INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

Materia

Interfaces y Multimedia

Docente

Ing. Richard Armijos

Estudiante

Wilson Martínez – Fabian Campoverde – Juan Campana

Repositorio GitHub: https://github.com/FabianSSJ/Interfaces-y-Multimedia-Trabajo-Videos

Estrategia de optimización de contenido de video multiplataforma

Introducción

En la actualidad, el video se ha convertido en uno de los formatos más dominantes en la web, esto es evidente no solo por su poder para captar la atención, sino también por su papel en las estrategias de educación, marketing, entretenimiento y comunicación, sin embargo, sin una optimización adecuada, los videos pueden generar problemas críticos de rendimiento, consumo excesivo de datos y experiencias de usuario frustrantes.

El objetivo de este trabajo es diseñar, implementar y evaluar una estrategia integral de video que funcione eficientemente tanto en web como en entornos móviles, considerando distintos anchos de banda y dispositivos. Se buscará:

- Definir y justificar los parámetros técnicos (resolución, bitrate, fps, audio) adecuados para distintos escenarios.
- Seleccionar formatos y códecs modernos (como H.264, VP9, etc.) para lograr una buena calidad con tamaños de archivo razonables.
- Implementar tanto entrega progresiva (archivos .mp4 / .webm) como streaming adaptativo (por ejemplo, HLS con varias calidades).
- Añadir accesibilidad mediante subtítulos (WebVTT) y controles adecuados.
- Medir métricas clave (tamaño, bitrate real, tiempo de arranque, estabilidad en redes limitadas) para comparar diferentes versiones.
- Con base en los resultados, proponer presets recomendados para futuros proyectos y un checklist de publicación.
- Con esta estrategia, buscamos crear una experiencia de video que sea ágil, accesible, eficiente y robusta, poniendo al centro tanto al usuario como las limitaciones técnicas reales.

Marco teórico

El video digital se fundamenta principalmente en tres parámetros técnicos: resolución, fotogramas por segundo (fps) y tasa de bits (bitrate). La resolución se refiere a cuántos píxeles tiene cada cuadro, lo que determina la nitidez de la imagen; los fps indican cuántas imágenes se muestran por segundo, lo que afecta la fluidez; y la tasa de bits es la cantidad de datos que se usan por segundo para representar ese video, influyendo directamente en la calidad visual y el tamaño del archivo.

En cuanto a los códecs y contenedores, un códec (codificador-decodificador) es el algoritmo que comprime el video para su almacenamiento o transmisión y luego lo descomprime para

reproducirlo. Los contenedores son “envoltorios” que agrupan video, audio y otros datos (como subtítulos) en un solo archivo. Por ejemplo, el códec H.264 es muy común por su equilibrio entre calidad y compatibilidad, mientras que H.265 / HEVC ofrece mayor eficiencia (más calidad con menos bitrate) pero requiere más recursos y no es tan universal aún.

Por otro lado, existen códecs más modernos y abiertos: VP9, desarrollado por Google, ofrece buena eficiencia y suele usarse especialmente junto con contenedor WebM. El AV1, creado por la Alliance for Open Media, es aún más eficiente en compresión, puede reducir significativamente el uso de ancho de banda comparado con H.264, aunque su codificación/decodificación puede ser más exigente computacionalmente. Para el audio, se usan códecs como AAC (muy compatible) o Opus, que es muy eficiente en tasas bajas y se adapta bien al streaming en web moderna.

Respecto al streaming adaptativo, existen dos formas comunes de entregar video: progresivo y adaptativo (como HLS o DASH). En el streaming progresivo, el usuario descarga el video como un archivo completo (.mp4, .webm) y lo reproduce, mientras que en el streaming adaptativo el video se divide en pequeños segmentos codificados a diferentes calidades. Un manifiesto (playlist) indica al reproductor qué segmentos hay disponibles y, según las condiciones de la red, este puede pedir segmentos de mayor o menor calidad para ofrecer la mejor experiencia posible sin interrupciones.En HLS, el manifiesto es un archivo M3U8 y los segmentos suelen ser de unos pocos segundos; en MPEG-DASH, el manifiesto es un MPD (Media Presentation Description) y puede usar diferentes duraciones y contenedores.

en términos de accesibilidad, es clave que el video incluya captions (subtítulos), una de las tecnologías más usadas para esto es WebVTT (Web Video Text Tracks), que permite sincronizar texto con el video y es compatible con HTML5 mediante la etiqueta <track>. Además, es importante que los controles del reproductor sean accesibles por teclado (para personas que no usan ratón) y que los subtítulos puedan activarse o desactivarse fácilmente, garantizando que el contenido sea usable para personas con discapacidad auditiva o con otras necesidades de accesibilidad.

