Laboratorio: planificación para un *rover* marciano

Objetivos

Con esta actividad se pretende resolver de forma automática un caso de planificación expresado el lenguaje PDDL. Para ello, es preciso ser capaz de analizar el ejemplo sobre el que se trabaja, generar soluciones utilizando el planificador, comparar resultados y modificar la definición del caso para añadir nuevas posibilidades.

Los objetivos de la actividad son:

* Especificación en el lenguaje PDDL de un problema del mundo real.
* Resolver el problema mediante un planificador de tipo *satisfactory* (no óptimo).
* Analizar la salida de la ejecución para evaluar las características del planificador (la búsqueda que realiza, su coste computacional en memoria o tiempo, etc.).

**Descripción**

Como base para este problema se va a utilizar el dominio *rovers,* de la competición ICAPS 2006.

Gerevini, A., Dimopoulos, Y., Haslum, P. y Saetti, A. (2006). *The International Planning Competition.* ICAPS. <http://idm-lab.org/wiki/icaps/ipc2006/deterministic/>

Sobre este dominio se propondrán preguntas y modificaciones.

El dominio modela uno o más *rovers* (vehículos autónomos de exploración espacial), que están situados sobre un planeta o satélite y tienen capacidad de movimiento, análisis de muestras de suelo y roca, y captura de imágenes. Los datos obtenidos se pueden transmitir a la Tierra al comunicar con un vehículo estático de descenso *(lander),* que mantiene un canal de comunicación con esta. El objetivo de los problemas que se plantean es transmitir información sobre algunos de esos análisis e imágenes recogidos en la misión (figuras 1 y 2).

Junto con la actividad, se proporcionan los ficheros que contienen el dominio y el problema base que el alumno analizará y modificará en las sucesivas partes de esta actividad.

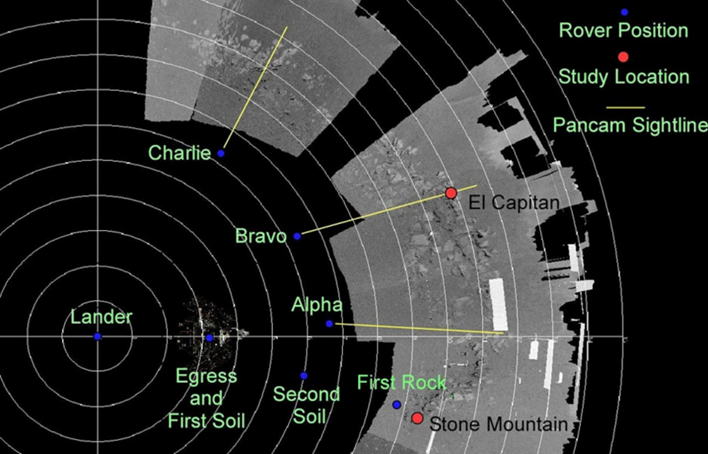


Figura 1: Diagrama del objetivos y posiciones del *rover Spirit*. Fuente: *Cold weather stalls Spirit rover’s sprint.* (2004). NBC News. <https://www.nbcnews.com/id/wbna4250175>

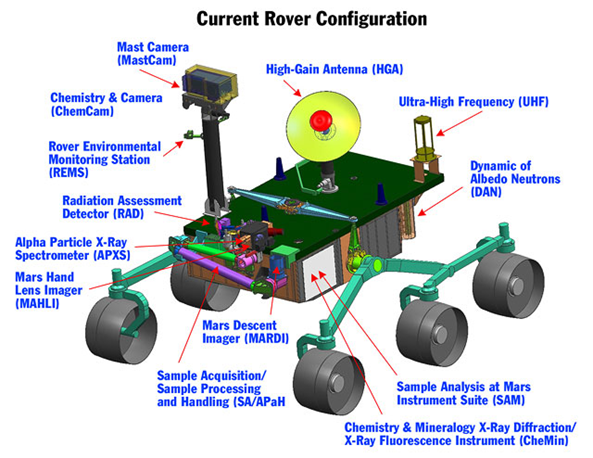


Figura 2: Instrumentos a bordo del *rover Curiosity*. Fuente: Paar, G.(2011). Robotics Vision for a Scouting Rover -PRoViScout. *EPSC Abstracts*, 6. <https://www.researchgate.net/publication/264885358_Robotics_Vision_for_a_Scouting_Rover_-PRoViScout>

Pautas de elaboración

La actividad consta de cuatro partes. En la primera se instala un entorno de desarrollo para PDDL que permite ejecutar planificadores de forma remota. En la segunda, se carga la especificación de un dominio y un problema de ejemplo que se proporcionan con la actividad y se pide cambiar la especificación de estados iniciales y objetivos para el problema. En la tercera, se pide ejecutar el planificador para resolver los problemas anteriores, comprender las métricas del planificador y sus salida. Finalmente, en la cuarta, se pide modificar el dominio del problema para incorporar una nueva funcionalidad.

Es importante que cuando se ejecute el planificador y se pida documentar el resultado se incluya siempre: búsqueda o búsquedas utilizadas, coste del plan, número de nodos generados y expandidos, y acciones que constituyen el plan encontrado.

