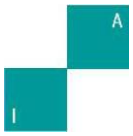
Inteligencia Artificial 17(54), 21-29
doi: 10.4114/ia.v17i54.1093



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

# Proyecto arquitectónico mediante gramáticas de formas sencillas y aprendizaje.

Manuela Ruiz-Montiel[*], Dr. Javier Boned[**], Dr. Juan Gavilanes[**], Dr. Eduardo Jiménez[**], Dr. Lawrence Mandow[*], Dr. José-Luis Pérez-de-la-Cruz[*].

*ETSI Informática-**ETS Arquitectura, Universidad de Málaga, España.
mruiz@lcc.uma.es,{fjboned,jgavilanes,eduardo_jm}@uma.es,{lawrence,perez}@lcc.uma.es

**Abstract.** This work presents a proposal for the automatic generation of architectural design. This scheme is based on the training of simple shape grammars through reinforcement learning technics. Finally, the results of the implemented system by this technic for the generation of dwelling design with certain restrictions are presented and analyzed.

**Resumen.** En este artículo se presenta una propuesta para la generación automática de planos arquitectónicos. La propuesta se basa en el entrenamiento de gramáticas de formas muy sencillas mediante técnicas de aprendizaje por refuerzo. Se presentan y analizan los resultados de un sistema implementado mediante esta técnica para la generación de planos de viviendas unifamiliares sujetas a ciertas restricciones.

**Keywords:** Computational Design, Architecture, Shape Grammars, Reinforcement Learning.
**Palabras clave:** Diseño Computacional, Arquitectura, Gramáticas de Formas, Aprendizaje por Refuerzo.

## 1 Introducción.

La Inteligencia Artificial ha abordado el problema del diseño ingenieril al menos desde la década de 1960 [1], [2]. A lo largo de estos años ha quedado claro que la resolución de problemas de diseño es una tarea especialmente compleja: se trata de problemas “ill-structured" [3] en los que en general no hay una “solución” bien definida. El experto humano se enfrenta a ellos no solo mediante el conocimiento de los requisitos que han de cumplir las soluciones y de la forma de cumplirlos, sino también en gran parte mediante su creatividad, la cual le permite alcanzar distintos diseños válidos muy diferentes entre sí.

El proyecto arquitectónico participa de estas características de las tareas de diseño. Concretamente, en el proyecto de viviendas el arquitecto ha de tener en cuenta numerosas variables que afectan a la viabilidad de las soluciones. Aspectos como las relaciones entre distintos espacios y el tamaño de estos, entre otros muchos, han de considerarse. Por otra parte, la creatividad del arquitecto le permite obtener planos muy diferentes partiendo del mismo conjunto de restricciones y guías.

En este trabajo se ha abordado el problema de la generación automática y parcialmente dirigida de planos de viviendas. Las diversas variantes de este problema son de gran interés tanto en arquitectura como en diseño computacional, pues un sistema ayudante de diseño que genere y proponga planos automáticamente podría ser de gran ayuda tanto a los profesionales como a los estudiantes de arquitectura. Más concretamente, en este trabajo hemos implementado un sistema cuyo objetivo es la generación automática de un conjunto de planos de viviendas básicas unifamiliares que se ajusten a ciertas directrices o guías. Como mecanismo de generación, el sistema emplea las *gramáticas de formas* [4], [5].

**ISSN: 1988-3064(on-line)**

Existen varios trabajos en la literatura que abordan de manera parecida este problema [6]; pero, a diferencia de ellos, nuestra propuesta es partir de gramáticas de formas muy sencillas. La ejecución a ciegas de un generador basado en estas gramáticas dará lugar a un número muy elevado de diseños, la mayor parte de ellos inaceptables. Para evitar esto y guiar la generación, nuestra propuesta integra una fase previa de aprendizaje en la que se aplican técnicas de *aprendizaje por refuerzo* [7]. Los criterios y restricciones de diseño se formulan como recompensas aplicadas en esta fase. La ventaja de esta propuesta es que, gracias al uso de gramáticas de formas genéricas, los planos generados son bastante diferentes entre sí, lo que puede considerase como un indicio de "creatividad"; pero, por otra parte, el uso de aprendizaje por refuerzo permite tratar eficazmente con la explosión combinatoria de estados, guiando el proceso hacia aquellos diseños que cumplen las restricciones deseadas.

