Un marco de evaluación para LLMs basado en la alineación texto-vídeo de código renderizable

Rodrigo Pascual-García[1 [0009−0005−8791−4159]], Gadea Lucas-Pérez[1 [0009−0009−2258−8507]], David Martínez-Acha[1 [0009−0005−4214−2845]], Lydia Blanco-Ruiz[1 [0009−0000−2220−1856]], José Francisco Díez-Pastor[1 [0000−0001−5013−7505]], Pedro Latorre-Carmona[1 [0000−0001−6984−5173]], and José Manuel Aroca-Fernández[1 [0000−0002−0312−5955]]

Departamento de Ingeniería Informática, Universidad de Burgos, España
rodrigopg@ubu.es

Resumen Los modelos de lenguaje de gran tamaño (LLMs) se evalúan habitualmente con pruebas textuales o de programación cuyos resultados tienden a saturarse conforme los modelos mejoran. Este trabajo propone un *benchmark* dinámico para evaluar una capacidad complementaria: transformar una descripción visual en código ejecutable que, al renderizarse, produzca un vídeo alineado semánticamente con la escena solicitada. La evaluación no se aplica al código como texto, sino al artefacto producido por su ejecución: cada vídeo renderizado se compara con el *prompt* original mediante *embeddings* multimodales texto-vídeo. La métrica principal calcula la similitud coseno entre ambos *embeddings* y ordena los vídeos dentro de cada escena. Para validar este *ranking* automático, se compara con *rankings* humanos mediante correlación de Spearman y con dos referencias comparativas: una evaluación visual estricta basada en GPT-4o sobre tres fotogramas del vídeo y SAVi (*Semantic-Animation Video*), una métrica híbrida propuesta en este trabajo para incorporar granularidad semántica y coherencia temporal. Los resultados preliminares muestran que la métrica basada en *embeddings* obtiene la mayor alineación con el juicio humano ($\rho = 0.795$), por encima de SAVi ($\rho = 0.766$) y de la métrica visual basada en GPT-4o ($\rho = 0.564$). Además, el *benchmark* selecciona modelos de forma periódica a partir de *rankings* públicos de OpenRouter, lo que permite mantener una comparación actualizable de LLMs en una tarea todavía no saturada.

Keywords: Modelos de lenguaje de gran tamaño · Alineación texto-vídeo · *Embeddings* multimodales · Evaluación de animaciones · Evaluación de LLMs

1. Introducción

La evaluación de modelos de lenguaje de gran tamaño se enfrenta a una tensión creciente entre utilidad práctica y saturación experimental. Muchos *benchmarks* textuales o de programación han permitido medir avances notables, pero

en tareas consolidadas, los modelos de frontera se aproximan a puntuaciones muy altas y cada vez resulta más difícil discriminar capacidades finas entre sistemas [6]. Esta situación motiva la exploración de tareas evaluativas en las que el modelo deba combinar comprensión semántica, razonamiento espacial, composición visual y generación de código ejecutable.

Este trabajo propone evaluar LLMs mediante una tarea indirecta pero observable: dado un *prompt* que describe una escena visual dinámica, el modelo debe generar código que pueda ejecutarse y renderizarse como vídeo. Luego, la evaluación no se realiza sobre la apariencia superficial del código ni sobre una explicación textual, sino sobre el resultado visual producido por su ejecución. De este modo, un modelo solo obtiene una buena posición si el programa generado materializa en vídeo los elementos relevantes de la escena.

La hipótesis principal es que la alineación entre el *prompt* y el vídeo puede estimarse proyectando ambos objetos en un espacio de *embeddings* multimodal compartido. Esta idea conecta con el aprendizaje de representaciones visuales guiadas por lenguaje natural [10] y con modelos de *embeddings* recientes capaces de representar distintas modalidades en un mismo espacio vectorial.

En esta versión experimental se usa Gemini Embedding 2 Preview [3], un modelo multimodal que proyecta texto, imágenes, vídeo, audio y documentos en un espacio común y que, por defecto, devuelve vectores de 3 072 dimensiones. Si un vídeo recrea correctamente una escena, su representación debería conservar una parte sustancial de la semántica contenida en la descripción textual. A partir de la similitud entre ambas representaciones se obtiene un *ranking* automático de los vídeos generados para una misma escena.

El *benchmark* se diseña como un sistema dinámico. Los modelos candidatos se seleccionan a partir de *rankings* de popularidad públicos de OpenRouter [9] y se invocan mediante su API, de forma que la evaluación pueda actualizarse conforme cambian los modelos disponibles o su relevancia práctica. Esta característica permite pasar de un experimento cerrado a un procedimiento repetible y extensible de comparación de modelos.