En esta actividad se usará únicamente lenguaje PDDL versión 1.2.

Green, A. (s.f). *PDDL 1.2.* Planning.wiki - The AI Planning & PDDL Wiki

Home. <https://planning.wiki/ref/pddl>

Nota: es muy importante que evite los caracteres internacionales (tildes, etc.) en estos ficheros, incluyendo los comentarios.

**Parte 1. Instalación y prueba del entorno de desarrollo**

En el apartado Apéndice se informa del entorno de desarrollo para la realización de la actividad. Conviene que empiece analizando el problema para hacerse una idea de cómo es el caso realmente (disposición y conexiones entre los diversos puntos, etc.).

* 1.1. Indique qué acciones serían las potencialmente ejecutables en un primer paso por un planificador que opera mediante encadenamiento hacia delante, partiendo desde el estado inicial del problema.
* 1.2. Suponga que tenemos un planificador que opera mediante encadenamiento hacia atrás. Indique qué acciones serían las consideradas en un primer paso a partir del objetivo *(goal)* y cuáles serían los valores de los parámetros.
* 1.3. Ejecute el planificador para resolver el problema y documente el resultado.

**Parte 2. Modificación del estado inicial y objetivos**

En esta parte se modificará únicamente el fichero de problema entregado como ejemplo.

Se deben resolver las siguientes dos cuestiones:

* 2.1. Aparece un nuevo objetivo (objetivo2) que debe ser fotografiado en alta resolución y en color. Este solo es visible desde una ubicación accesible por el *rover*0 desde waypoint1. Desde dicha ubicación no es visible la base tipo *lander.*
* 2.2. Está disponible un segundo *rover* (rover1) que no tiene capacidad para análisis de suelo y rocas, y sí para tomar imágenes. Este *rover* empieza en la ubicación waypoint2.

Añada a la entrega el nuevo fichero «rovers\_parte2.1\_problema.pddl» con su respuesta comentada a la cuestión 2.1, y otro «rovers\_parte2.2\_problema.pddl» con la respuesta a la cuestión 2.2.

**Parte 3. Ejecución y evaluación del planificador**

Ejecute el planificador para generar un plan con cada uno de los ficheros anteriores. Responda a las siguientes preguntas:

* 3.1. Documente la ejecución del planificador para el caso 2.1. Compare con el plan original obtenido en la prueba de la instalación. ¿El *rover* realiza algún movimiento innecesario? ¿Por qué cree que ocurre esto?
* 3.2. Documente la ejecución del planificador para el caso 2.2 y compare con el plan original. ¿El nuevo plan utiliza el nuevo *rover* (rover1) introducido en el problema? ¿Puede evaluar por qué se da este efecto? Haga ahora que el objetivo de objetivo1 no sea visible desde la posición inicial del rover0, **ejecute el planificador y documente la ejecución de nuevo.** ¿Hay cambios? Si los hay, comente las razones de dichos cambios.

**Parte 4. Modificación del dominio**

Partiendo del caso con un *rover* o bien con dos, si lo ha conseguido resolver, debe incorporarse la siguiente modificación.

Importante: en la parte cuatro cabe mencionar que PDDL 1.2 no admite *fluents* numéricos, con lo que para evitar el uso de un planificador avanzado no podremos usar valores de ese tipo, ni funciones ni operaciones de comparación numérica (>, <, *increase*, etc.). Para ver cómo se puede resolver esta cuestión del nivel de carga se puede consultar el dominio *transport* de la competición IPC 2008.

* 4.1. Cada *rover* tiene una batería con cierto número de niveles de carga, que empieza cargada. Esta batería se va agotando cada vez que se hace una acción de movimientos y solo se puede recargar con una acción que se realiza en la ubicación donde está el *lander*. Nota: esto puede requerir cambios en el fichero del dominio y del problema.
* 4.2. Plantee un problema (estado inicial y objetivos) donde el efecto de esta batería se haga notar (por ejemplo, que se observe la vuelta al punto de recarga o que se vea cómo cada *rover* hace acciones separadas para aprovechar el nivel inicial).
* 4.3. Describa brevemente en la memoria cómo se ha resuelto esta parte (significado de los elementos introducidos, funcionamiento de las nuevas acciones, etc.).
* 4.4. Ejecute el planificador y documente el resultado.
* 4.5. Comente en la memoria el resultado e indique si se encuentra solución, si el tipo de solución obtenida corresponde con su expectativa, etc.

Añada a la entrega los nuevos ficheros **«rovers\_parte4\_dominio.pddl»** y **«rovers\_parte4\_problema.pddl».**

**Es imprescindible demostrar la autoría del código entregado**. En caso de partir de un dominio previo (diferente del *rovers*), **hay que poner la referencia e indicar con claridad la aportación verdaderamente hecha por el alumno**. Ningún dominio de competición cumple exactamente los requisitos planteados. Incorpore comentarios a los **ficheros .pddl** para explicar cómo utilizar su código.

