

CARVid: Content-Aware Retargeting of Videos With On-Screen Graphics

Anonymous Author(s)

Submission Id: 7605



Figure 1: CARVid is a mixed-initiative tool for retargeting on-screen graphics in videos to new aspect ratios while preserving readability and original design intent. Given an input video with embedded graphics (a), CARVid first decomposes the animated graphic from the video (b), then performs layout-aware segmentation on a reference frame to split meaningful elements such as logos, text blocks, and branding (c). Using the recovered structure together with its motion over time, CARVid can automatically recompose the graphic through text-aware scaling, content-aware space removal, and layout reflow, or expose direct editing controls that let users rearrange, group, duplicate, or remove elements (d).

ABSTRACT

Video editors routinely reframe landscape videos with on-screen graphics, such as lower thirds, title cards, and scoreboards, to portrait aspect ratio for sharing on social media. Existing methods often crop, shrink, or distort these graphics, reducing readability and disrupting design intent, while manual redesign is slow and

frustrating to scale due to difficulty dealing with layered compositions. We present CARVid, a content-aware retargeting tool for videos with on-screen graphics that helps editors adapt graphics to new aspect ratios while preserving readability and original design intent. CARVid leverages recent advances in foundation vision and segmentation models, in conjunction with a novel image processing algorithm, to detect graphical overlay bounding boxes, extract graphics from the video, and perform content-aware recomposition. Our human-in-the-loop approach combines text-aware scaling, layout-aware segmentation, and content-aware space removal to support both automatic and user-directed recomposition. We conducted a technical evaluation using a custom dataset of 60 videos from popular internet sources containing on-screen graphics. Results show that our method effectively aids users compared to traditional workflows.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...\$15.00

<https://doi.org/XXXXXXXX.XXXXXXX>

CCS CONCEPTS

• Information systems → Multimedia content creation; • Human-centered computing → Interactive systems and tools.

KEYWORDS

video, retargeting, content-aware, on-screen graphics

ACM Reference Format:

Anonymous Author(s). 2018. CARVid: Content-Aware Retargeting of Videos With On-Screen Graphics . In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 13 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 INTRODUCTION

Social media has created a landscape in which creators need to make multiple versions of a video to suit the constraints of different platforms and viewers. Videos are often reframed to different resolutions for smaller screens or displays with varied aspect ratios to fit different devices. Many of these kinds of social media videos feature on-screen graphics such as title cards, scoreboards, banners, and lower thirds, which are placed over the source video during editing. However, these overlaid graphics can create problems when videos are naively reframed. Traditional workflows for reframing involve cropping the video focusing on the salient content such as people or places by using tools such as Premiere Auto reframe [1] or OpusClip [27]. However, during this process, graphic overlays are not taken into consideration resulting in them being cropped, cut out or partially blocked from the main view. Many lower thirds or on-screen graphics take up considerable width, so repositioning alone is often not enough. In many practical cases, editors do not have access to the original video project file or underlying composition layers, and therefore resort to cropping tools. In other cases, the entire 16:9 video is uniformly scaled down to fit the target aspect ratio, which preserves more of the composition but can severely reduce the readability of the graphics. As shown in Figure 2, these are not merely hypothetical failures: they appear in published, widely viewed videos by official networks and professional editors, where graphics are either cut out from the frame or reduced to the point that critical information is obscured. Together, these examples highlight an important gap in current retargeting workflows: existing methods prioritize the main footage, but do not provide a way to adapt overlaid graphics to new aspect ratios while preserving readability and original design intent.

While recent advancements in video generation and AI-assisted video editing show promise for creating or modifying videos, these approaches are not well suited to our problem. Our goal is not to generate a new graphic from scratch, but to retarget an existing authored graphic which was designed to communicate a message to the audience while preserving its text, branding, layout hierarchy, and animation structure. In this setting, plausible synthesis is not enough: even small changes to fonts, spacing, visual grouping, or motion timing can conflict with the creator's original intent and alter how the graphic communicates information. Existing generative graphics and text rendering methods face inconsistent fonts, style variation, and limited fine-grained control [35]. This intuition informs our approach. Rather than treating retargeting as a fully

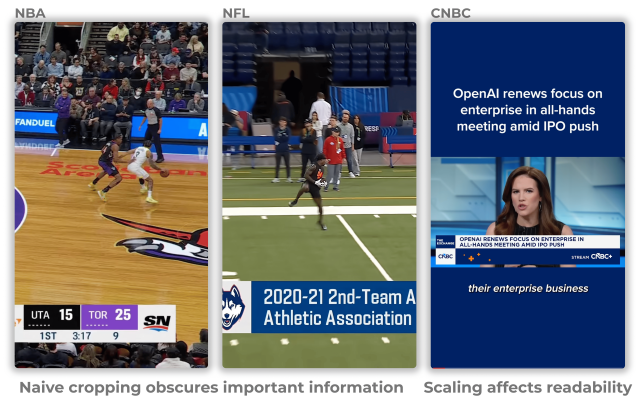


Figure 2: Snapshots above are taken from published YouTube videos from popular sports and news channels (© NBA [24], © NFL [25], © CNBC [7]) with millions of viewers. These videos are often repurposed into short-form, reframed content for social media. However, traditional reframing workflows tend to prioritize the main footage while overlooking on-screen graphics. As shown here, naive cropping can obscure important scoreboards and captions, while uniform scaling can make the original graphic unreadable.

generative video task, we focus on recovering the structure of existing on-screen graphics and transforming them in a way that respects both creators' and viewers' goals.