Las contribuciones principales son: (i) una tarea de evaluación para LLMs basada en generar código renderizable desde descripciones visuales; (ii) una métrica automática de alineación texto-vídeo basada en *embeddings* multimodales; (iii) una validación preliminar frente a *rankings* humanos mediante correlación de Spearman; y (iv) una aplicación del método a un *benchmark* dinámico de modelos seleccionados mediante *rankings* públicos de OpenRouter.

2. Trabajo relacionado

2.1. Evaluación de LLMs y saturación de *benchmarks*

La evaluación sistemática de LLMs ha evolucionado desde pruebas centradas en tareas lingüísticas hasta marcos holísticos que integran razonamiento, robustez, equidad y eficiencia [6]. Sin embargo, los *benchmarks* estáticos pueden perder capacidad discriminativa cuando los modelos se optimizan explícita

o implícitamente para ellos. En este contexto, las tareas que requieren producir artefactos ejecutables ofrecen una alternativa: el resultado puede inspeccionarse fuera del dominio textual y medirse mediante propiedades observables.

2.2. Evaluación automática y jueces basados en modelos

Una estrategia frecuente para evaluar salidas generativas consiste en usar otro modelo como juez, especialmente en tareas abiertas [12]. Aunque estos enfoques son flexibles, introducen variabilidad, dependencia del *prompt* de evaluación y posibles sesgos del juez. En este caso, la generación del código por parte del LLM no es determinista, pero la puntuación automática sí es reproducible para un par fijo *prompt*-vídeo bajo la misma configuración de extracción, modelo de *embeddings* y fórmula de similitud.

2.3. Evaluación de vídeo generado

La evaluación de modelos generativos de vídeo ha recibido atención reciente. VBench [4] propone una suite amplia para medir dimensiones como calidad visual, consistencia temporal y alineación con el *prompt*. VideoPhy [1] estudia la comprensión física en modelos texto-vídeo y muestra que incluso modelos avanzados presentan limitaciones en adherencia semántica y física. Este trabajo se diferencia en que el vídeo no es generado directamente por un modelo texto-vídeo: se obtiene al ejecutar código producido por un LLM. Por tanto, la tarea evalúa una cadena compuesta de comprensión textual, generación de código, ejecución y resultado visual.

2.4. Evaluación ordinal y comparación relativa

En tareas generativas abiertas, las puntuaciones absolutas pueden depender fuertemente del *prompt*, de la escena y del rango natural de la métrica. Por ello, este trabajo adopta una formulación ordinal: para cada escena se comparan los modelos entre sí y la concordancia entre *rankings* se mide con coeficientes de correlación de rango [5,11], en coherencia con una validación humana basada en ordenar vídeos dentro de una misma escena.

3. Metodología y diseño del *benchmark*

3.1. Definición de la tarea

Cada instancia del *benchmark* contiene una descripción textual de una escena visual dinámica. Dado un *prompt* t_j, cada modelo m_i recibe una instrucción de sistema fija y debe devolver únicamente código ejecutable usando JavaScript, en concreto la librería gráfica `P5.js`[1]. Ese código se ejecuta en un entorno controlado y se renderiza como un vídeo v_{ij}. Se considera vídeo válido aquel cuyo código

[1] `https://p5js.org/`

generado no produce errores de ejecución y permite obtener un vídeo evaluable mediante las métricas automáticas. La unidad evaluada no es el texto del código, sino el vídeo producido por su ejecución.

La tarea puede verse como una traducción de descripción visual a programa renderizable:

$$m_i(t_j) \rightarrow c_{ij} \rightarrow v_{ij},$$

donde c_{ij} es el código generado por el modelo m_i para la escena j, y v_{ij} es el vídeo renderizado. La Fig. 1 resume el flujo completo: selección de modelos, generación de código, renderizado, cálculo de métricas automáticas y validación con *rankings* humanos.



Figura 1. Flujo general del *benchmark*. A partir de una descripción visual se seleccionan modelos candidatos, cada LLM genera código renderizable, el código se ejecuta para obtener un vídeo y el resultado se evalúa mediante *embeddings* multimodales.

3.2. *Prompt* de generación

El *system prompt* fija el formato y las restricciones de la tarea. La versión utilizada solicita al modelo actuar como desarrollador experto de `P5.js` y animador procedural, generar escenas reconocibles, visualmente ricas y físicamente fluidas, mantener un lienzo de 1024×1024, evitar recursos externos e interactividad, y devolver únicamente código válido. Además, explicita criterios de realismo semántico, riqueza visual, cinemática fluida y coherencia temporal. La instrucción completa utilizada en los experimentos se recoge en el Código 1.1.

