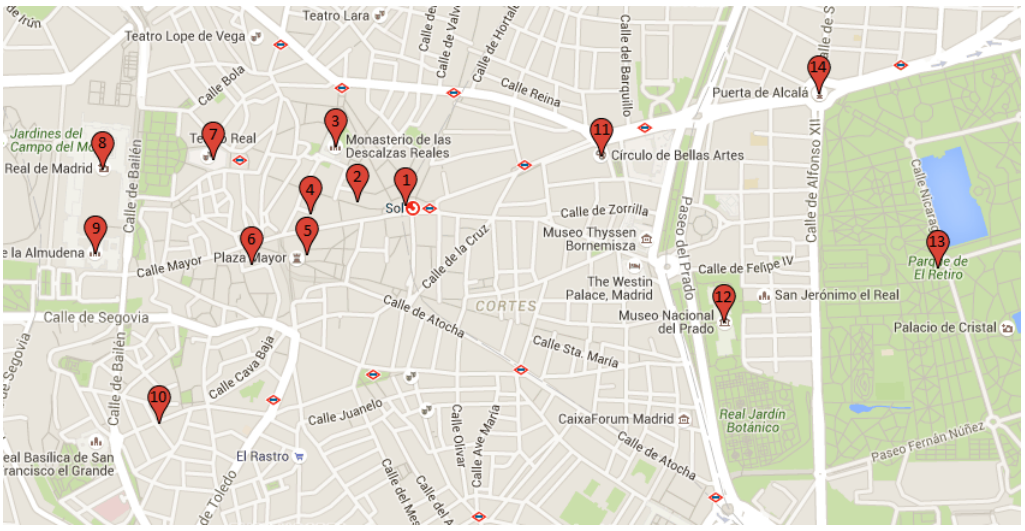

Figura 4.13: Recomendador de turismo en Madrid

4.2.2. Recorridos de prueba

Basándonos en las etiquetas creadas anteriormente, hemos realizado varias visitas prototípicas. Todos los recorridos analizados a continuación presuponen un límite de tiempo de 6 horas en las visitas.

Familia

En el primer recorrido, el grupo de visitantes será una familia con hijos pequeños, por lo que las preferencias serán *entretenimiento*, *gastronomía*, *infantil* y *monumento*. Veamos las rutas recomendadas por el algoritmo A* (verde) y voraz (azul) en la Figura 4.14.

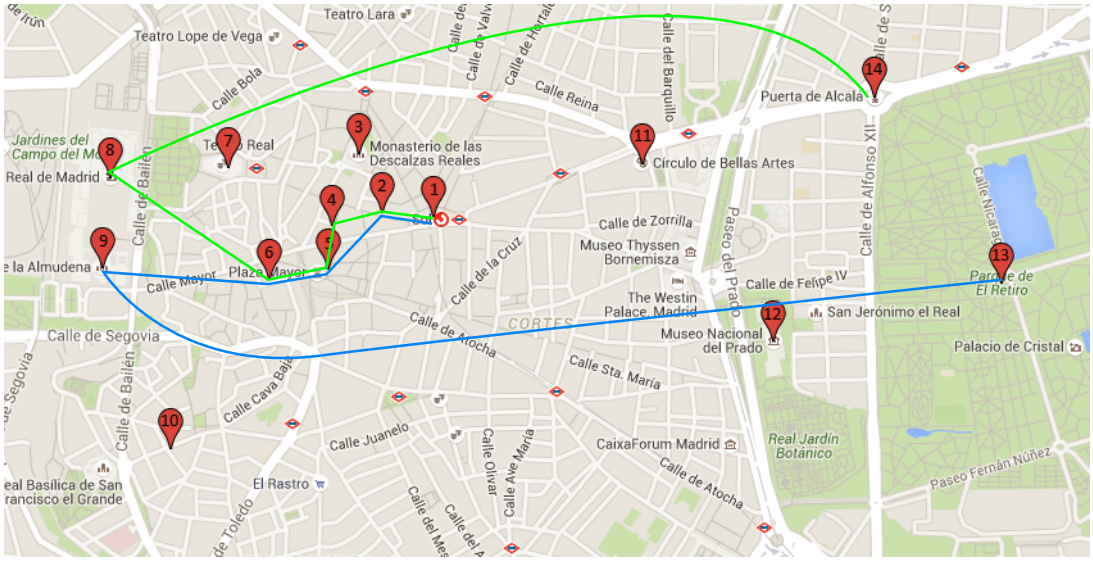

Figura 4.14: Recomendación para familias

Hay cuatro nodos elegidos por los dos algoritmos: la Puerta del Sol, la Casa Museo del Ratón Pérez, la Plaza Mayor y el Mercado de San Miguel. El alto valor de la etiqueta *infantil* lleva a los algoritmos a escoger la Casa Museo del Ratón Pérez, y lo mismo ocurre con la etiqueta *gastronomía* y el Mercado de San Miguel.

Por contra, el algoritmo voraz selecciona la Chocolatería San Ginés, el Palacio Real y la Puerta de Alcalá, mientras que el A* opta por la Catedral de la Almudena y el Parque del Retiro.

Analicemos los puntos visitados únicamente por el algoritmo voraz. La elección de la Chocolatería se apoya en que sus etiquetas son gastronomía e infantil, con alta importancia entre los gustos de los usuarios. Análogamente se explica la selección de la Puerta de Alcalá, con su alto valor en la categoría *monumento*. Por su parte, el Palacio Real está etiquetado en 3 categorías distintas, lo que le proporciona una gran cobertura.