El resto del artículo se estructura como sigue: en la sección 2 se presentan los antecedentes necesarios para la comprensión del problema abordado y de la solución propuesta. En ella exponemos los conceptos fundamentales de las gramáticas de forma y de las técnicas de aprendizaje empleadas, y describimos brevemente el problema concreto considerado en este artículo. A continuación, en la sección 3 se describen la estructura y funcionamiento del sistema, tanto en los aspectos de generación como de aprendizaje. En la sección 4 se muestra su viabilidad ofreciendo los resultados obtenidos y especialmente algunos conjuntos de planos generados automáticamente. Por último, en la sección 5, se recopilan las conclusiones extraídas y se apuntan las líneas posibles de continuación y generalización de este trabajo.[1]

# 2 Antecedentes

## 2.1 Gramáticas de formas

Una gramática de formas [4] es un conjunto de reglas definidas sobre *formas*. Una forma es una colección de segmentos junto con una colección de puntos etiquetados, que pueden ser intersecciones entre segmentos o simplemente puntos arbitrarios distinguidos. El conjunto de reglas se aplica sobre una forma inicial (el *axioma*). Cada regla tiene la estructura $A \rightarrow B$, donde $A$ y $B$ son formas. El significado de una regla es: identificar la forma $A$ dentro de la forma que se está generando (empleando si es necesario una transformación compuesta de giro, simetría y cambio de escala) y sustituirla por la forma $B$ adecuadamente transformada. En la figura 1 se representan un axioma, una regla y una derivación, es decir, una sucesión de formas generadas aplicando la regla.

Las gramáticas de formas han sido usadas en la literatura para numerosas tareas de diseño como el proyecto arquitectónico [6], el diseño industrial [8] o la realidad virtual [9].

La potencia de las gramáticas de formas viene dada por su capacidad para producir una gran variedad de formas. En efecto, no solo es posible que varias reglas sean simultáneamente aplicables a una forma $F$, sino que pueden existir múltiples aplicaciones de una regla $A \rightarrow B$ a $F$, debido a que podemos identificar dentro de $F$ la forma $A$ del antecedente transformada de varias maneras. Para controlar esta variabilidad y dotar de un carácter más determinista a la gramática, se suelen añadir a las formas etiquetas cuyo único fin es controlar la generación. Otra alternativa es definir una función que para cada forma determina la regla (y la transformación) que se debe aplicar preferentemente [10].

Tradicionalmente, las gramáticas que han sido utilizadas en las tareas de diseño han seguido la primera de las alternativas mencionadas: el *experto* humano que las definía debía compilar en cada regla la información de control necesaria para generar precisamente las formas finales deseadas. Ello dificulta su creación, mantenimiento y modificación.

[1] Este trabajo está parcialmente financiado por el Plan Nacional de I+D+I, proyecto TIN2009-14179 (Gobierno de España, Ministerio de Ciencia e Innovación) y fue presentado originalmente en la XIV Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA 2011), organizada por el Grupo de Computación Inteligente de la Universidad de La Laguna y la Asociación Española de Inteligencia Artificial (AEPIA).

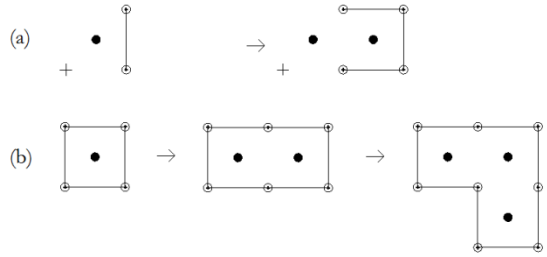



Figura 1. Una regla (a) y una derivación (b) de formas.