Los *prompts* de usuario describen escenas concretas. Por ejemplo: *"A traditional windmill on a green hill with large wooden sails covered in white cloth rotating at a steady pace driven by an invisible breeze, casting moving shadows on the grass below as slow clouds drift in the background sky."*

3.3. Selección dinámica de modelos

La selección de modelos se basa en *rankings* públicos de OpenRouter [9]. El sistema consulta los modelos más populares del momento, normaliza sus identificadores y los invoca mediante la API de OpenRouter. Esta decisión evita fijar manualmente una lista cerrada de modelos y permite que el *benchmark* evolucione con el ecosistema real de LLMs. En la práctica, cada escena se evalúa con un subconjunto de modelos disponibles y relevantes en el momento de ejecución.

La experimentación no se realiza de forma manual: un flujo automatizado en n8n orquesta la ejecución, mientras que microservicios en Node/Express realizan la generación y el renderizado. El código se ejecuta en Chromium mediante

Código 1.1. *System prompt* completo utilizado en la generación.

```
You are an expert P5.js developer and procedural animator.
Your goal is to create highly recognizable, visually rich, and physically fluid animations
     that strictly adhere to the given prompt.

Hash out your plan in detailed comments before writing the code.

When given a prompt:
1. Semantic Realism: The subject MUST be clearly recognizable and central to the canvas. Do
     not create abstract interpretations. Use custom geometry, advanced path drawing,
     gradients, and detailed structures to create high-fidelity, realistic representations.
2. Visual Richness: Make the scene aesthetically pleasing by using layering, shadows,
     highlights, and rich color palettes. Avoid flat, basic primitives--add depth and detail
     .
3. Fluid Kinematics: The animation will be recorded for exactly 15 seconds. Movement must be
     continuous, smooth, and physically logical. Utilize advanced time-based math, organic
     noise, or wave functions to drive the animation.
4. Temporal Coherence: AVOID abrupt cuts, flashing, or sudden teleportations, as these ruin
     kinematic fluidity. If an object moves across the screen, implement seamless spatial
     wrap-around or cyclic easing. The main subject must NEVER permanently leave the canvas
     or stop moving.

Constraints:
- Canvas: 1024x1024 (square).
- No external assets (images/fonts).
- No interactivity (mouse/keyboard).

Return ONLY the valid P5.js code.
```

Puppeteer, el vídeo se genera con FFmpeg y cada evaluación almacena modelo, proveedor, tiempo de inferencia, *tokens*, coste, código generado, vídeo renderizado y puntuaciones automáticas. Este diseño permite repetir el proceso mediante activadores periódicos, por ejemplo diarios o semanales.

3.4. Métricas automáticas

Alineación texto-vídeo mediante *embeddings*. La métrica principal compara el *embedding* del *prompt* con el *embedding* del vídeo renderizado. En la implementación experimental se obtienen *embeddings* con Gemini Embedding 2 Preview [3]: el *prompt* se envía como texto y cada vídeo se envía como fichero con su tipo MIME. Sea $E_t(t_j)$ la representación vectorial del texto de la escena y $E_v(v_{ij})$ la representación del vídeo generado por el modelo m_i. La puntuación se calcula como:

$$s_{\text{emb}}(i,j) = 100 \cdot \max\left(0, \cos(E_t(t_j), E_v(v_{ij}))\right).$$

El objetivo de esta puntuación no es interpretar la similitud absoluta como una nota universal, sino estimar cuánta semántica del *prompt* textual puede recuperarse del vídeo renderizado. En la práctica, los rangos de similitud coseno varían entre escenas y las diferencias entre vídeos pueden ser pequeñas. Por ello, para cada escena j se define el *ranking* automático π^{j}_{emb} como la permutación de los modelos evaluados en dicha escena que ordena sus vídeos de mayor a menor puntuación $s_{\text{emb}}(i,j)$. Para el agregado se usa la posición relativa normalizada $q_{ij} = (n_j - r_{ij} + 1)/n_j$, donde $r_{ij} = 1$ indica el primer puesto del modelo m_i en

la escena j y n_j el número de vídeos válidos de esa escena. Esta lectura ordinal es la que se usa tanto para la correlación con humanos como para el *benchmark* agregado. La Fig. 2 ilustra esta comparación en el espacio multimodal.