Metodología y Desarrollo

Para el desarrollo de esta actividad se trabajó con un entorno mixto de herramientas de edición, codificación y análisis web. El clip base empleado fue un video producido en VEO2, con una duración aproximada de 30–60 segundos, que funcionó como insumo para todas las pruebas de codificación y streaming.

Herramientas y entorno de trabajo

1.Codificación y procesamiento

Para generar las distintas variantes del video (progresivo y HLS), se utilizó FFmpeg, debido a su precisión en el control de parámetros como resolución, tasa de bits, fps y códecs. Desde esta herramienta se exportaron:

Versiones progresivas en MP4 (H.264 + AAC).

Versiones en WebM (VP9 + Opus).

Un conjunto de calidades para streaming adaptativo mediante HLS, segmentado en resoluciones como 480p, 720p y 1080p.

Los parámetros clave definidos durante la codificación fueron:

Resoluciones objetivo según variante.

CRF / bitrate objetivo, dependiendo del códec.

FPS igual al del clip original para evitar artefactos.

Duración de segmentación en HLS (entre 2 y 6 segundos).

Códecs utilizados: H.264/AAC para MP4 y VP9/Opus para WebM.

2.Implementación y pruebas

Para integrar las salidas generadas se preparó una página HTML sencilla que incorporó:

Fuentes múltiples para video progresivo (.mp4 y .webm).

Carga adaptativa mediante hls.js en navegadores sin soporte nativo de HLS.

Subtítulos en formato WebVTT, añadidos mediante <track> para asegurar accesibilidad.

Controles personalizables compatibles con teclado.

La página fue servida mediante un servidor local, lo que permitió reproducir el comportamiento real de carga, solicitudes HTTP y tiempo de arranque.

Procedimiento de medición

Para evaluar peso, velocidad de carga y estabilidad de cada variante, se utilizaron las Herramientas de Desarrollador del navegador (DevTools). El procedimiento fue:

Abrir DevTools con

Ctrl + Mayús + J (Windows) o Cmd + Opción + J (macOS).

Acceder a la pestaña Network.

Recargar la página para registrar desde cero todo el tráfico.

Medir:

Tamaño real descargado (transfer size).

Bitrate efectivo por segmento o archivo progresivo.

Tiempo de arranque (primer frame visible).

Variaciones en redes simuladas (buena y limitada) mediante el panel “Throttling”.

Limitaciones del entorno

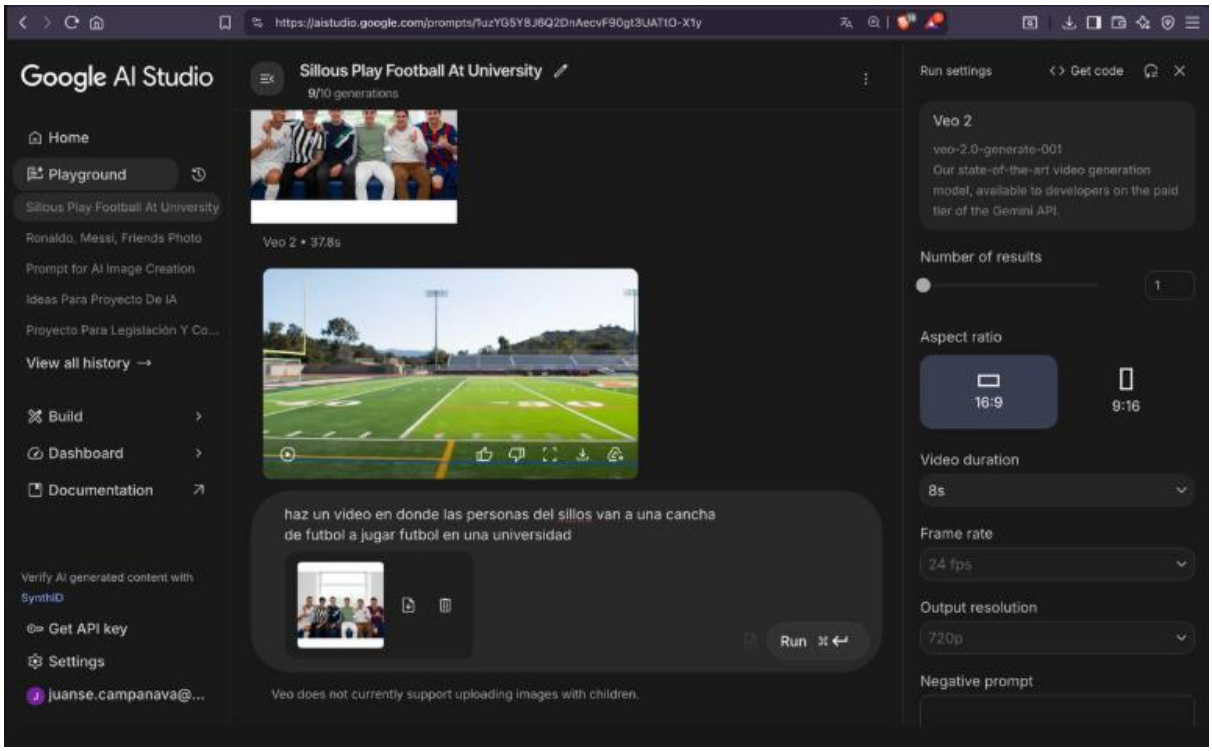
El proceso estuvo condicionado por:

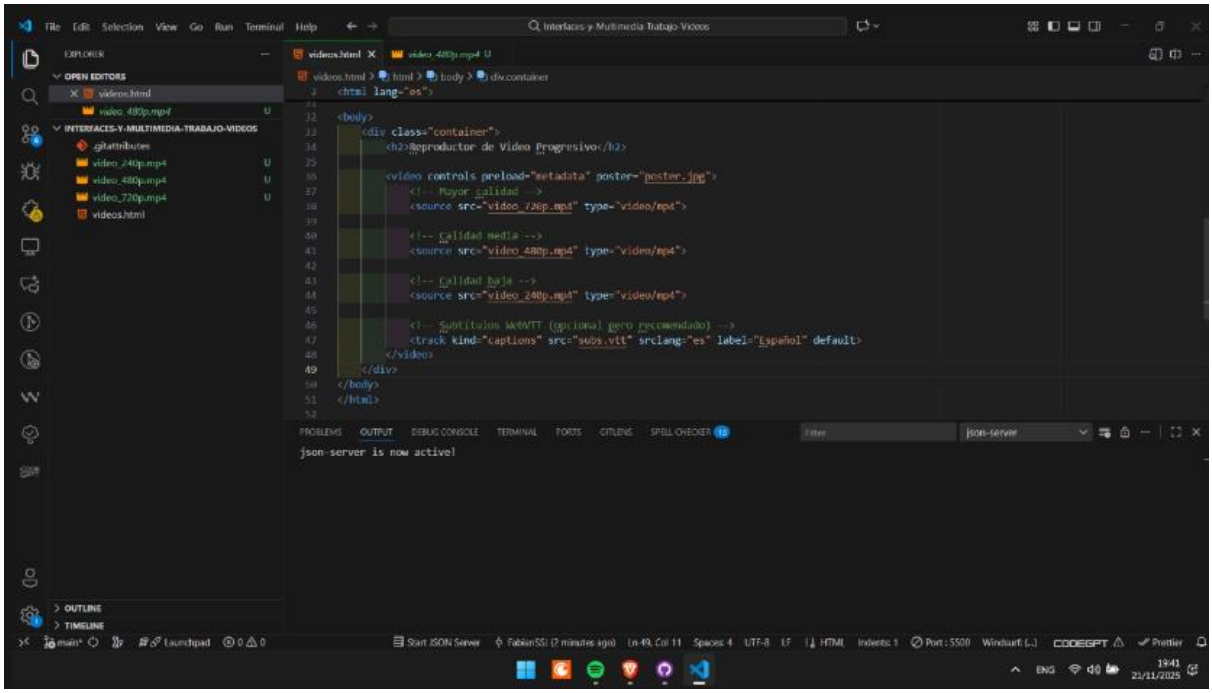
Equipo local: CPU y GPU influyen en la velocidad de codificación y en la decodificación de códecs más pesados como VP9 o AV1.