Pasemos ahora a los puntos seleccionados sólo por el algoritmo A*. El Parque de El Retiro muestra un alto valor en *entretenimiento*, una etiqueta muy importante para los usuarios, y que apenas se encuentra en el resto del recorrido. Lo mismo ocurre con la Catedral de la Almudena, que presenta un alto valor en la etiqueta *iglesia*, que no aparece en el resto de puntos de interés visitados. Como la heurística utilizada por el A* tiene en cuenta la diversidad de las etiquetas cubiertas por la ruta, se explican estas elecciones.

Turismo

En este recorrido, el grupo de visitantes serán turistas, cuyas preferencias son principalmente *entretenimiento*, *gastronomía* y *monumento*, mostrando un interés nulo por la categoría *infantil*. Veamos las rutas recomendadas por el algoritmo A* (verde) y voraz (azul) en la Figura 4.15.

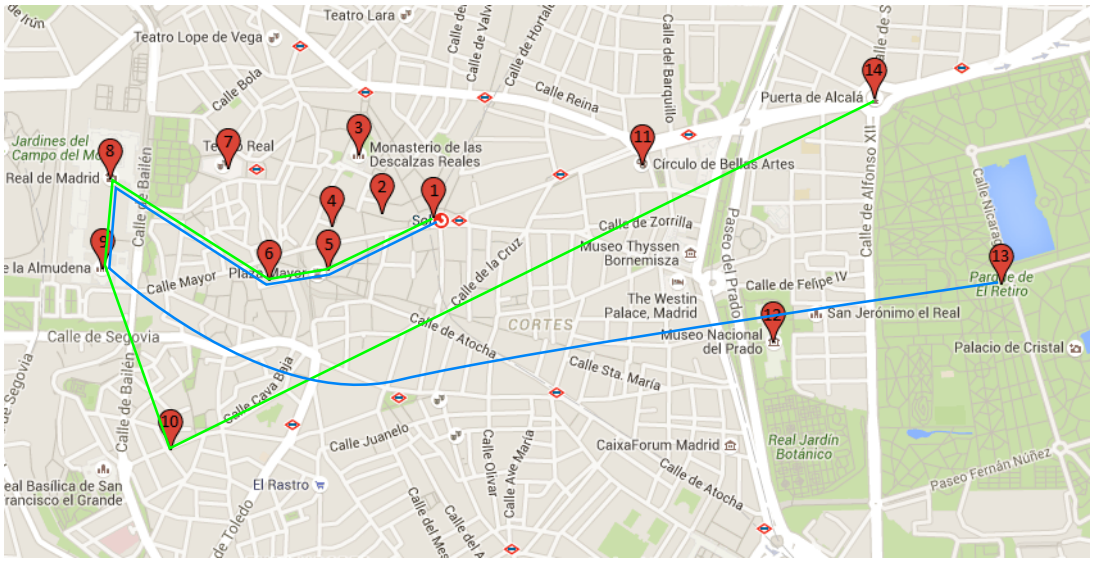

Figura 4.15: Recomendación para turistas

Hay cinco nodos elegidos por los dos algoritmos: la Puerta del Sol, la Plaza Mayor, el Mercado de San Miguel, el Palacio Real y la Catedral de la Almudena. El alto valor de la etiqueta *gastronomía* lleva a los algoritmos a escoger el Mercado de San Miguel, mientras que las elecciones del Palacio Real y la Catedral de la Almudena se sustentan en que cubren tres etiquetas distintas cada uno.

En este caso, el algoritmo voraz opta por visitar el barrio de La Latina, con buenos valores en *gastronomía* y *entretenimiento*, y la Puerta de Alcalá (*monumento*), mientras que el A* elige el Parque de El Retiro, con un valor muy alto en *entretenimiento*, categoría que apenas ha sido cubierta en el recorrido.

Mayores

Por último, nuestro grupo de visitantes serán personas mayores, con alto interés por las categorías de *iglesia*, *monumento* e *historia*, mostrando también un interés nulo por la categoría *infantil*, y valores bajos en *gastronomía* y *entretenimiento*. Veamos las rutas recomendadas por el algoritmo A* (verde) y voraz (azul) en la Figura 4.15.

De nuevo, tanto el algoritmo voraz como el A* eligen 5 nodos en común: la Puerta del Sol, la Plaza Mayor, el Teatro Real, la Catedral de la Almudena y la Puerta de Alcalá. El alto interés por las categorías de *historia*, *iglesia* y *monumento* lleva a la elección del