## 2.2 Aprendizaje por refuerzo

El aprendizaje por refuerzo [7] es una técnica para aprender una *política* de actuación ante un determinado problema (idealmente, la *política óptima*). Esta política selecciona la acción a tomar en cada momento, con el objetivo de maximizar la *recompensa* obtenida a largo plazo. Para cada par (*estado, acción*) se tiene un *valor*, de forma que, si nos encontramos en un estado *s*, la política queda definida por la regla: "elegir la acción *a* que produce un par (s, a) con valor máximo". Los valores de cada par se van modificando en un proceso iterativo dividido en *episodios*, a su vez divididos *en pasos*. En cada episodio se parte de un estado inicial y se intenta llegar a un estado final mediante la elección de acciones apropiadas (cada elección determina un paso del episodio). En determinados pasos, los valores obtenidos hasta el momento se van *reforzando* con nuevos valores. Así, tras un número de episodios de aprendizaje adecuado, los valores se habrán modificado tendiendo a definir una política próxima a la óptima.

En nuestro caso, cada forma producida por la aplicación de una gramática es un estado. Una acción sobre ese estado es un par (*regla, transformación*) tal que, tras aplicar la transformación, la regla resulte aplicable al estado.

Los métodos más directos de aprendizaje por refuerzo utilizan una tabla para almacenar los valores asignados a estados y acciones. En nuestro caso el tamaño del espacio de estados impide utilizar este método. Por tanto, es necesario emplear un método de generalización que tenga en cuenta únicamente determinados rasgos o características de los pares (*s, a*), y que permita el aprendizaje de una función en lugar de una tabla a modo de política.

El método concreto de aprendizaje por refuerzo que hemos empleado es el algoritmo $Q$ ($\lambda$) [11] con un aproximador lineal y rasgos binarios, tal como aparece en [7] (p. 213).

## 2.3 El proyecto de viviendas

El proyecto de viviendas unifamiliares es un proceso restringido por numerosas condiciones. La naturaleza de estos requisitos oscila entre aspectos como el área de cada espacio según el número de habitantes, relaciones de proximidad entre espacios y, por supuesto, restricciones más detalladas dependiendo del tipo de habitación que se esté diseñando. En este trabajo hemos partido de la propuesta elaborada por el estudio Montaner y Muxí [12] para la Junta de Andalucía. En ella, partiendo de consideraciones funcionales, se detallan las condiciones y criterios que ha de cumplir el diseño de una *vivienda básica* según el número de personas que van a habitarla. Por vivienda básica se entiende una vivienda que, además de cumplir unas condiciones mínimas de habitabilidad, también ofrece cierta adaptabilidad, haciendo posible una evolución de su composición si el número de habitantes se ve incrementado. Nuestro sistema produce diseños de viviendas básicas para dos personas.

En la propuesta se establecen varios tipos de ámbitos (figura 2) que han de estar presentes en una vivienda básica, de entre los cuales hemos considerado los tres tipos principales: 1) *ámbitos especializados* (necesitan instalaciones específicas para su funcionamiento), 2) *ámbitos no especializados* (no necesitan instalaciones específicas, su uso queda determinado por los usuarios: comedor, salón, dormitorio...) y 3) *ámbitos complementarios* (espacios de distribución).

