Extensión para un generador de instrucciones GIVE: Landmarks en Navegación

María Celeste Gunski, Daniel Koile

Departamento de Computación - Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Pabellón I - Ciudad Universitaria - (C1428EGA) - Buenos Aires - Argentina
cgunski@gmail.com, dkoile@gmail.com
http://www.dc.uba.ar

Resumen El desafío GIVE consiste en generar automáticamente instrucciones en lenguaje natural para guiar a un jugador a través de habitaciones de un mundo virtual con el objetivo de conseguir un trofeo que se halla escondido en una de las habitaciones. Actualmente, para la generación de las instrucciones, dicho sistema utiliza como referencias para la orientación únicamente objetos del mundo con los que el jugador interactúa (e.g.: botones que tiene que apretar, puertas que tiene que atravezar). La modificación que se logra en el presente proyecto es la inclusión de *landmarks* en la navegación del jugador. Los landmarks son objetos presentes en las habitaciones pero con los que el jugador no puede interactuar. Para incluirlos en las instrucciones del sistema se consideraron diferentes conjuntos de objetos y filtros para estos conjuntos, de manera que finalmente fueran útiles para dar direcciones mas precisas en la navegación del mundo.

Keywords: GIVE, GLN, Landmarks, Generación de Instrucciones

1. Introducción

El problema de Generación de Lenguaje Natural (GLN) consiste en producir texto comprensible en lenguaje natural (Inglés, Español) de manera automática a partir de información no lingüística (por ej.: datos estadísticos, bases de datos). Los sistemas de GLN toman como entrada datos no lingüísticos y generan como salida una porción de texto en lenguaje natural.
Las tareas básicas que realiza un sistema de GLN son:

Planificación del documento (Document planning) En esta etapa se determina qué información se incluye en el texto de salida y cómo estará estructurado el documento.
- Determinación de Contenido (Content determination)
- Estructuración del Documento (Document structuring)

Microplanificación (Microplanning) Se define de qué manera se referenciarán los objetos, se selecciona qué palabras utilizar, se decide si se utilizarán recursos gramaticales y cuáles.
- Agregación (Aggregation)
- Lexicalización (Lexicalisation)
- Generación de Expresiones Referenciales (Referring expression generation)

Surface realization Se adapta el contenido del documento al idioma de salida de manera que sea gramaticalmente correcto y exprese el significado deseado.
- Realización Lingüística (Linguistic realisation)
- Realización de Estructura (Structure realisation)

El GIVE (Generating Instructions in Virtual Environments) Challenge es un evento organizado anualmente desde el año 2008 para evaluar sistemas de GLN. El objetivo es comparar sistemas que generan instrucciones en idioma Inglés para guiar a un jugador a través de un laberinto de habitaciones en un entorno 3D, a fin de que el mismo encuentre un trofeo escondido en una caja fuerte dentro de una de las habitaciones.

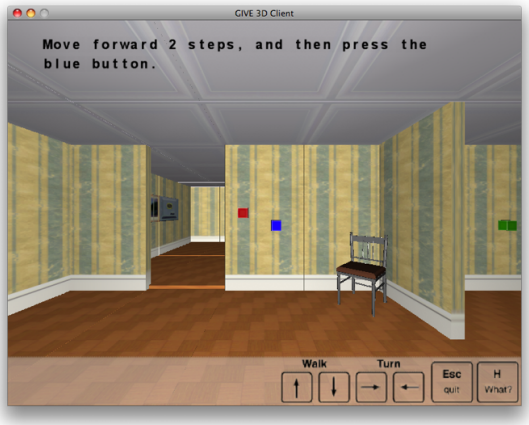

Figura 1: Vista del jugador durante un GIVE Challenge

Para generar las instrucciones el sistema primero elabora un *plan* de acciones a realizar por el jugador, el cual se va actualizando de acuerdo a cómo el jugador se desenvuelve dentro del mundo. Este proceso se denomina *planning* y entre las acciones que puede realizar el jugador se encuentran, por ejemplo: `move, manipulate, take`.

En cada habitación el jugador debe interactuar con distintos objetos para, mediante objetivos intermedios (abrir puertas, desactivar alarmas, etc.), finalmente alcanzar el objetivo final de tomar el trofeo. El sistema de base provisto por GIVE utiliza como referencias para la generación de las instrucciones de los objetivos intermedios en cada habitación, únicamente aquellos objetos con los que el jugador precisa interactuar. En este trabajo extendemos el sistema de base para tomar en cuenta también otros objetos presentes en la habitación que puedan ser útiles para orientar mejor al jugador hacia el objetivo.