Extensión y formato

* Código fuente de los ficheros con los nombres exactos pedidos en el enunciado: rovers\_parte2.1\_problema.pddl, rovers\_parte2.2\_problema.pddl, rovers\_parte4\_dominio.pddl, rovers\_parte4\_problema.pddl
* Memoria explicativa en formato PDF dividida en cuatro partes, en la que se incluya:
  + Respuestas a las cuestiones planteadas en cada parte.
  + Dificultades encontradas (especialmente problemas con instalación o ejecución del entorno).
  + Apéndice con las capturas de pantalla de la salida de la ejecución del planificador, que sirva de evidencia para mostrar que se ha realizado.
  + Referencias con normas APA.

**Extensión máxima de la memoria**

Ocho páginas (más apéndices opcionalmente).

Rúbrica

* El código PDDL debe ser de la autoría al menos parcial del estudiante. Está permitido tomar como base un dominio de competición (no una práctica anterior preexistente) siempre que se haga referencia al mismo.
* Cualquier página, libro u otro material consultado debe ser referenciado con las normas APA. Tenga en cuenta que no referenciar correctamente materiales usados literalmente será considerado plagio y, por lo tanto, afectará a la nota.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Descripción | Puntuación máxima  (puntos) | Peso  % |
| Criterio 1 | Instala y prueba el entorno de desarrollo, los códigos del dominio y el problema de la parte 1. Se documenta la ejecución y se responde a las cuestiones planteadas. | 4 | 40 % |
| Criterio 2 | Se realizan las modificaciones indicadas en la parte 2 y se responde a las preguntas indicadas demostrando conocimiento de la teoría. El código entregado está comentado y es trabajo original del alumno. | 2 | 20 % |
| Criterio 3 | Se elabora y documenta la ejecución de la planificación, al contestar razonadamente a las preguntas planteadas en la parte 3. | 2 | 20 % |
| Criterio 4 | Se modela una situación adicional planteada en la parte 4, que presenta código original del alumno y un caso de ejecución que verifique su buen funcionamiento. Se analizan y comentan resultados que demostren conocimiento de la teoría. | 2 | 20 % |
|  |  | **10** | **100 %** |

Apéndice

**Instalación del *software* de desarrollo recomendado**

Para el desarrollo y la ejecución se recomienda utilizar el entorno de desarrollo recomendado Visual Studio Code.

Microsoft. (2024). *Code editing. Redefined.* Visual Studio Code. https://code.visualstudio.com/

A continuación, instalar y habilitar el *plugin* Planning Domain Descripcion Language (PDDL) de Jan Dolejsi, que permite acceder a planificadores tanto instalados localmente como en forma de servicio remoto.

Dolejsi, J. (2024). *Planning Domain Description Language Support.* Visual Studio. https://marketplace.visualstudio.com/items?itemName=jan-dolejsi.pddl

Para esta actividad se debe configurar el planificador remoto del sitio: [solver.planning.domains](http://solver.planning.domains/). Para seleccionar el planificador en VSC teclear: **Control+Alt+P.** También es posible desarrollar el código y ejecutarlo en línea en [editor.planning.domains](http://editor.planning.domains/). Para ejecutar el planificador teclear: **Control-P,** seleccionar los ficheros relevantes y no introducir opciones de ejecución adicionales.

El planificador clásico de planning-domains se denomina siw-then-bfs y soporta PDDL 1.2. No soporta por tanto *fluents* numéricos. Está optimizado para ejecutar con rapidez y tiene fijado un plazo máximo de tiempo al cabo de cual puede terminar la búsqueda incluso aunque el problema tenga solución.

Para conocer la parte de PDDL que este planificador interpreta correctamente, se puede consultar en la siguiente página: <https://planning.wiki/ref/pddl/requirements>.

Green, A. y Dolejsi, J. (2024). *Requirements (PDDL 1.2).* Planning wiki. https://planning.wiki/ref/pddl/requirements

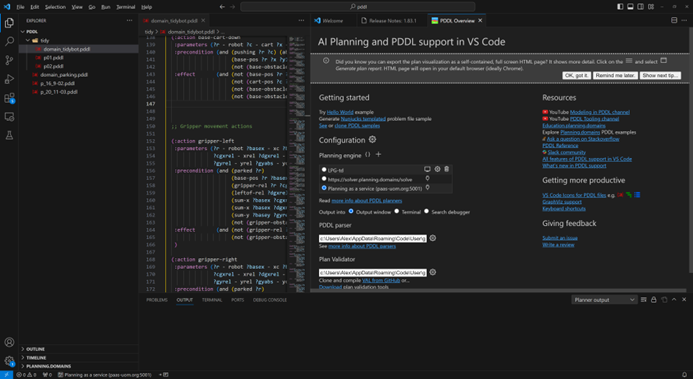


Figura 1. Ejemplo 1. Fuente: Green y Dolejsi, 2024.

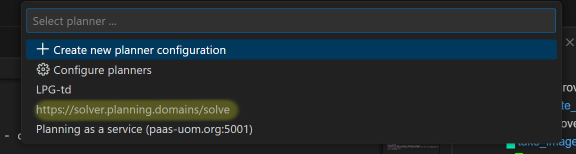


Figura 2. Ejemplo 2. Fuente: Green y Dolejsi, 2024.