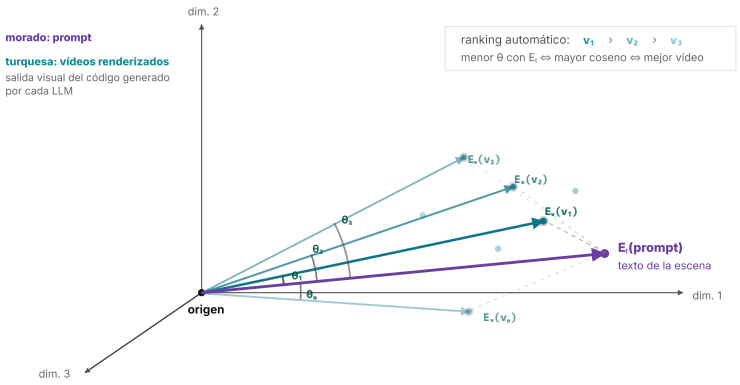


Figura 2. Intuición de la métrica de alineación texto-vídeo. El *prompt* de usuario y los vídeos renderizados se proyectan en un mismo espacio vectorial multimodal, los vídeos cuyo *embedding* queda más alineado con el *embedding* textual reciben mayor puntuación s_{emb}.

Métricas comparativas. Además de la métrica principal, se calculan dos métricas automáticas adicionales como referencias comparativas. La primera, denominada en adelante *Visión*, es una evaluación visual basada en GPT-4o [7]. Para cada vídeo se construye una imagen compuesta por tres fotogramas, al 25 %, 50 % y 75 % de la animación. El modelo recibe esa imagen junto con el *prompt* original y una rúbrica estricta que le obliga a describir objetivamente los tres fotogramas, detectar fallos graves y puntuar por separado fidelidad al *prompt*, complejidad visual, movimiento y estética. La puntuación final se calcula posteriormente mediante una combinación fija: 40 % fidelidad, 20 % complejidad, 20 % movimiento y 20 % estética. Esta métrica es útil como comparación externa, pero no está diseñada específicamente para medir alineación texto-vídeo en un espacio latente común.

La segunda métrica es SAVi (*Semantic-Animation Video*), propuesta en este trabajo como una métrica híbrida para comprobar si la métrica de *embeddings* podía mejorarse incorporando información más granular y temporal. SAVi combina tres componentes. El primero, S_{global}, repite la alineación global texto-vídeo mediante *embeddings*. El segundo, S_{fine}, descompone el *prompt* en átomos semánticos y compara cada átomo con fotogramas muestreados del vídeo. El tercero, S_{temp}, estima la dinámica mediante flujo óptico de Farneback [2] y aplica una penalización basada en variaciones bruscas del movimiento. La puntuación final es una media ponderada:

$$s_{\text{SAVi}} = w_g S_{\text{global}} + w_f S_{\text{fine}} + w_t S_{\text{temp}},$$

con pesos normalizados. En esta versión se usaron pesos iniciales estimados ($w_g = 0.6$, $w_f = 0.2$, $w_t = 0.2$), pendientes de ajustar empíricamente en trabajo futuro. La hipótesis era que los *embeddings* globales podían reconocer bien los objetos de la escena pero ignorar parte de la fluidez o riqueza temporal. Sin embargo, los resultados preliminares muestran que SAVi no supera a la métrica pura de *embeddings* en correlación con el *ranking* humano.

Una vez calculadas las puntuaciones, cada escena se trata de manera independiente: los vídeos generados para un mismo *prompt* se ordenan por s_{emb} y se conservan también las puntuaciones SAVi y Visión para comparar el comportamiento relativo de las métricas. La Fig. 3 muestra ejemplos de esta ordenación intra-escena, donde n indica vídeos válidos por escena.