En la primer iteración del challenge[3] (GIVE-1) la navegación era discreta, el mundo consistía en una cuadrícula de baldosas y el movimiento del jugador se limitaba a moverse a una baldosa contigua, además las direcciones de movimiento también estaban limitadas por giros de 90° (e.g. no se podía avanzar en diagonal).

Para la segunda iteración[4] (GIVE-2) la navegación es continua, el jugador se puede mover libremente y en cualquier dirección.

La extensión que buscamos incluir demostró ser efectiva en los entornos discretos de GIVE-1 [5] y es interesante analizar cómo puede resultar en entornos continuos como los de GIVE-2.

2. Desarrollando la extensión

Antes de comenzar con el desarrollo de las modificaciones veamos un ejemplo de las instrucciones que genera el sistema GIVE y cómo deberían modificarse con la extensión. Al inicio de la interacción el jugador se encuentra viendo la figura 2.

Actualmente el sistema (NLG) genera lo siguiente:

```
NLG: Go straight.
User: [goes straight towards a chair]
NLG: Excellent! Look around until you see a button near a green thing,
NLG: then go towards it.
NLG: It should be in front of you slightly to the right.
User: [turns right]
User:[goes straight towards the red button by a plant]
NLG: Excellent! Press the button
```

Figura 2: Inicio de la interacción

Se espera que luego de la implementación, la salida sea similar a la siguiente:

```
NLG: Go towards the plant.
User: [goes towards the plant]
NLG: Excellent! Look around until you see a button near a green thing,
NLG: then go towards it.
NLG: It should be in front of you slightly to the right.
User: [turns right]
User:[goes straight towards the red button]
NLG: Excellent! Press the button
```

Partiendo del sistema de base provisto por GIVE, se busca modificar la forma en que el sistema determina el contenido a transmitir. Para ello es necesario agregar al sistema la capacidad de utilizar, además de los objetos interactuables, aquellos con los que el jugador no puede interactuar denominados *landmarks*.

En el sistema de base los únicos objetos nombrados son aquellos con los que el jugador debe interactuar para ir cumpliendo los objetivos. Con lo cual, en lo primero en que nos enfocamos fue en agregar al conjunto de objetos a tener en cuenta para las instrucciones, aquellos otros objetos que se encontraban en la habitación y que pudieran servir para orientar mejor al jugador.

Inicialmente se tomaron todos los elementos que se encontraban en el campo de visión del jugador y se seleccionaba el que se encontraba mas cercano en distancia euclideana. Pudimos observar que, si bien el sistema tomaba en cuenta otros objetos en el mundo, lo hacía con poco criterio, ya que si el jugador se dirigía hacia donde indicaba la instrucción, no siempre se acercaba más al objetivo. Y si llegaba hasta donde se encontraba el landmark el sistema tampoco realizaba ninguna corrección.

El problema radica en que en este punto estamos pidiendo el más cercano dentro de los objetos visibles, pero si, por ejemplo, el jugador se encuentra mirando hacia el lado contrario al objetivo, el sistema lo puede dirigir en esa dirección, lo que lo termina alejando del objetivo. Lo que se puede ver, por ejemplo, en la figura 3, donde el objetivo es un botón azul a la izquierda del jugador (fuera de su campo visual) y el sistema le indica que se dirija hacia el cuadro. Por lo tanto, agregamos la condición de que el objeto esté más cerca del objetivo que el jugador.

Cuando tenemos en cuenta la posición del jugador al momento de seleccionar los objetos cercanos a la región objetivo, aún considerando "todos" los objetos cercanos al iniciar la búsqueda, nos encontramos con que el concepto de más cerca que utilizamos puede ser engañoso, ya que al utilizar coordenadas euclideanas se podría estar considerando algún objeto que tal vez se encuentre en alguna habitación contigua justo del otro lado de una pared, por ejemplo.

Para evitar considerar estos objetos, se inicia la búsqueda con los objetos que se encuentren dentro de la habitación actual. De esta manera, limitamos la selección a los objetos dentro de la

Figura 3: Alejamiento del objetivo

habitación y los filtramos de acuerdo a la distancia euclideana entre cada objeto y la región de destino, teniendo en cuenta además la posición del jugador para evitar los landmarks que puedan alejar potencialmente al jugador del objetivo.