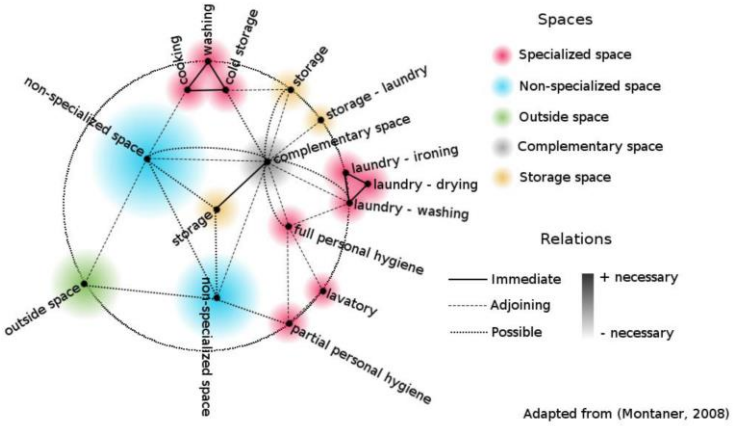



Figura 2. Elementos y relaciones en una vivienda (adaptado de [12]).

En la figura 2 se representan gráficamente las relaciones de proximidad entre los distintos ámbitos. En la tabla 1 se recogen las condiciones impuestas para cada uno de los ámbitos y para la superficie total de la vivienda en el caso de dos habitantes. Nuestro sistema también considera dos condiciones adicionales en la tabla 2 que no están recogidas en la propuesta [12], pero que hemos elegido para acotar en cierta forma la investigación: 1) el plano de la vivienda ha de maximizar la fórmula de la compacidad, es decir, se primarán aquellos planos con un índice de compacidad[2] alto; 2) la entrada a la vivienda ha de estar a una distancia del distribuidor y de la cocina menor de dos y de cuatro metros, respectivamente.

Los espacios especializados, es decir, la cocina y el baño, se configuran en base a la agregación de pequeños módulos predefinidos. En el caso de la cocina, estos módulos son de 60x60 cm, mientras que en caso del baño, son de 90x180 cm. El uso de estos módulos permite establecer las condiciones de área de los espacios especializados en base al número de módulos existentes en la vivienda.

# 3 Un sistema para generación de viviendas básicas

Según la propuesta descrita, hemos implementado un sistema de generación de viviendas básicas sujetas a las condiciones de la tabla 1. Para la programación del sistema se ha utilizado el lenguaje Ruby y para la visualización de los resultados hemos hecho uso de Google Sketchup, una herramienta de modelado 3D que permite la incorporación de *plugins* descritos en Ruby. Para la ejecución de las gramáticas de formas se han desarrollado intérpretes *ad-hoc* para cada una de ellas.

| | |
|---|---|
| Condiciones globales | CM-1: La superficie total ha de ser como mínimo de 46 $m^2$ |
| Condiciones para la cocina | CM-2: El espacio lineal mínimo sería de 6 módulos de 60x60 cm<br>CM-3: Distancia mínima entre módulos y pared: 1,10 m<br>CM-4: Distancia mínima entre módulos: 1,10 m |
| Condiciones para el baño | CM-5: Como mínimo habrá dos módulos de 90x180 cm |
| Condiciones para ámbitos no especializados | CM-6: El área de cada ámbito no especializado ha de ser mayor a 9 $m^2$<br>CM-7: En cada ámbito no especializado ha de poder inscribirse un círculo de como mínimo 2,80 metros de diámetro |
| Condiciones para ámbitos complementarios | CM-8: Ha de existir un espacio de apoyo que permita la circulación entre ámbitos. |

Tabla 1. Condiciones para una vivienda para dos personas (adaptado de [12]).

[2] Índice de compacidad = área/ perímetro$^2$

| | |
|---|---|
| Condiciones globales | CA-1: el perímetro de la vivienda debe ser compacto, es decir, ha de maximizar el índice de compacidad (Ic= Área/Perímetro $^2$). |
| Condiciones para la entrada | CA-2: la distancia desde la entrada al distribuidor y a la cocina ha de ser menor de 2 y 4 metros, respectivamente. |

Tabla 2. Condiciones adicionales a las recogidas en la propuesta [12].