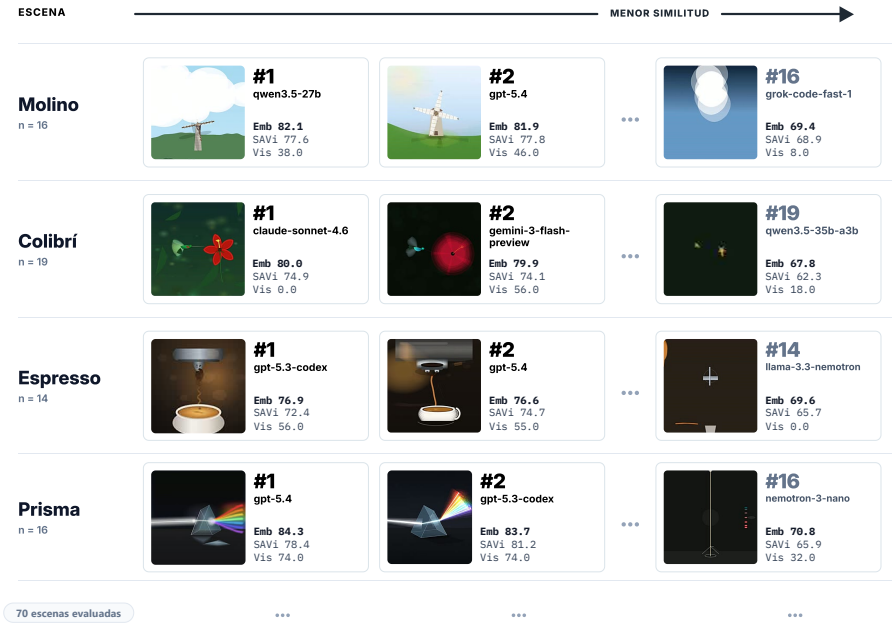


Figura 3. Ejemplos de *rankings* por escena con la métrica de *embeddings*. Cada fila muestra los dos vídeos mejor alineados y el peor candidato; SAVi y Visión se incluyen como métricas comparativas.

3.5. Evaluación humana

La validación humana permite comprobar si el *ranking* automático captura preferencias similares a las de evaluadores externos. Para ello se diseñó una interfaz en la que cada participante ordena los vídeos de una misma escena según el grado en que recrean la descripción textual. La tarea humana es relativa, no de puntuación absoluta: se pide arrastrar los vídeos para formar un *ranking* del mejor al peor. Esta decisión reduce la ambigüedad de asignar notas numéricas y se alinea con el objetivo experimental, que es comparar órdenes.

Para cada escena j y evaluador h, se obtiene un *ranking* humano π_h^j. La concordancia entre una métrica automática y el juicio humano se mide mediante la correlación de Spearman, $\rho(\pi_{\text{auto}}^j, \pi_h^j)$, y posteriormente se agregan las correlaciones sobre evaluadores y escenas. Además, el acuerdo inter-evaluador se estimó mediante el coeficiente W de Kendall, $W = 12S/(m^2(n^3 - n))$, donde m es el número de evaluadores y n el número de vídeos ordenados en la escena. La Fig. 4 resume este protocolo.

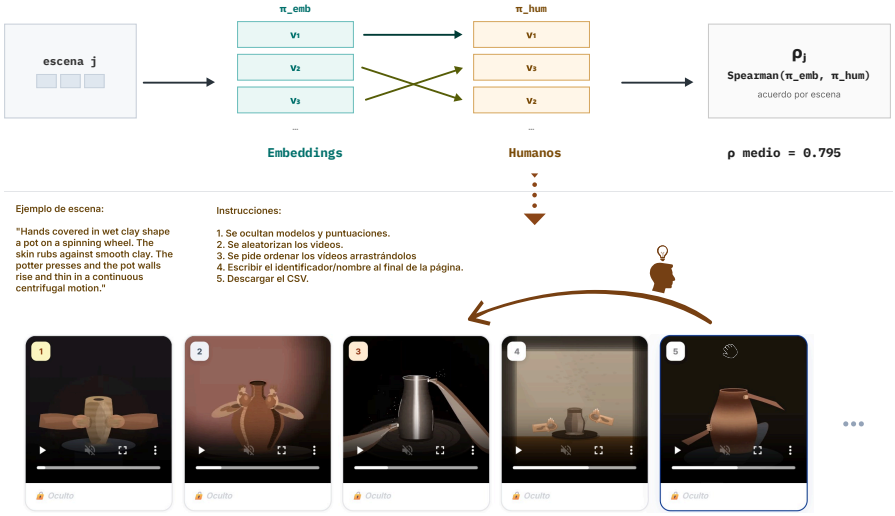


Figura 4. Protocolo de validación humana. Los participantes ordenan los vídeos de una misma escena sin ver el nombre del modelo ni las puntuaciones automáticas. El *ranking* humano resultante se compara con el *ranking* generado por la métrica de *embeddings* mediante correlación de Spearman por escena.

4. Experimentos

4.1. Conjunto de escenas y modelos

El sistema se ha ejecutado sobre escenas visuales variadas, incluyendo mecanismos, fenómenos ópticos, objetos animados y escenas naturales. En los experimentos automáticos acumulados se consideraron 70 escenas únicas, 40 modelos únicos, 791 pares modelo-escena por métrica y 2 373 puntuaciones automáticas en total. Para aumentar la comparabilidad de la validación humana, 12 participantes evaluaron el mismo subconjunto de 10 escenas, generando 120 evaluaciones individuales de *ranking* bajo las mismas condiciones experimentales.