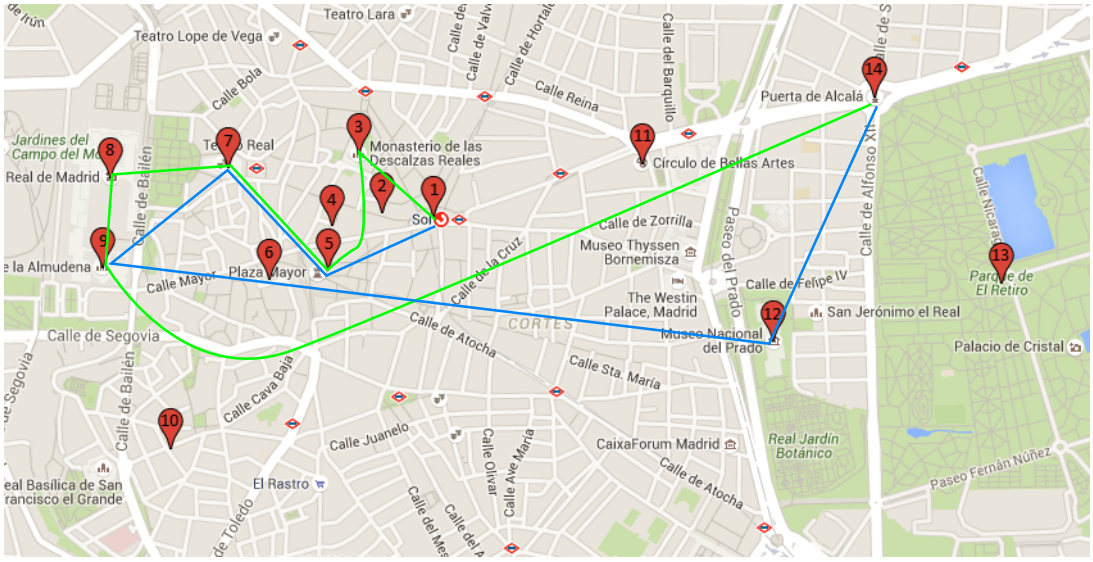

Figura 4.16: Recomendación para personas mayores

Teatro Real, la Catedral de la Almudena y la Puerta de Alcalá, respectivamente.

Para este grupo, el algoritmo voraz opta por visitar el Monasterio de las Descalzas Reales y el Palacio Real, apoyándose en los altos valores de las etiquetas *iglesia* e *historia* respectivamente. Por su parte, el algoritmo A* elige el Museo del Prado, que cubre la etiqueta de *arte*, no considerada en el resto del recorrido.

Conclusión de las visitas

En las tres visitas hemos visto cómo los algoritmos A* y voraz proporcionaban buenas recomendaciones en función de las preferencias de los grupos de usuarios a considerar. Ambos algoritmos coincidían en gran parte de la recomendación, variando ligeramente debido a las diferentes características de sus heurísticas. Como la heurística utilizada por el algoritmo A* se preocupa más por la variedad de las etiquetas, las elecciones que diferían del algoritmo voraz se debían precisamente a incluir categorías no consideradas en el resto del recorrido. Por su parte, la heurística del algoritmo voraz se preocupa de obtener la máxima cobertura en cada punto de interés sin tener en cuenta los demás, tratando de maximizar la satisfacción del usuario, en ocasiones a costa de recomendar rutas ligeramente monotemáticas.

4.2.3. Prototipo de aplicación con Realidad Aumentada

Con el objetivo de enriquecer la experiencia del usuario utilizando realidad aumentada, utilizaremos la señal GPS de nuestro teléfono para ubicar los diferentes puntos de interés. Facilitaremos el manejo alojando estos puntos en un servidor, al que nos conectaremos desde la aplicación.

Así, una vez hayamos obtenido las coordenadas de los diferentes puntos, mostraremos unas tarjetas en cada uno de estos puntos donde aparezca el nombre del punto, la distancia hasta él, y una descripción del lugar. También mostraremos un radar donde aparezcan los puntos, permitiendo ajustar la distancia máxima hasta la que se mostrarán los ítems.

En la Figura 4.17 podemos ver una captura de la aplicación donde se muestran varios tarjetas asociadas a puntos de interés recomendados (Círculo de Bellas Artes, Monasterio de las Descalzas Reales, Palacio Real). En la imagen de la derecha vemos la información correspondiente al Monasterio de las Descalzas Reales, así como la distancia a la que nos encontramos de él.

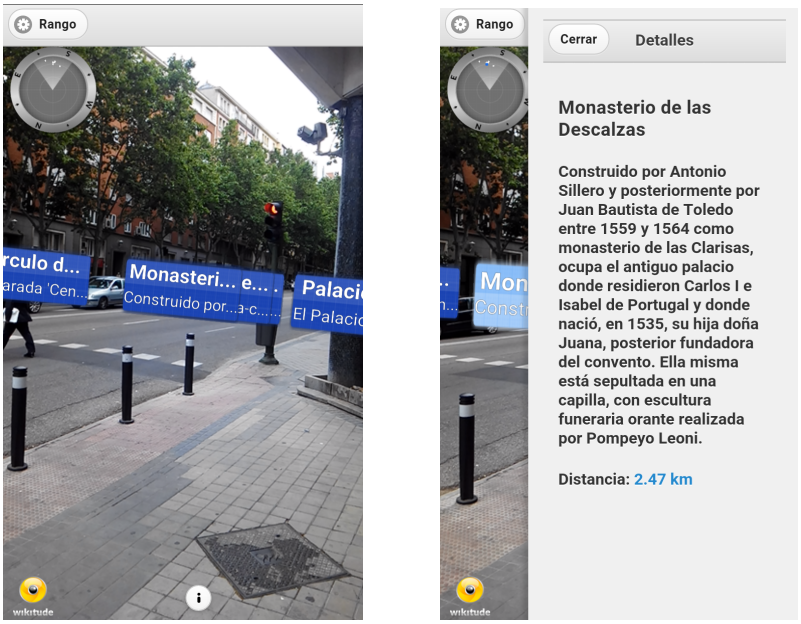

Figura 4.17: Realidad Aumentada en el recomendador en Madrid

4.2.4. Conclusiones

En este capítulo hemos podido explicado la implementación sobre dos casos de estudio muy diferentes entre sí del marco teórico definido en el Capítulo 3. En el Museo García Santesmases nos encontramos en un edificio de dos plantas, donde los puntos de interés apenas están separados unos pocos metros entre sí. En este caso optamos por generar las aristas automáticamente según la posición de los puntos en el mapa dado el poco impacto de las distancias en la elección de los puntos de la ruta. Dado que se detectaron 52 puntos de interés, la aplicación del algoritmo A* resultaba inviable, por lo que se optó tanto por el algoritmo voraz como por el basado en casos. La comparación experimental entre ambos algoritmos concluyó que las recomendaciones proporcionadas por el algoritmo voraz son ligeramente mejores, aunque el algoritmo CBR permite incluir conocimiento de situaciones reales y específicas del dominio.