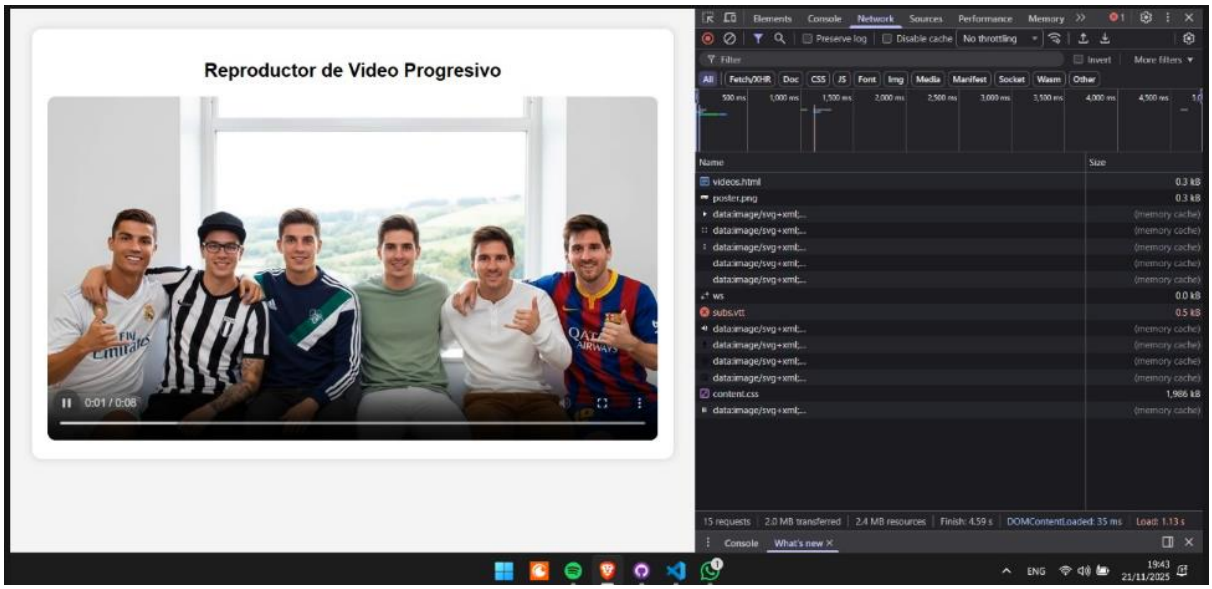
Condiciones reales de red: Aunque se aplicó throttling, la simulación no representa todas las variaciones posibles en redes móviles.

Navegador: Algunas mediciones dependen de la forma en que cada navegador maneja buffering, caché y soporte nativo de HLS/WebM.

Estas limitaciones se consideraron al interpretar los resultados, priorizando comparaciones internas entre variantes generadas bajo las mismas condiciones.







Resultados y discusión

Durante las pruebas comparativas entre las versiones progresivas (MP4 y WebM) y la variante adaptativa mediante HLS, se observaron diferencias claras en tiempos de carga, estabilidad en redes limitadas y peso final de los archivos.

La versión progresiva en MP4 (H.264/AAC) mostró la mejor compatibilidad y un tiempo de arranque relativamente rápido en redes estables. Su peso fue menor que otras configuraciones cuando se usaron valores de CRF equilibrados, por lo que resultó adecuada como opción base. Sin embargo, en redes limitadas la reproducción tendió a pausarse debido a la descarga lineal del archivo completo, lo que afectó la experiencia del usuario.

El WebM con VP9/Opus logró una mejor relación calidad–peso comparado con H.264, manteniendo buena nitidez incluso en archivos más pequeños. Su desventaja fue la menor compatibilidad en ciertos navegadores móviles, por lo que se recomienda como formato alternativo más que primario.

La solución que presentó el mejor desempeño general fue el HLS con múltiples calidades. En redes buenas cargó rápido gracias a su segmentación, y en redes lentas se adaptó automáticamente a resoluciones menores, evitando interrupciones. Aunque genera más archivos y requiere más procesamiento inicial (segmentación, playlist y varias codificaciones), su estabilidad en condiciones variables lo convierte en la opción más robusta para entornos reales.

Conclusiones y Recomendaciones

- Siempre usar múltiples fuentes (MP4 + WebM)
- Esto garantiza compatibilidad amplia sin aumentar en exceso la carga de trabajo. Es lo mínimo viable para cualquier proyecto web que incluya video.
- Priorizar HLS para contenido principal
- Cuando el tiempo lo permita, generar al menos dos calidades (por ejemplo, 480p y 720p). –
- Esto mejora radicalmente la estabilidad en redes variables.
- Optimizar con CRF y bitrates razonables
- Un CRF equilibrado (ni muy alto ni muy bajo) permite obtener buena calidad con un peso reducido, lo que acelera la carga y mejora la experiencia.
- Integrar subtítulos WebVTT y controles accesibles

Referencias

Google AI Studio. (s. f.). Prompt en AI Studio. https://aistudio.google.com/prompts/1uzYG5Y8J6Q2DnAecvF90gt3UATtO-X1y

AI-FutureSchool. (s. f.). Códecs de video H264, H265 y VP9 para alta calidad. https://www.ai-futureschool.com/es/informatica/codecs-de-video-h264-h265-y-vp9.php

Flussonic. (s. f.). Codificadores de Video: Cómo Funcionan y Su Papel en el Streaming y la Videovigilancia. https://flussonic.com/es/blog/news/videokoder

Hipertextual. (2015). H.265 y VP9: así son los codec que hacen posible el 4K. https://hipertextual.com/movilidad/h-265-y-vp9/

Xataka Home. (s. f.). Te contamos qué es el códec H.265 …
https://www.xatakahome.com/reproductores/te-contamos-que-es-el-codec-h-265-y-porque-mas-que-futuro-ya-es-el-presente-de-la-compresion-en-video