Cada evaluación registra el modelo, proveedor, tiempo de inferencia, *tokens* de entrada y salida, coste aproximado, código generado, vídeo renderizado y puntuaciones automáticas. Los hiperparámetros de generación, como temperatura o límite de *tokens*, se mantuvieron según la configuración disponible para

cada proveedor o modelo, omitiendo parámetros no soportados por la API correspondiente. Esta trazabilidad permite analizar no solo la calidad visual, sino también la relación entre rendimiento, coste y comportamiento del modelo.

5. Resultados y discusión

Los resultados preliminares muestran que la métrica basada en *embeddings* multimodales es la que mejor se alinea con los *rankings* humanos. Como se observa en la Fig. 5, la correlación media de Spearman alcanza $\rho = 0.795$, frente a $\rho = 0.766$ para SAVi y $\rho = 0.564$ para la evaluación visual. Las medianas siguen el mismo patrón ordinal: 0.820 para *Embeddings*, 0.783 para SAVi y 0.645 para Visión. Además, el acuerdo inter-evaluador fue elevado, con W de Kendall medio de 0.832 y mediana de 0.840 sobre las 10 escenas evaluadas. Estos resultados sugieren que, en esta tarea, la similitud global entre la descripción textual y el vídeo renderizado captura una parte importante de un criterio humano relativamente consistente de fidelidad semántica.

SAVi se comporta como una alternativa competitiva, pero no mejora la métrica de *embeddings* en este conjunto preliminar. Una posible explicación es que el juicio humano prioriza la presencia de los elementos semánticos principales de la escena, mientras que las señales de detalle o coherencia temporal pueden introducir ruido si se ponderan antes de disponer de una muestra humana mayor.

La Fig. 6 muestra el *ranking* agregado de modelos, ordenado por *Embeddings* al ser la métrica con mayor concordancia humana. En las primeras posiciones aparecen modelos recientes como `claude-opus-4.7`, `gpt-5.3-codex`, `gpt-5.4`, `mimo-v2-pro`, `gemini-3-flash-preview`, `claude-sonnet-4.6` y `minimax-m2.5`, lo que sugiere que la tarea discrimina entre modelos capaces de combinar comprensión semántica, generación de código y composición visual dinámica. Además, los modelos de Anthropic muestran una ordenación coherente con la jerarquía esperada de la familia (`Opus` > `Sonnet` > `Haiku`), aunque `claude-opus-4.7` debe interpretarse con cautela por su baja cobertura ($n = 5$).

El *ranking* agregado no debe interpretarse como una medida universal de calidad de los LLMs, sino como una estimación de su capacidad en esta tarea concreta: transformar una descripción visual en código ejecutable que produzca una animación semánticamente alineada. Para construirlo se promedian posiciones relativas por escena, evitando mezclar puntuaciones absolutas entre escenas con rangos de similitud distintos, y se aplica un umbral mínimo de cobertura para excluir evaluaciones testimoniales. Esta formulación permite observar qué modelos tienden a quedar por encima de otros dentro de la misma tarea y facilita el seguimiento de cambios a lo largo del tiempo.

La columna de costes de la Fig. 6 añade una lectura práctica. Algunos modelos situados en la parte alta del *ranking*, como `gpt-5.3-codex` o `gpt-5.4`, pueden implicar costes superiores, mientras que otros modelos, como `mimo-v2-pro` o `minimax-m2.5`, ofrecen resultados competitivos con precios potencialmente más moderados. Otros modelos de menor coste, como algunas variantes de `deepseek`,

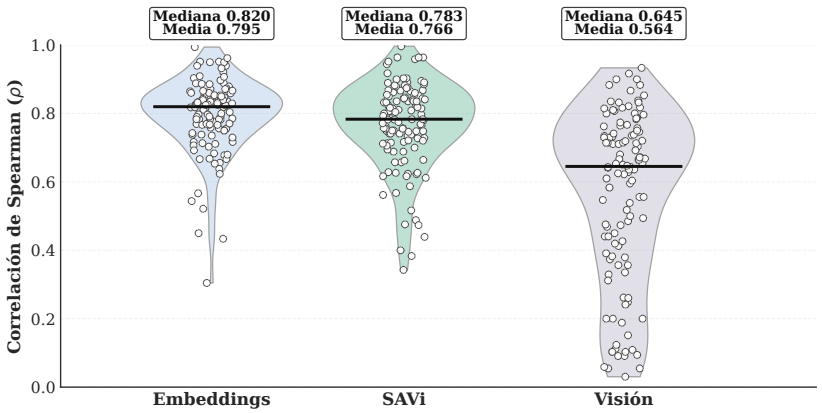


Figura 5. Correlación de Spearman entre *rankings* automáticos y *rankings* humanos. La métrica basada en *embeddings* obtiene la mayor correlación media con el juicio humano. Los puntos muestran las observaciones individuales de las 120 evaluaciones totales.