Por otra parte, para mejorar la experiencia del usuario, y dado que la precisión de la

localización en interiores imposibilitaba guiar al usuario mediante Realidad Aumentada, se colocaron marcadores en las vitrinas del museo que permitían al usuario obtener información adicional de forma interactiva.

En cuanto al recomendador de turismo en Madrid, su reducido tamaño sí permitía la aplicación del algoritmo A*, junto al que se usó el algoritmo voraz. En esta ocasión, puesto que las distancias entre puntos podían alcanzar los 30 minutos, se decidió obtener esas distancias de forma precisa, utilizando Google Maps para calcular el tiempo requerido, ya fuese andando o en transporte público. Dada su localización en exteriores, es posible el uso de la señal GPS del *smartphone* para localizar los distintos puntos de interés y guiar al usuario hacia ellos.

Por tanto, aunque las diferencias entre los dos casos de estudio fuesen sustanciales, la amplitud del marco nos ha posibilitado encontrar aproximaciones para proporcionar rutas satisfactoriamente en ambos casos, aplicando los pasos enumerados en la Sección 3.3.

5

Conclusiones y trabajo futuro

5.1. Conclusiones

En este trabajo hemos planteado una solución genérica al problema de planificación de rutas personalizadas para individuos y grupos. Tras estudiar las técnicas de personalización de rutas, hemos propuesto tres algoritmos diferentes (A*, voraz y basado en casos) que nos han permitido resolver el problema.

Posteriormente, hemos aplicado el modelo teórico a dos casos concretos: el Museo García Santesmases y la personalización de rutas de turismo en Madrid. La aplicación de los algoritmos propuestos sobre estos casos ha permitido realizar una comparativa experimental entre ellos. Además, se han realizado prototipos de aplicaciones móviles que pretenden guiar al usuario durante el recorrido de las rutas en ambos dominios.

A continuación expondremos algunos resultados específicos obtenidos en el trabajo. Respecto a los algoritmos propuestos, A* proporciona una solución óptima, aunque no es eficaz para grafos con más de 15 nodos, pues sus tiempos de cálculo son muy grandes. El algoritmo voraz sí que se puede escalar a casos como el expuesto en la Sección 4.1, que contaba con 52 nodos. Por último, el algoritmo CBR permite incluir la información experta del dominio en la recomendación, realizando buenas recomendaciones con una base de casos relativamente pequeña.

En los experimentos en el Museo García Santesmases se ha comparado la aproximación CBR a la búsqueda heurística utilizando la cobertura para medir cómo se ajusta la ruta recomendada a las preferencias de los usuarios. Los resultados permiten sintetizar las ventajas de utilizar el enfoque CBR para resolver este problema. Puesto que el algoritmo voraz realiza una búsqueda exhaustiva, sus resultados en cobertura mejoran

los del enfoque CBR. No obstante, los casos han resultado una forma excelente de obtener conocimiento experto proviniente de situaciones reales, como el hecho de que en grandes grupos hay limitaciones de espacio en las vitrinas. Otra ventaja del algoritmo CBR es que funciona incluso con un conjunto vacío de preferencias, por lo que es aplicable con menos conocimiento en la consulta.

Sobre el caso de uso del recomendador de turismo en Madrid visto en la Sección 4.2, es interesante ver cómo el algoritmo A* y el voraz coinciden en la mitad de los puntos recomendados. Las diferencias entre ambos están fundamentadas en la preferencia del algoritmo A* por las rutas más variadas, mientras que el voraz se centra en recomendar al usuario las categorías que este prefiere, aunque resulte en rutas monotemáticas. Así, podrían ofrecerse ambas opciones al usuario para que este elija la que prefiera.

Por último, la aplicación de técnicas de Realidad Aumentada sobre las rutas personalizadas ha resultado una forma sencilla e intuitiva para que el usuario pueda seguir el camino indicado, ya sea porque recibe información sobre cómo llegar hasta el siguiente punto (en el recomendador de turismo en Madrid) o porque puede consultar información extra sobre los puntos que visita (en el Museo García Santesmases).

5.2. Trabajo futuro

Aunque se han cumplido los objetivos propuestos, este trabajo abre nuevas líneas de trabajo futuro. Se pretende extender el número de casos base utilizando un procedimiento de adquisición automática de casos, donde se almacenen los datos de las visitas del museo utilizando dispositivos de localización como *beacons*, que permiten detectar la posición del usuario. También se podrían estudiar otras medidas de similitud u otros métodos de adaptación más complejos. Así, se espera que la calidad de la solución (medida por la cobertura) se incremente al añadir más casos reales.

Por otra parte, en el caso de visitas grupales, las preferencias de los usuarios introducidas en las consultas están calculadas como media de los miembros del grupo, lo que resulta en una tarea tediosa de descripción de la entrada. Una alternativa es usar una descripción aproximada de las preferencias del grupo basada en visitas prototípicas, lo cual es adecuado para reutilizar soluciones CBR.

Los casos en el sistema CBR representan visitas reales, y los primeros experimentos con ellos mostraron que se puede simplificar la representación del mapa del museo donde todos los nodos están conectados a los demás: a pesar de que es teóricamente posible visitar un nodo después de cualquier otro, esto no ocurre en las visitas reales, pues en el camino a los nodos más lejanos, los usuarios siempre visitan nodos intermedios. Este conocimiento ha sido capturado observando los casos base y podría ser usado para simplificar el mapa agrupando diferentes objetos de cara a aplicar el algoritmo A*.