In this paper we ask *what if we could extract usable graphics from existing broadcast videos and automatically recompose these graphics to new aspect ratios just as we do for regular content*. And going beyond that to semantically segment parts of these graphics to enable editability. For example, take the case of the video in figure 1. The graphics in the lower part of the video spans the whole width and contains important branding and communicative information. If we attempt to reframe this graphic using existing tools, we would crop to salient content like the driver. However, we lose the title card information and the branding information which hold important communicative intent. Many examples are shown in figure 2.

To tackle this problem, we first devise an automatic algorithm with a set of simplifying assumptions and design goals about our problem. Our algorithm takes an input overlaid graphic and automatically recomposes it to a target width. We then conduct a formative study with four professional motion graphic designers who have experience with retargeting overlaid graphics. Their feedback informed our three key design goals: 1) supporting retargeting from flat or already-published videos without requiring original project files, 2) enabling structured editing of recovered graphic elements, and 3) preserving direct editorial control over retargeting decisions

Taking these design goals into consideration, we present CARVid, a content-aware retargeting tool for videos with on-screen graphics that helps editors adapt graphics to new aspect ratios while preserving readability and original design intent. Given an input flat video with baked graphics, CARVid first performs graphics Decomposition to extract the animated graphic from the video. It then performs Layout-Aware Segmentation on a representative reference

frame to split the recovered graphic into meaningful elements such as logos, text blocks, and branding. Once this semantic representation is recovered with its motion over time, CARVid supports graphic recomposition using custom methods in two ways: 1) Layout Reflow, which restructures the graphic to better fit the target aspect ratio, and 2) Content-Aware Space Removal, which removes low-importance space between elements while preserving important content and overall design structure.

To enable this workflow, CARVid leverages recent advances in foundation vision and segmentation models such as Gemini [12] and Segment Anything [31] to detect graphical overlay bounding boxes, extract graphics from flat videos, and recover a decomposed representation for downstream editing. Our decomposition pipeline relies on the notion of a *representative frame*, which a VLM agent uses to localize the main graphic and guide segmentation across the video. However, recovering an extracted graphics video in pixel space is not enough by itself. CARVid additionally performs Layout-Aware Segmentation by horizontally and vertically segmenting the recovered graphic while semantically considering its layout and content, producing *semantic primitives* that can be independently recomposed. By recovering semantic structure directly from flat videos, CARVid avoids requiring editors to rebuild or manually recover layered compositions, while still supporting settings in which separate graphic and background layers are already available.

To evaluate CARVid, we conducted technical evaluations of both our decomposition and recomposition techniques with using a custom corpus of 60 popular videos with graphic overlay videos found on YouTube. Our findings suggest that CARVid successfully decomposes 85% of the graphic overlays found in-the-wild and 96% of decomposed results fit to the target width.

In summary, this paper makes the following contributions: 1) a **graphics decomposition pipeline** that recovers usable on-screen graphics from flat videos through text refinement and user editing, producing, 2) a **layout-aware graphics segmentation** method that leverages critic based VLM agents to produce semantic primitives such as logos, text blocks, and branding, and 3) a content-aware recomposition pipeline that retargets recovered on-screen graphics to new aspect ratios while preserving readability and original design intent through layout reflow and content-aware space removal.

2 RELATED WORK

Our work builds on prior research in content-aware reframing, motion graphics editing, and generative video editing, and extends these ideas to the specific challenge of retargeting videos with on-screen graphics.

2.1 Content-Aware Reframing

Prior work adapts video to new aspect ratios by estimating subject saliency and scene structure using gradient energies, face and body detections, or saliency maps. Commercial tools such as *Auto Reframe* crop footage to keep salient objects like people or dogs centered when converting to new aspect ratios [1, 9]. Recent systems extend this idea. *VidSTR* [44] uses speech content and visual

semantics such as pose to guide spatiotemporal reframing, improving continuity across cuts and scenes while avoiding salient background occlusions. *FitVid* introduces a decomposition pipeline to retarget video content such as images and text to new layouts [16].