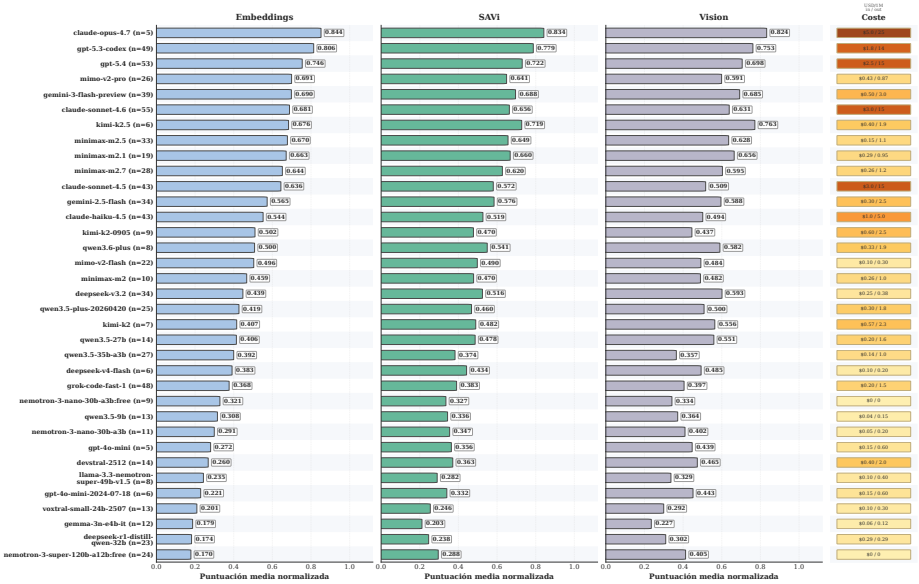


Figura 6. *Ranking* agregado de modelos por métrica automática. La figura resume 70 escenas, 40 modelos candidatos y 791 pares modelo-escena por métrica, equivalentes a 2373 puntuaciones automáticas. Las filas se ordenan por la media normalizada de *Embeddings*; SAVi y Visión muestran las mismas salidas bajo métricas comparativas. Para evitar coberturas testimoniales, solo se muestran modelos con al menos 5 escenas válidas, entendidas como ejecuciones que producen un vídeo evaluable; las etiquetas "n" indican esa cobertura. La columna derecha muestra el precio de entrada/salida en USD por millón de *tokens*, tomado del catálogo público de OpenRouter el 1 de junio de 2026 [8]; el color se escala con el coste medio.

no necesariamente alcanzan las primeras posiciones en esta instantánea, pero permiten analizar el compromiso entre calidad, cobertura y coste.

La tarea resulta exigente incluso para modelos avanzados: requiere comprender la escena, decidir una representación visual, producir código ejecutable y mantener coherencia temporal. Durante la primera mitad de 2026 se observaron mejoras cualitativas en los modelos más recientes, pero solo algunos modelos frontera generaron animaciones reconocibles, fluidas y semánticamente alineadas de forma consistente, lo que indica que el *benchmark* todavía no está saturado.

6. Limitaciones

Este trabajo presenta una validación preliminar y tiene varias limitaciones. En primer lugar, la generación de código por parte de los LLMs no es determinista: varias ejecuciones del mismo *prompt* podrían producir vídeos distintos. Aunque la métrica automática es reproducible para un par fijo *prompt*-vídeo, futuras versiones deberían estimar la estabilidad del *pipeline* con varias generaciones por modelo y escena. En segundo lugar, los resultados dependen del modelo de *embeddings*; otro modelo podría alterar distancias y *rankings*, y favorecer propiedades semánticas, como el reconocimiento de objetos, frente a dimensiones temporales o físicas. En tercer lugar, la muestra humana debe ampliarse para equilibrar perfiles de evaluadores y escenas.