Respecto a las recomendaciones, sería interesante poder aplicar los conceptos del filtrado colaborativo. Tras realizar la ruta propuesta por el recomendador, los usuarios podrían valorar su grado de satisfacción con cada nodo visitado, y añadir esta información

al sistema para futuras consultas, pasando así de un sistema recomendador basado en contenido a uno híbrido.

La parte de Realidad Aumentada en el Museo García Santesmases también podría ser mejorada, ampliando el número de vitrinas que ofrecen contenidos de Realidad Aumentada y mejorando la calidad de esos contenidos, incluyendo animaciones o sonidos. Además, se propuso una idea (desestimada por su alta complejidad) consistente en realizar un guía virtual que llevase al usuario a través del museo, parándose en las vitrinas seleccionadas por el recomendador y proporcionando información adicional a la mostrada en la propia vitrina. En el Recomendador en Madrid se podrían refinar los contenidos de Realidad Aumentada, incluyendo más datos o interacción mediante botones virtuales.

Para localizar al usuario dentro del museo y así poder guiarle también serían de utilidad los *beacons* o cualquier sistema de posicionamiento en interiores (IPS, en sus siglas en inglés). No obstante, la tecnología relativa a este campo todavía no está lo suficientemente avanzada como para que el error relativo al posicionamiento sea asumible, sobre todo al tratarse de museos con tan poco espacio entre sus vitrinas.

Bibliografía

[1] Yifeng Zeng y col. «Optimal Route Search with the Coverage of Users' Preferences». En: *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*. AAAI Press, 2015, págs. 2118-2124. ISBN: 9781577357384.

[2] Kenneth H. Rosen. *Discrete Mathematics and Its Applications*. 5th. McGraw-Hill Higher Education, 2002. ISBN: 0072424346.

[3] Stuart Russell y Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009. ISBN: 0136042597.

[4] G.F. Luger. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Addison-Wesley, 2005. ISBN: 9780321263186. URL: `https://books.google.co.uk/books?id=QcTuJb7Hi40C`.

[5] Elaine Rich y Kevin Knight. *Artificial Intelligence*. 2nd. McGraw-Hill Higher Education, 1990. ISBN: 0070522634.

[6] *Informed search algorithms*. URL: `https://courses.cs.washington.edu/courses/csep573/11wi/lectures/03-hsearch.pdf`.

[7] Paul Resnick y Hal Varian. *Recommender systems*. Vol. 40. New York, NY, USA: ACM, 1997, págs. 56-58.

[8] Robin Burke. «The Adaptive Web». En: ed. por Peter Brusilovsky, Alfred Kobsa y Wolfgang Nejdl. Berlin, Heidelberg: Springer-Verlag, 2007. Cap. Hybrid Web Recommender Systems, págs. 377-408. ISBN: 978-3-540-72078-2. URL: `http://dl.acm.org/citation.cfm?id=1768197.1768211`.

[9] Francesco Ricci y col. *Recommender Systems Handbook*. 1st. New York, NY, USA: Springer-Verlag New York, Inc., 2010. ISBN: 0387858199, 9780387858197.

[10] Damianos Gavalas y col. «Mobile recommender systems in tourism». En: *Journal of Network and Computer Applications* 39 (2014), págs. 319-333.

[11] Robert Parviz Biuk-Aghai, Simon Fong y Yain-Whar Si. «Design of a recommender system for mobile tourism multimedia selection». En: *Internet Multimedia Services Architecture and Applications, 2008. IMSAA 2008. 2nd International Conference on*. IEEE. 2008, págs. 1-6.

[12] Liliana Ardissono y col. «Intrigue: personalized recommendation of tourist attractions for desktop and hand held devices». En: *Applied Artificial Intelligence* 17.8-9 (2003), págs. 687-714.

[13] Munmun De Choudhury y col. «Automatic construction of travel itineraries using social breadcrumbs». En: *Proceedings of the 21st ACM conference on Hypertext and hypermedia*. ACM. 2010, págs. 35-44.

[14] Pierpaolo Di Bitonto y col. «User Preferences in Tourist Itineraries Recommendation». En: *Proceedings Conference on Information and Communication Technologies in Tourism*. 2011.

[15] Lorraine McGinty y Barry Smyth. «Personalised route planning: A case-based approach». En: *Advances in Case-Based Reasoning*. Springer, 2000, págs. 431-443.

[16] Juan M. Corchado y col. «Development of CBR-BDI Agents: A Tourist Guide Application». En: *Advances in Case-Based Reasoning: 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004. Proceedings*. Ed. por Peter Funk. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, págs. 547-559. ISBN: 978-3-540-28631-8. DOI: 10.1007/978-3-540-28631-8_40. URL: http://dx.doi.org/10.1007/978-3-540-28631-8_40.

[17] Borko Furht, ed. *Handbook of Augmented Reality*. Springer, 2011. ISBN: 978-1-4614-0063-9. URL: http://dblp.uni-trier.de/db/books/daglib/0027797.html.

[18] Paul Milgram y col. «Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum». En: 1994, págs. 282-292.

[19] Morton L. Heilig. *Sensorama simulator - Patent 3050870*. 1962. URL: http://www.freepatentsonline.com/3050870.html.

[20] Ronald T. Azuma. «A survey of augmented reality». En: *Presence: Teleoperators and Virtual Environments* 6.4 (ago. de 1997), págs. 355-385.