## 3.1 Procesos de generación

Para la síntesis de los planos se han utilizado gramáticas de formas muy simples y genéricas, que no integran conocimiento de control. Esto quiere decir que no se utilizan marcas para guiar la ejecución de las reglas.

El proceso de generación de planos está dividido en 12 fases. Excepto las fases10, 11 y 12, todas cuentan con una única regla. Cada regla se ejecuta hasta quesea posible aplicarla o bien hasta alcanzar un estado final. Dependiendo de cada fase, el test de estado final es diferente. Adicionalmente se establece un número máximo de pasos en las derivaciones.

Como entrada a la primera fase, partimos de un axioma sencillo (un módulo de 1 m$^2$ convenientemente etiquetado). Cada fase aplica su gramática de formas sobre el diseño que toma como entrada de la fase anterior. En la figura 3 se ilustran las gramáticas utilizadas en las fases 1-9.

La descripción de cada fase es la siguiente:

- Fase 1: Generación de un contorno. Test de estado final: el contorno encierra 46 m$^2$.
- Fase 2: Eliminación de paredes que no delimitan el contorno. La ejecución se realiza hasta que ya no es posible aplicar la regla.
- Fase 3: Marcación del distribuidor. Test de estado final: existe una marca de distribuidor.
- Fase 4: Ubicación del primer módulo de la cocina. Test de estado final: el primer módulo de la cocina está convenientemente ubicado.
- Fase 5: Ubicación del resto de módulos de la cocina. Test de estado final: hay seis módulos de la cocina convenientemente ubicados.
- Fase 6: Ubicación del primer módulo del baño. Test de estado final: el primer módulo del baño está convenientemente ubicado.
- Fase 7: Ubicación del resto de módulos del baño. Test de estado final: hay dos módulos del baño convenientemente ubicados.
- Fase 8: Marcación de los ámbitos no especializados: hay dos ámbitos no especializados convenientemente ubicados.
- Fase 9: Ubicación de la entrada a la vivienda. Test de estado final: la entrada a la vivienda está convenientemente ubicada.
- Fase 10: Tabicado de la vivienda. La ejecución se realiza hasta que ya no es posible aplicar las reglas.
- Fase 11: Adición de elementos especializados (para el baño y la cocina: inodoro, lavamanos, pie de ducha, fregadero y vitro-cerámica). Test de estado final: existencia de los elementos.
- Fase 12: Eliminación de paredes internas inservibles. La ejecución se realiza hasta que ya no es posible aplicar las reglas.

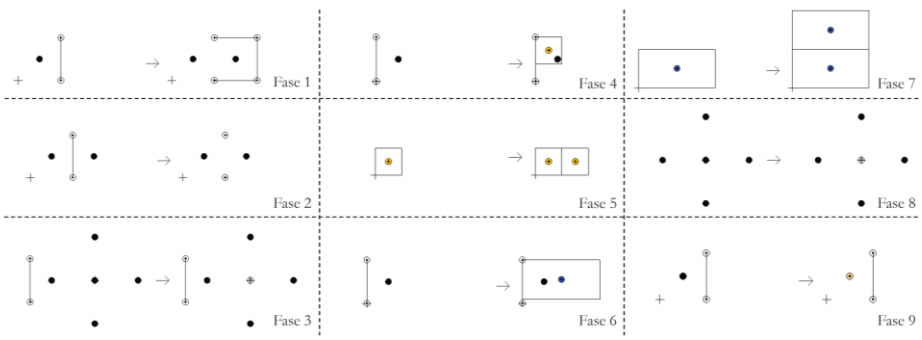



Figura 3. Reglas utilizadas en las fases 1-9.

Para la presentación final de los planos se realizan dos operaciones adicionales: a) borrado de los puntos manejados por las gramáticas para identificar interiores/exteriores, diversos tipos de espacios, etc.; y b) adición de texto significativo que identifique cada habitación.