También pudimos observar que si hay más de un objeto referenciable en la región de destino, el sistema genera instrucciones que pueden llevar a confusiones del jugador al utilizar cualquiera de estos objetos en lugar de utilizar el objetivo. En la figura 4 observamos un ejemplo de lo mencionado, el objetivo es dirigir al jugador hacia el botón azul ubicado en la pared de la derecha,

Figura 4: Mas de un botón en la región destino

en la imagen se puede ver que en la región de destino hay mas de un objeto manipulable (en este caso botones) y el sistema selecciona alguno de ellos (en el caso del ejemplo, el de color verde) lo que puede generar confusión al jugador ya que el mismo podría acercarse y presionar directamente el de la referencia, lo cual no es lo deseado. Para solucionar este inconveniente decidimos que sólo se referencien objetos que sean potencialmente objetivos (como un botón) cuando efectivamente es el próximo objeto a manipular. En cualquier otro caso, se usan únicamente los landmarks como referencia. Esto lo logramos usando un método que filtra los objetos dejando únicamente los "pasivos" (no manipulables), y si el próximo paso del plan es manipular un objeto, entonces utilizar directamente dicho objeto como referencia.
En la figura 5 se ve reflejado cómo afecta el cambio al sistema: ante la misma situación, en que

se tienen varios objetos manipulables, el sistema usa como referencia el objetivo y direcciona al jugador directamente hacia el botón azul.

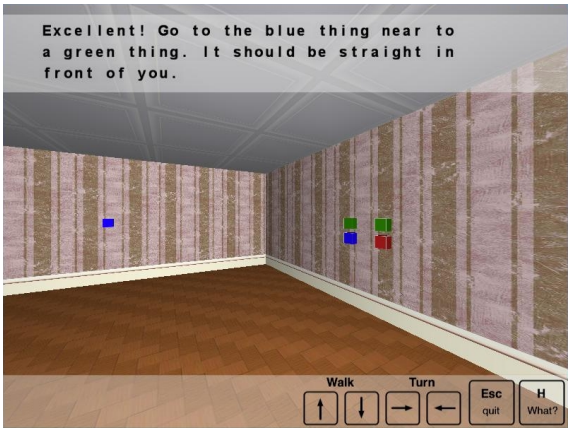

Figura 5: Mas de un botón en la región destino

En la versión finalmente implementada, se construye el conjunto de objetos que se utilizarán para la determinación de las instrucciones obteniendo los objetos cercanos a la región de destino utilizando el método `selectCloseObjects`, generado por nosotros que devuelve los objetos que se encuentren *cerca* según el resultado del método `isNear` provisto por GIVE.

Luego se filtran dichos objetos con un método (`filterLandmarksAndTarget`) que deja en el listado solamente los objetos no manipulables y además si en el siguiente paso hay que manipular un objeto, agrega el mismo al inicio de la lista para garantizar que tenga prioridad de ser elegido para las referencias. Este método también tiene en cuenta el caso especial del trofeo, el cual tiene prioridad sobre todos los demás.

Finalmente, se buscan aquellos visibles por el jugador y se los vuelve a ordenar por su proximidad. Si hay objetos visibles y cerca, se obtiene el primero de ellos (que al estar ordenados es el más cercano al objetivo) para armar la instrucción. Si los objetos más cercanos no están visibles, se toma el primero de ellos y se arma una expresión que indica al jugador que lo busque dentro de la habitación. Finalmente, si no hay objetos cercanos a la región de destino se orienta al jugador para que avance a la siguiente región.

Adicionalmente se agrega un poco mas de código para que en caso de que se esté en una habitación con varias puertas, se oriente mejor al jugador hacia qué puerta debe dirigirse. Esto resulta muy útil también cuando se está en un pasillo con diferentes direcciones a las que se podría avanzar, en mundos con muchas puertas ayuda para decidir hacia donde moverse.
A continuación se puede ver un ejemplo de esto, la figura 6a se extrajo del sistema de base y se puede ver que ante dos puertas hacia donde podría ir el jugador, el sistema no indica hacia cuál. Sin embargo, en la figura 6b el sistema que utiliza landmarks lo dirige a la puerta correspondiente, que es la que se encuentra justo al frente.