[21] Daniel Wagner y Dieter Schmalstieg. «First Steps Towards Handheld Augmented Reality». En: *Proceedings of the 7th International Conference on Wearable Computers*. IEEE Computer Society Press, 2003, págs. 127-135. ISBN: 0-7695-2034-0. URL: http://www.ims.tuwien.ac.at/media/documents/publications/HandheldAR_ISWC03final.pdf.

[22] Gerhard Reitmayr y Tom Drummond. «Going out: Robust Model-based Tracking for Outdoor Augmented Reality». En: *Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality*. ISMAR '06. Washington, DC, USA: IEEE Computer Society, 2006, págs. 109-118. ISBN: 1-4244-0650-1. DOI: 10.1109/ISMAR.2006.297801. URL: http://dx.doi.org/10.1109/ISMAR.2006.297801.

[23] Georg Klein y David Murray. «Parallel Tracking and Mapping for Small AR Workspaces». En: *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. ISMAR '07. Washington, DC, USA: IEEE Computer Society, 2007, págs. 1-10. ISBN: 978-1-4244-1749-0. DOI: 10.1109/ISMAR.2007.4538852. URL: http://dx.doi.org/10.1109/ISMAR.2007.4538852.

[24] N. Navab y col. «First Deployments of Augmented Reality in Operating Rooms». En: *Computer* 45.7 (2012), págs. 48-55. ISSN: 0018-9162. DOI: 10.1109/MC.2012.75.

[25] Tobias Blum y col. «mirracle: An augmented reality magic mirror system for anatomy education.» En: *VR*. Ed. por Sabine Coquillart, Steven Feiner y Kiyoshi Kiyokawa. IEEE Computer Society, 2012, págs. 115-116. ISBN: 978-1-4673-1247-9. URL: http://dblp.uni-trier.de/db/conf/vr/vr2012.html#BlumKBN12.

[26] Steven Henderson y Steven Feiner. *Augmented Reality for Maintenance and Repair (ARMAR)*. Inf. téc. Technical Report 86500526647. United States Air Force Research Lab, 2007.

[27] *Anuncio de Mini utilizando Realidad Aumentada*. URL: http://technabob.com/blog/2008/12/17/mini-augmented-reality-ads-hit-newstands/\#.

[28] Vassilios Vlahakis y col. «Archeoguide: An Augmented Reality Guide for Archaeological Sites». En: *IEEE Comput. Graph. Appl.* 22.5 (sep. de 2002), págs. 52-60. ISSN: 0272-1716. DOI: 10.1109/MCG.2002.1028726. URL: http://dx.doi.org/10.1109/MCG.2002.1028726.

[29] Tim Bailey y Hugh Durrant-Whyte. «Simultaneous Localisation and Mapping (SLAM): Part II State of the Art». En: *Robotics & Automation Magazine, IEEE* 13.3 (sep. de 2006), págs. 108-117. ISSN: 1070-9932. DOI: 10.1109/mra.2006.1678144. URL: http://dx.doi.org/10.1109/mra.2006.1678144.

[30] *Wikitude 3D (SLAM)*. URL: http://www.wikitude.com/blog-wikitude-3d-tracking-beta/.

[31] Kevin Curran y col. «An Evaluation of Indoor Location Determination Technologies». En: *J. Locat. Based Serv.* 5.2 (jun. de 2011), págs. 61-78. ISSN: 1748-9725. DOI: 10.1080/17489725.2011.562927. URL: http://dx.doi.org/10.1080/17489725.2011.562927.

[32] Kenneth C. Cheung, Stephen S. Intille y Kent Larson. *An Inexpensive Bluetooth-Based Indoor Positioning Hack*. URL: http://architecture.mit.edu/house_n/documents/CheungIntilleLarson2006.pdf.

[33] M. Rey-López y col. «moreTourism: Mobile recommendations for tourism». En: *Consumer Electronics (ICCE), 2011 IEEE International Conference on*. 2011, págs. 347-348. DOI: 10.1109/ICCE.2011.5722620.

[34] «RAMCAT: Modelo para Generar Recomendaciones en un Sistema de Realidad Aumentada Contextual Basándose en las Preferencias del Turista». En: *PASOS. Revista de Turismo y Patrimonio Cultural* 13 (2015), págs. 649-668.

[35] Masato Kasahara, Kosuke Takano y Kin Li. «A Personalized Learning System with an AR Augmented Reality Browser for Ecosystem Fieldwork». En: *Proceedings of the 2014 IEEE 28th International Conference on Advanced Information Networking and Applications*. AINA '14. Washington, DC, USA: IEEE Computer Society, 2014, págs. 89-97. ISBN: 978-1-4799-3630-4. DOI: 10.1109/AINA.2014.16. URL: http://dx.doi.org/10.1109/AINA.2014.16.

[36] Amarjeet Singh y col. «Efficient planning of informative paths for multiple robots». En: *IJCAI*. Vol. 7. 2007, págs. 2204-2211.

[37] Chandra Chekuri y Martin Pal. «A Recursive Greedy Algorithm for Walks in Directed Graphs». En: *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*. FOCS '05. Washington, DC, USA: IEEE Computer Society, 2005, págs. 245-253. ISBN: 0-7695-2468-0. DOI: 10.1109/SFCS.2005.9. URL: http://dx.doi.org/10.1109/SFCS.2005.9.

[38] Agnar Aamodt y Enric Plaza. «Case-based Reasoning: Foundational Issues, Methodological Variations, and System Approaches». En: *AI Commun.* 7.1 (mar. de 1994), págs. 39-59. ISSN: 0921-7126. URL: http://dl.acm.org/citation.cfm?id=196108.196115.