Siguiendo este proceso se han generado 20 diseños distintos; en la figura 4 se muestran dos de ellos. El tiempo de generación de cada diseño oscila en torno a un minuto. Ninguno de los planos generados cumple los requisitos establecidos en la tabla 1.

## 3.2 Procesos de aprendizaje

Como podemos apreciar en la figura 4, las fases de generación por sí solas no producen soluciones factibles. Para conseguir diseños válidos, hemos incorporado políticas en distintas fases de la generación. Así, en cada momento, se decide qué aplicación de las reglas es la mejor con vistas a producir diseños válidos.

Estas políticas se generan mediante procesos de aprendizaje por refuerzo. Las fases que incluyen políticas son las fases 1, 4, 5, 6, 7, 8 y 9. Por su naturaleza, las fases 2 y 3 pueden ser ejecutadas aleatoriamente sin que esto afecte a la calidad de los diseños. En concreto, la fase 2 simplemente se encarga de eliminar paredes residuales. La fase 3 ubica el distribuidor ejecutando la regla correspondiente de manera aleatoria, de modo que las políticas de las fases posteriores tendrán en cuenta esta ubicación para colocar el resto de elementos de la vivienda.

Para cada fase, tal y como se describe en la Sección 2.2, el algoritmo de aprendizaje trabaja con valores *Q(s,a)* asociados a cada par (*estado, acción*). En este contexto, el estado es un diseño concreto, y la acción es un par (*regla, transformación*) a aplicar sobre tal diseño.

En cada episodio de aprendizaje, el algoritmo comienza a aplicar reglas sobre el axioma hasta llegar a un estado final. Durante el aprendizaje, la elección del par (*regla, transformación*) a aplicar se realizará o bien explotando la política aprendida hasta el momento (es decir, suponiendo que estamos en el estado *s*, seleccionando aquella acción *a* tal que el valor *Q(s,a)* es máximo), o bien explorando de manera aleatoria (es decir, eligiendo una acción al azar, por si acaso tal acción lleva a un resultado mejor que el almacenado hasta el momento). En determinados pasos, los valores obtenidos hasta el momento se van reforzando con las recompensas obtenidas. Así, tras un número adecuado de episodios de aprendizaje, los valores *Q(s,a)* se habrán modificado tendiendo a definir una política próxima a la óptima.

Tal y como describimos en la Sección 2.2, la gran cantidad de estados posibles hace inviable la utilización de una tabla para almacenar los valores *Q(s,a)*. En su lugar, utilizaremos una función lineal sobre determinados rasgos o características de los estados. Para más detalles acerca del funcionamiento del algoritmo implementado, llamado *Q* (λ), consultar el trabajo de Sutton y Barto [7] (p. 213). A continuación se detallan los rasgos utilizados en cada fase:

- Fase 1 (generación de un contorno). Recompensa: área/perímetro$^2$, definida en los estados finales. Esta fase consta de 4 rasgos continuos[3] normalizados entre 0 y 1:
  - R1: nº de módulos de 1 m$^2$ con 1 vecino.
  - R2, R3, R4: de manera análoga a R1, con 2, 3 y 4 vecinos respectivamente.

  Tiempo de aprendizaje: 4728,42 segundos.
- Fase 4 (ubicación del primer módulo de la cocina). Recompensa: R1 + R2 siendo:
  - R1: 1 sii el módulo está a más de 2 metros y menos de 4 del distribuidor.
  - R2: 1 sii la distancia entre el módulo y las paredes es mayor a 1,1 metros.

  Tiempo de aprendizaje: 76,57 segundos.
- Fase 5 (ubicación del resto de módulos de la cocina). Recompensa: 3 * R1 + R2 + R3 + R4 + R5 + R6, siendo:
  - R1: 1 sii todos los módulos se encuentran en el interior del contorno.
  - R2: 1 sii todos los módulos son accesibles.
  - R3: 1 sii la distancia entre los módulos y las paredes es mayor a 1,1 m.
  - R4: 1 sii los módulos están separados entre sí al menos 1,1 m.
  - R5: 1 sii los módulos están a más de 1,2 m y menos de 5 del distribuidor.
  - R6: 1 sii hay al menos 6 módulos.