(a) Sistema de base (b) Sistema con landmarks

Figura 6: Pasillo con varias puertas

2.1. Calculando proximidad

A partir de las diferentes pruebas surgió más de una vez la necesidad de crear un método para filtrar los objetos que se podían utilizar como landmarks de acuerdo a su proximidad respecto de la región de destino, como resultado, se generó un conjunto de estos métodos y a continuación se detalla qué criterio utiliza cada uno de ellos:

getClosestObjectsEuclideano Selecciona los objetos del conjunto que están a distancia menor a 2 o a la distancia a la que se encuentra la posición pasada por parámetro. Esto es para que descarte los objetos que se encuentren mas alejados del objetivo que la posición en la que se encuentra el jugador.
Se utilizó en una de las versiones de desarrollo para seleccionar los objetos que se encontraban cerca de la región de destino pero teniendo en cuenta a su vez la distancia a la que se encontraba el jugador.

selectNearestObjectEuclideano Devuelve el objeto más cercano de los que se encuentran cerca de la posición dada, tomando como medida la distancia euclideana.
Se utiliza como método auxiliar para ordenar los objetos de acuerdo a su distancia euclideana respecto de la región de destino.

sortByProximity Ordena los objetos pasados por parámetro de acuerdo al criterio de proximidad especificado por el método `selectNearestObjectEuclideano`.
Este método es el utilizado en la versión final para ordenar los objetos de manera que el más cercano a la región de destino se encuentre en la primera posición de la lista.

selectNearestObject Devuelve el objeto más cercano de los que se encuentran cerca de la posición dada de acuerdo al método `isNear`.

selectCloseObjects Devuelve todos los objetos que se encuentran cerca de la posición dada de acuerdo al método `isNear`.
Se utiliza en la versión implementada para obtener el conjunto inicial de objetos que luego serán filtrados y seleccionados para utilizarlos en las instrucciones.

La necesidad de estos cálculos proviene de que, por el tipo de discretización que tiene GIVE-2 mencionado anteriormente, las regiones del mundo son amplias y pueden contener más de un landmark, en GIVE-1 al tener una discretización mas fina, las regiones son mas pequeñas y en cada una solo podía haber un landmark lo que facilitaba la localización, además del hecho que el jugador se podía mover solo a pasos discretos.

3. Evaluando el sistema

- En la figura 7 se puede observar cómo el sistema extendido utiliza como referencia una planta ubicada cerca de la región de destino, que en este caso es la puerta.

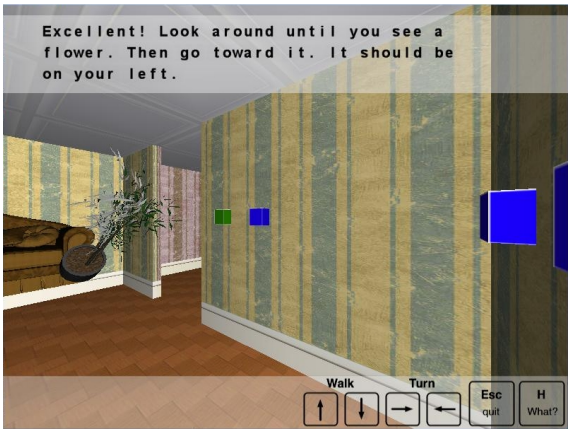

Figura 7: Utilización de una planta como landmark

- Podemos ver en la figura 8 cómo el sistema utiliza el trofeo como referencia aunque éste no esté en el campo de visión del jugador. Esto forma parte del criterio de selección de objetos utilizado en la extensión.

Figura 8: Si el trofeo está descubierto se lo utiliza como referencia

- Todavía quedan instrucciones ambiguas en algunos casos, como el de la figura 9, en la cual hay un conjunto de botones azules alrededor de uno verde, y uno de esos botones es el objetivo, pero el sistema no indica específicamente cuál de ellos hay que presionar. Esto podría formar parte de una futura extensión en la cual se utilice, por ejemplo, el botón verde para dar la localización precisa del objetivo con una instrucción como *"Press de blue button on the right of the green one"*

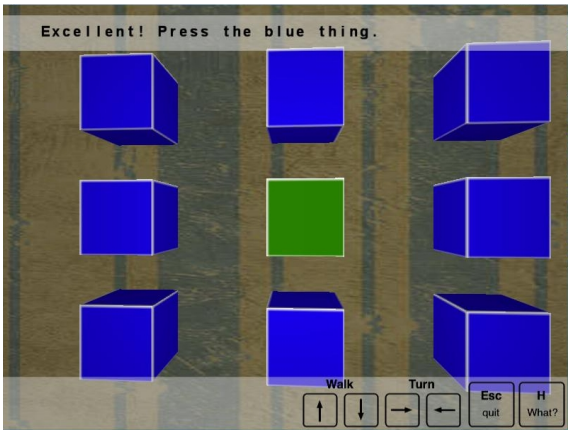

Figura 9: Varios botones del mismo color en la región de destino

- Algo similar ocurre en la figura 10 donde el objetivo es el botón de más a la derecha de los que se encuentran en la pared de la izquierda.