La selección dinámica mediante OpenRouter introduce además variabilidad temporal: disponibilidad, popularidad, versión y precio de los modelos pueden cambiar entre ejecuciones. Esta variabilidad no es solo una limitación, sino también parte del objetivo del *benchmark*, ya que permite generar instantáneas sucesivas del ecosistema de modelos y analizar su evolución relativa a lo largo del tiempo. Por tanto, los resultados agregados deben interpretarse como una fotografía del periodo evaluado, correspondiente a marzo, abril y mayo de 2026. También será conveniente registrar de forma agregada la tasa de error por modelo, diferenciando fallos de generación, errores de ejecución y vídeos no evaluables. Por último, las escenas consideradas cubren un rango inicial de fenómenos visuales, pero no agotan la variedad de razonamientos espaciales, físicos y temporales posibles.

7. Conclusiones y líneas futuras

El artículo presenta un *benchmark* dinámico para evaluar LLMs en la generación de código renderizable desde descripciones visuales. La propuesta desplaza la evaluación desde el texto generado hacia el artefacto ejecutado: el código se renderiza como vídeo y se compara con el *prompt* mediante *embeddings* multimodales texto-vídeo. Así, el método evalúa una capacidad compuesta que integra comprensión de escena, generación de código, composición visual y resultado dinámico.

Los resultados preliminares muestran una alta correlación entre el *ranking* automático basado en *embeddings* y el juicio humano, y el *ranking* agregado

permite analizar también el coste de inferencia. Como líneas futuras, se ampliará la muestra humana, se incrementará la diversidad de escenas, se estudiará la estabilidad ante múltiples generaciones por modelo y se explorará la publicación de *prompts*, códigos, vídeos, *rankings*, precios, metadatos y tasas de error para facilitar comparaciones independientes.

Agradecimientos Este trabajo se ha desarrollado en el grupo ADMIRABLE de la Universidad de Burgos y forma parte de los proyectos de I+D+i PID2023-150694OA-I00 y PID2025-171001OB-I00, financiados por MICIU/AEI/10.13039/501100011033 y "FEDER/UE".

Declaración de intereses Los autores declaran no tener intereses en conflicto relevantes para el contenido de este artículo.

Referencias

1. Bansal, H., Lin, Z., Xie, T., Zong, Z., Yarom, M., Bitton, Y., Jiang, C., Sun, Y., Chang, K.W., Grover, A.: Videophy: Evaluating physical commonsense for video generation (2024), https://arxiv.org/abs/2406.03520
2. Farnebäck, G.: Two-frame motion estimation based on polynomial expansion. In: Scandinavian conference on Image analysis. pp. 363–370. Springer (2003)
3. Google DeepMind: Gemini Embedding. https://deepmind.google/models/gemini/embedding/ (2026), accedido el 20 de mayo de 2026
4. Huang, Z., He, Y., Yu, J., Zhang, F., Si, C., Jiang, Y., Zhang, Y., Wu, T., Jin, Q., Chanpaisit, N., Wang, Y., Chen, X., Wang, L., Lin, D., Qiao, Y., Liu, Z.: Vbench: Comprehensive benchmark suite for video generative models (2023), https://arxiv.org/abs/2311.17982
5. Kendall, M.G.: A new measure of rank correlation. Biometrika **30**(1-2), 81–93 (1938)
6. Liang, P., Bommasani, R., Lee, T., et al.: Holistic evaluation of language models (2023), https://arxiv.org/abs/2211.09110
7. OpenAI: GPT-4o System Card. https://cdn.openai.com/gpt-4o-system-card.pdf (2024), accedido el 28 de mayo de 2026
8. OpenRouter: OpenRouter Model Catalog: API Pricing and Providers. https://openrouter.ai/models (2026), los precios de entrada y salida por modelo fueron consultados en las páginas individuales del catálogo de OpenRouter; accedido el 1 de junio de 2026
9. OpenRouter: OpenRouter Rankings. https://openrouter.ai/rankings (2026), accedido el 20 de mayo de 2026
10. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision (2021), https://arxiv.org/abs/2103.00020
11. Spearman, C.: The proof and measurement of association between two things. In: Jenkins, J.J., Paterson, D.G. (eds.) Studies in Individual Differences: The Search for Intelligence, pp. 45–58. Appleton-Century-Crofts (1961). https://doi.org/10.1037/11491-005, https://doi.org/10.1037/11491-005
12. Zheng, L., Chiang, W.L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E.P., Zhang, H., Gonzalez, J.E., Stoica, I.: Judging llm-as-a-judge with mt-bench and chatbot arena (2023), https://arxiv.org/abs/2306.05685