  Tiempo de aprendizaje: 270,758 segundos.
- Fase 6 (ubicación del primer módulo del baño). La recompensa y rasgos son similares a los de la fase 4. Tiempo de aprendizaje: 126,75 segundos.

[3] En esta fase, el algoritmo de aprendizaje utilizado ha sido levemente modificado para tratar con rasgos continuos.

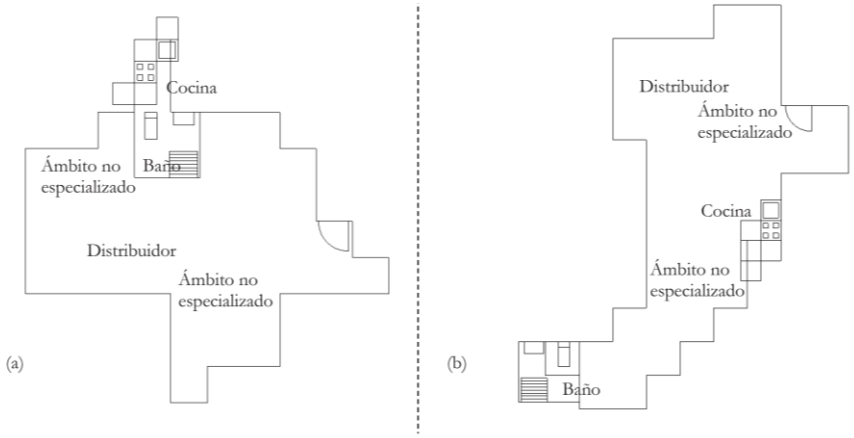



Figura 4. Dos planos generados aleatoriamente.

- Fase 7 (ubicación del resto de módulos del baño). La recompensa y rasgos son similares a los de la fase 5. Tiempo de aprendizaje: 481,215 segundos.
- Fase 8 (marcación de los ámbitos no especializados). Recompensa: 3 * R1 + R2 + R3 + R4 + R5 + R6, siendo los más significativos:
  - R1: 1 sii en cada marca se puede centrar un círculo de 3 m de diámetro sin solaparse con paredes, esquinas o módulos.
  - R2: 1 sii hay una marca a menos de 4,5 m del baño.
  - R3: 1 sii hay una marca a menos de 4,5 m de la cocina.
  - R5: 1 sii la distancia de cada marca al distribuidor es mayor de 2 m y menor de 6.

  Tiempo de aprendizaje: 130,198 segundos.
- Fase 9 (ubicación de la entrada). Recompensa: R1 + R2 + R3 + R4, siendo los más significativos:
  - R2: 1 sii la entrada está a menos de 2 m de la cocina.
  - R4: 1 sii la entrada está a menos de 4 m del distribuidor.

  Tiempo de aprendizaje: 53,611 segundos.

Excepto en el primer proceso, en el cual el estado inicial es siempre el mismo (un módulo de 1 $m^2$), en el resto de los procesos los estados iniciales pueden ser muy distintos. Por ejemplo, en el caso del proceso correspondiente a la fase de generación cuatro, el cual aprende a añadir el primer módulo de la cocina sobre contornos compactos con un distribuidor marcado, existen múltiples estados iniciales que aglutinen estos dos aspectos. Para que el aprendizaje tenga sentido, no nos podemos limitar a aprender a partir de un estado inicial único. Es necesario el uso de un conjunto de posibles estados iniciales, de forma que los episodios del aprendizaje se realicen a partir de estados iniciales distintos. Estos estados iniciales se generan utilizando los procesos de generación correspondientes a las fases anteriores, utilizando las políticas generadas hasta el momento.