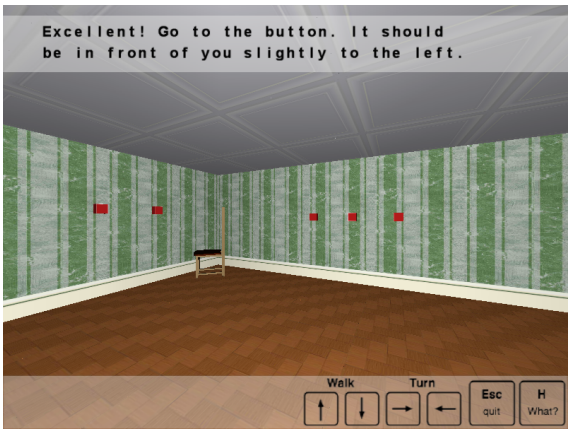

Figura 10: Varios botones del mismo color en el campo visual

4. Conclusiones y trabajo futuro

Para incluir landmarks a la navegación hay dos puntos fundamentales a tener en cuenta: el conjunto de objetos con los que se van a generar las instrucciones y el análisis de las distancias y posiciones del jugador, los objetivos a alcanzar y los objetos dentro de la habitación.

En el primer punto hay que tener en cuenta no solo los objetos cercanos a la región de destino sino también aquellos que se encuentren dentro de la habitación que puedan ayudar al jugador a dirigirse a una posición desde la cual tenga un mejor acceso al objetivo. Es importante también no quedarse únicamente con los objetos visibles, ya que el jugador podría estar cerca del objetivo pero orientado hacia otra dirección y en tal caso el objeto visible no constituye la mejor opción.

Aquí es donde entra en juego la importancia del segundo punto mencionado anteriormente, al analizar las posiciones de cada objeto en la habitación, del jugador y del objetivo, se incrementa la posibilidad de generar instrucciones más precisas en el sentido de que permiten al jugador ubicar el objetivo mas facilmente. Es importante utilizar métodos de medición de distancias para comparar la proximidad de los distintos objetos y poder seleccionar aquél que resulte mas conveniente en cada caso. En un entorno continuo, no es suficiente basarse en regiones discretas para identificar los landmarks relevantes, el cálculo de distancias es crucial.

Finalmente, es también necesario tener en cuenta que, dependiendo de qué método se utilice para la medición de distancias, hay que modificar la condición de éxito de las acciones realizadas, ya que si se pasa por alto esta modificación es posible que la navegación con landmarks no tenga el éxito esperado, porque no se estaría reconociendo el movimiento hacia el landmark como un movimiento exitoso.

Todavía quedan varias extensiones que se pueden realizar al sistema con landmarks para que sea aún más específico en el direccionamiento, una de ellas es la mencionada en la sección anterior relacionada a la figura 9, en la cual se podría utilizar la localización espacial (e.g.: a la derecha, arriba, abajo) del objetivo respecto de otro objeto en la misma región. También sería interesante agregar otro tipo de referencias, como las paredes sobre las que estan ubicados los objetos (lo que podría ser útil en el caso de la figura 10).

Referencias

1. C. Areces, L. Benotti, 2010. Generación de Lenguaje Natural y Aplicaciones. Escuela de Lingüística Computacional 2010 `http://www.glyc.dc.uba.ar/elic2010/curso2.php`. Buenos Aires, Argentina.
2. E. Reiter, R. Dale. Building Applied Natural Language Generation Systems. Cambridge University Press, 2000.
3. D. Byron, A. Koller, K. Striegnitz, J. Cassell, R. Dale, J. Moore, and J. Oberlander, 2009. Report on the first NGL challenge on generating instructions in virtual environments (GIVE). In Proceedings of the 12th European Workshop on Natural Language Generation (ENLG2009), Athens, Greece.
4. A. Koller, K. Striegnitz, A. Gargett, D. Byron, J. Cassell, R. Dale, J. Moore, and J. Oberlander, 2009. . Report on the Second NLG Challenge on Generating Instructions in Virtual Environments (GIVE-2). In Proceedings of the 6th International Natural Language Generation Conference (INLG 2010), Dublin, Ireland.
5. K. Striegnitz, F. Majda, 2009. Landmarks in Navigation Instructions for a Virtual Environment. In Proceedings of the 12th European Workshop on Natural Language Generation (ENLG2009), Athens, Greece.