[39] *El legado de José García Santesmases*. URL: http://www.scie.es/historia-progreso-informatica-cientifica-universitaria-espana/legado-jose-garcia-santesmases-1952-a-1977/.

Apéndices

A

Relación de objetos del Museo García Santesmases

1. Cartel de García Santesmases
   - Tiempo de visita: 2 minutos 30 segundos
   - Etiquetas: Historia 0,95; España 0,9; FDI 0,95
2. Supercomputador Cerebro
   - Tiempo de visita: 2 minutos 30 segundos
   - Etiquetas: Historia 0,8; España 0,85; Hardware 0,85; FDI 0,85; PCS 0,7
3. IBM Versión comercial
   - Tiempo de visita: 1 minuto 45 segundos
   - Etiquetas: Historia 0,3; PCs 0,7; FDI 0,7
4. IEA-FI 1973: Primer minicomputador de España
   - Tiempo de visita: 1 minuto 45 segundos
   - Etiquetas: Historia 0,7; España 0,8; PCs 0,8
5. Motorola Exor-ciser, ONTEL OP 1 (1979): Procesador de textos integrado, Centro de cálculo HP 9430A
   - Tiempo de visita: 2 minutos 45 segundos
   - Etiquetas: FDI 0,3 ; Docencia 0,2

6. IBM 7090 (1959) Primer computador comercial transistorizado
    - Tiempo de visita: 1 minuto 45 segundos
    - Etiquetas: FDI 0,3; Videojuegos 0,5; Cine 0,3
7. Cursos de automática de García Santesmases
    - Tiempo de visita: 1 minuto 15 segundos
    - Etiquetas: Docencia 0,6; España 0,4
8. Enseñanza asistida por computador
    - Tiempo de visita: 2 minutos
    - Etiquetas: Docencia 0,8; FDI 0,7; España 0,3; Historia 0,3
9. Calculadoras
    - Tiempo de visita: 1 minuto 30 segundos
    - Etiquetas: Cine 0,2; Ciencias 0,5
10. Friden EC-132 (1965) + Tubo de Rayos Catódicos
    - Tiempo de visita: 30 segundos
    - Etiquetas: Hardware 0,7; PCs 0,3; Ciencias 0,2
11. Calculadora de raíces cuadradas
    - Tiempo de visita: 30 segundos
    - Etiquetas: Curiosidad 0,6; Ciencias 0,3; Hardware 0,2
12. PDA's y tablets
    - Tiempo de visita: 30 segundos
    - Etiquetas: Ciencias 0,5
13. Periféricos de IBM
    - Tiempo de visita: 30 segundos
    - Etiquetas: Periféricos 0,7
14. Arte por computadora
    - Tiempo de visita: 1 minuto 45 segundos
    - Etiquetas: FDI 0,8; Arte 0,9; Ciencias 0,9
15. Unidades de cinta + Discos físicos
    - Tiempo de visita: 45 segundos
    - Etiquetas: Cine 0,7; Curiosidad 0,6

16. Teléfonos
    - Tiempo de visita: 1 minuto 45 segundos
    - Etiquetas: Curiosidad 0,5; FDI 0,3
17. Sony AIBO ERS-7 (2003): Perro robótico
    - Tiempo de visita: 1 minuto
    - Etiquetas: Videojuegos 0,3; Curiosidad 0,3
18. PDP 11
    - Tiempo de visita: 1 minuto 45 segundos
    - Etiquetas: FDI 0,3; Curiosidad 0,2; Redes 0,8
19. Ordenadores circa 1990 (Amstrad, Tulip, Tamdon, Toshiba)
    - Tiempo de visita: 1 minuto
    - Etiquetas: PCs 0,7; Curiosidad 0,3
20. Memorex 1337 + Fichas perforadas
    - Tiempo de visita: 1 minuto 30 segundos
    - Etiquetas: PCs 0,8; Historia 0,3; Curiosidad 0,4; Cine 0,2
21. Carteles
    - Tiempo de visita: 1 minuto
    - Etiquetas: Docencia 0,4
22. HP 21MX + Cinta de papel
    - Tiempo de visita: 1 minuto
    - Etiquetas: PCs 0,7; Redes 0,4; Cine 0,35
23. Simulador de centrales térmicas
    - Tiempo de visita: 1 minuto 30 segundos
    - Etiquetas: Curiosidad 0,7; Cine 0,4; Docencia 0,8; Periféricos 0,2
24. IBM-PC XT 1983
    - Tiempo de visita:1 minuto
    - Etiquetas: PCs 0,8
25. Plotter HP9872C
    - Tiempo de visita: 1 minuto
    - Etiquetas: PCs 0,5

26. HP 9000/319
    - Tiempo de visita:1 minuto
    - Etiquetas: PCs 0,65
27. HP 1000 (1982)
    - Tiempo de visita: 1 minuto
    - Etiquetas: Redes 0,7
28. Ordenadores Amstrad
    - Tiempo de visita: 4 minutos
    - Etiquetas: Videojuegos 0,95; Curiosidad 0,7; Arte 0,85; España 0,85; PCs 0,8
29. Consolas Atari y Nintendo
    - Tiempo de visita: 3 minutos
    - Etiquetas: Videojuegos 0,95; Curiosidad 0,7; FDI 0,6; Cine 0,5; Historia 0,3
30. Ordenadores Sony, Amstrad, ...
    - Tiempo de visita: 1 minuto 30 segundos
    - Etiquetas: Historia 0,4; Videojuegos 0,85; España 0,4; PCs 0,8
31. Máquina recreativa
    - Tiempo de visita: 2 minutos 15 segundos
    - Etiquetas: Curiosidad 0,5; Cine 0,7; Videojuegos 0,95
32. Cables
    - Tiempo de visita: 1 minuto 15 segundos
    - Etiquetas: FDI 0,4; Hardware 0,6
33. Procesadores de Intel
    - Tiempo de visita: 1 minuto 15 segundos
    - Etiquetas: FDI 0,4; Hardware 0,95
34. IBM 3097: refrigeración líquida
    - Tiempo de visita: 2 minutos 15 segundos
    - Etiquetas: Ciencias 0,8; Hardware 0,7
35. Disco duro 8 GB
    - Tiempo de visita: 45 segundos
    - Etiquetas: Almacenamiento 0,85