Una vez aprendidas todas las políticas, podemos generar planos automáticamente lanzando los procesos de generación, guiados cuando proceda por dichas políticas. A continuación se muestran los resultados obtenidos.

# 4 Resultados

Para mostrar la capacidad de nuestro sistema, 100 planos fueron producidos mediante los procesos de generación guiados por políticas. De ellos se muestran algunos en la figura 5. La ejecución tardó unos 100 minutos aproximadamente. Todos los planos generados son diferentes y cumplen las condiciones consideradas.


Figura 5. Algunos diseños generados.

# 5 Conclusiones

En este artículo hemos presentado una propuesta para complementar la potencia generativa de las gramáticas de formas con técnicas de aprendizaje por refuerzo. La potencia generativa de las gramáticas de formas permite una gran diversidad de diseños, mientras que el aprendizaje por refuerzo guía el proceso de generación hacia aquellas soluciones que satisfacen las condiciones de diseño.

En concreto, hemos aplicado esta propuesta en la implementación de un sistema que genera planos de viviendas sencillas que satisfacen las restricciones de una determinada guía de diseño elaborada para la Junta de Andalucía. El sistema consta de reglas muy simples. El aprendizaje se basa en recompensas y rasgos en su mayoría definidos de forma directa a partir de lo prescrito en dicha guía. Los procesos de aprendizaje han requerido tiempos del orden de minutos.

Los planos generados por el sistema representan diseños claramente superiores a los generados mediante las gramáticas sin aprendizaje. Además, los diseños presentan una amplia variabilidad y la mayoría de ellos son soluciones razonables al problema planteado.

La continuación natural del trabajo aquí expuesto es la aplicación de esta misma propuesta (gramáticas de formas simples y aprendizaje con refuerzo) a casos más complejos de diseño arquitectónico. Igualmente, sería interesante establecer una comparativa del rendimiento de esta técnica con técnicas meta-heurísticas como la computación evolutiva.

# Referencias

[1] H.A. Simon. *The sciences of the artificial*. MIT Press, Cambridge, MA, 1968.

[2] C.M. Eastman. Cognitive processes and ill-defined problems: a case study from design. In *IJCAI'69*, pages 669-690, 1969.

[3] H.A. Simon. The structure of ill structured problems. *Artificial Intelligence*, 4: 181-201, 1973. doi:10.1016/00043702(73)90011-8.

[4] G. Stiny. Introduction to shape and shape grammars. *Environment and Planning B,* 7: 343-351, 1980. doi: 10.1068/b070343.

[5] G. Stiny. *Shape. Talking about seeing and doing*. MIT Press, Cambridge, MA, 2006.

[6] J.P. Duarte. A discursive grammar for customizing mass housing: the case of Siza's houses at Malagueira. *Automation in Construction*, 14: 265-275, 2005. doi: 10.1016/j.autcon.2004.07.013

[7] R.S. Sutton, A. G. Barto. *Reinforcement learning: an introduction*. MIT Press, Cambridge, MA, 1998.

[8] H.C. Lee, M. X. Tang. Evolving product form design using parametric shape grammars integrated with genetic programming. *Artificial Intelligence in Engineering Design, Analysis and Manufacturing*, 23: 131-158, 2009. doi: 10.1017/S0890060409000031

[9] P. Muller, P. Wonka, S. Haegler, A. Ulmer, L. V. Gool. Procedural modeling of buildings. *ACM Transactions on Graphics,* 25(3): 614-623, 2006. doi: 10.1145/1141911.1141931

[10] T.W. Knight. Shape grammars: six types. *Environment and Planning B*, 26: 15-31, 1999. doi: 10.1068/b260015

[11] C.J. Watkins. *Learning from delayed rewards*. PhD thesis, University of Cambridge, 1989.

[12] Montaner Muxí arquitectes. *Propuesta de nueva normativa de viviendas*. Technical report, Dirección general de ordenación del territorio, Junta de Andalucía, 2008.