36. Plato 6 MB
    - Tiempo de visita: 1 minuto 30 segundos
    - Etiquetas: Cine 0,5; Almacenamiento 0,95; Historia 0,2
37. Memorias de ferrita y otros dispositivos de almacenamiento
    - Tiempo de visita: 1 minuto 15 segundos
    - Etiquetas: Almacenamiento 0,8; Historia 0,3
38. Transistores, ROMs, circuitos integrados, obleas
    - Tiempo de visita: 1 minuto 45 segundos
    - Etiquetas: Hardware 0,95; Curiosidad 0,3; Almacenamiento 0,3
39. Periféricos y conectividad
    - Tiempo de visita: 1 minuto 15 segundos
    - Etiquetas: Redes 0,3; Cine 0,3; Periféricos 0,5
40. Disquetes $3^1/_2$; $5^1/_4$
    - Tiempo de visita: 1 minuto
    - Etiquetas: Periféricos 0,9; Cine 0,2
41. Impresoras
    - Tiempo de visita: 1 minuto
    - Etiquetas: Periféricos 0,7
42. Periféricos: ratones, teclados, lápiz táctil
    - Tiempo de visita: 1 minuto 15 segundos
    - Etiquetas: Periféricos 0,8
43. Discos duros
    - Tiempo de visita: 1 minuto
    - Etiquetas: Almacenamiento 0,85
44. Discos duros antiguos
    - Tiempo de visita: 1 minuto
    - Etiquetas: Almacenamiento 0,9
45. PCs
    - Tiempo de visita: 2 minutos 30 segundos
    - Etiquetas: PCs 0,95; Historia 0,3; Videojuegos 0,35

46. IBM 5110 Computing System
    - Tiempo de visita: 2 minutos
    - Etiquetas: PCs 0,95; Periféricos 0,3
47. DEC AlphaServer 2100
    - Tiempo de visita: 1 minuto 15 segundos
    - Etiquetas: Servidores 0,85; Redes 0,8
48. Ordenadores de Apple
    - Tiempo de visita: 2 minutos
    - Etiquetas: PCs 0,95; Cine 0,5; Curiosidad 0,5
49. Servidores Unix
    - Tiempo de visita: 1 minuto 15 segundos
    - Etiquetas: Servidores 0,9; Redes 0,8
50. Cray Y-MP EL (1991)
    - Tiempo de visita: 2 minutos
    - Etiquetas: Servidores 0,9; Hardware 0,6
51. Origin 2000
    - Tiempo de visita: 1 minuto
    - Etiquetas: FDI 0,7; Servidores 0,8
52. Servidores actuales
    - Tiempo de visita: 1 minuto
    - Etiquetas: Servidores 0,9

B

Relación de puntos de interés en Madrid

1. Plaza Mayor
   - Tiempo de visita: 30 minutos
   - Etiquetas: Monumento 0,9; Historia 0,5
2. Puerta del Sol
   - Tiempo de visita: 20 minutos
   - Etiquetas: Historia 0,4; Monumento 0,8
3. Catedral de la Almudena
   - Tiempo de visita: 40 minutos
   - Etiquetas: Iglesia 0,95; Monumento 0,3; Arte 0,3
4. Chocolatería San Ginés
   - Tiempo de visita: 40 minutos
   - Etiquetas: Gastronomía 0,9; Infantil 0,3
5. Mercado de San Miguel
   - Tiempo de visita: 1 hora
   - Etiquetas: Gastronomía 0,8
6. Palacio Real
   - Tiempo de visita: 1 hora 10 minutos

- Etiquetas: Historia 0,8; Cultura 0,7

7. Museo del Prado
   - Tiempo de visita: 2 horas
   - Etiquetas: Museo 0,9; Cultura 0,6; Arte 0,9
8. Barrio de La Latina
   - Tiempo de visita: 1 hora 30 minutos
   - Etiquetas: Gastronomía 0,6; Entretenimiento 0,6
9. Casa Museo del Ratón Pérez
   - Tiempo de visita: 40 minutos
   - Etiquetas: Infantil 0,95; Museo 0,4; Entretenimiento 0,4
10. Teatro Real
    - Tiempo de visita: 1 hora 10 minutos
    - Etiquetas: Cultura 0,8; Historia 0,4
11. Círculo de Bellas Artes
    - Tiempo de visita: 30 minutos
    - Etiquetas: Arte 0,9; Entretenimiento 0,2
12. Puerta de Alcalá
    - Tiempo de visita: 15 minutos
    - Etiquetas: Monumento 0,8
13. Parque de El Retiro
    - Tiempo de visita: 2 horas
    - Etiquetas: Entretenimiento 0,7; Infantil 0,2
14. Monasterio de las Descalzas Reales
    - Tiempo de visita: 1 hora
    - Etiquetas: Iglesia 0,7; Historia 0,4; Arte 0,5