Much content-aware resizing traces back to seam carving [3], which removes connected low-energy pixel paths for adaptive image resizing. Later work extended this idea through multi-operator formulations combining carving, scaling, and cropping [32, 34], video adaptations with forward energy for temporal coherence [33], and analyses of the trade-off between content preservation and spatiotemporal consistency [6]. Other methods explored discontinuous seam carving and related strategies to reduce distortion in regions containing people or moving cars and improve consistency during resizing [29, 42]. Building on these ideas, *SeamCrop* combines cropping with seam carving to place an optimal reframing window while removing less informative seams [15]. Earlier vision work also explored non-homogeneous content-driven warps for video retargeting [43], the trade-off between content preservation and spatiotemporal consistency [6], and pixel stretching or warping for resizing [29, 40–42]. Soe et al. study interactive reframing for narrower displays through an end-to-end tool that makes content-aware decisions about people and related elements [36].

Collectively, these methods reframe video by tracking faces, bodies, and moving objects or by optimizing generic saliency. As a result, they often treat overlaid graphics as background, or treat all pixels as equally expendable as in seam carving [3]. Semantically critical elements such as titles, logos, scoreboards, and lower-thirds are therefore often cropped, squeezed, or misaligned. These approaches also do not explicitly process graphics as structured layouts composed of text, logos, and branding elements, nor do they account for motion-graphics behaviors such as animated slide-ins and slide-outs. For on-screen graphics, preserving readability, hierarchy, branding, and original design intent is often more important than uniformly reducing content as in ordinary video.

2.2 Motion Graphics Editing

A complementary line of work makes motion graphics *editable* by recovering structure from pixel videos and modeling layout as a first-class design object. Zhang et al. present a motion vectorization pipeline that converts motion-graphics video into an SVG motion program, enabling object-level edits such as retiming, motion transformations, and appearance changes [48]. Beyond vectorization, layout-centric systems model the relationship between content and design: *content-aware generation of graphic design layouts* learns layout structures from visual and textual semantics for layout-aware retrieval and content-conditioned composition [49], while *Attribute-Conditioned Layout GAN* studies controllable attribute-based layout generation [21]. Other graphics and HCI systems target branded motion assets: *Mover* [23] generates parameterized motion behaviors for design elements, and *LogoMotion* [22] produces structured logo animations that preserve style and layout intent. Together, these works treat motion graphics as programmable structures governed by constraints and attributes rather than purely pixel-based edits.

However, these methods rely on representations such as SVG programs, layout graphs, or parameterized animation code that are

far removed from everyday professional workflows. In practice, editors often work with MP4s and Adobe After Effects projects or templates, where overlays are effectively black boxes and difficult to recover as structured elements. Even when structure can be recovered, it may not be readily editable through motion programs, and it remains unclear how such representations support *reframing* decisions. More importantly, generative motion systems can suggest edits or synthesize simple motion, but they do not guarantee preservation of original design intent and may make decisions that conflict with a designer’s choices. Thus, although these works improve the programmability of motion graphics, they do not solve the central problem of preserving original text and design during aspect-ratio retargeting.

2.3 LLM-Assisted and Generative Video Editing

Recent systems combine multimodal understanding, language interfaces, and generative models to support video editing. *LAVE* introduces an LLM-powered agent that plans and executes editing tasks from language prompts, from brainstorming and storyboarding to concrete timeline manipulations [38]. *PodReels* supports human-AI co-creation of short podcast teasers by automating clip discovery and assembly based on audience-oriented criteria [39]. In advertising workflows, *Text2Edit* proposes a controllable end-to-end pipeline for video ad creation from multimodal prompts [5], while *VC-LLM* targets automated ad production from raw footage with multimodal LLM planning and selection [30], similar to ExpressEdit, which uses language and sketches to perform editing tasks [37]. Several other works use natural language to guide cuts and edits [13, 19, 20].

Another active area is generative video AI, including OpenAI’s Sora [26] and Google’s Veo3 [11], which generate videos from text or image prompts. Agentic reframing has also emerged, with LLM-based controllers orchestrating detection, selection, and windowing for new canvases [4]. Together, these works mark a shift from manual, clip-centric editing toward language-conditioned, content-aware, and sometimes generative pipelines. Yet they focus primarily on *editing* and *creation*, not aspect-ratio retargeting with explicit preservation of overlaid graphics and original design intent. Editing agents generally lack access to motion graphics’ underlying structure and may crop or scale them without awareness, while generative models may distort fonts, logos, motion, and design because they do not reliably reproduce exact graphics [47, 50].

3 AUTOMATIC GRAPHICS RETARGETING

To design our system, we began by exploring the automatic retargeting of the readily available motion graphic templates (MOGRTs). We chose MOGRTs because they are a common resource used in professional editing workflows and provide access to structured graphic compositions with editable text and layout properties. This initial exploration allowed us to formulate the problem in a controlled setting and build an early automatic pipeline for recomposing graphics to a target width. We then conducted formative interviews with professional motion graphics designers to understand real-world retargeting workflows and to derive the design goals that shaped CARVid.

Problem Statement. We define a “graphical element” to be a single on-screen graphic such as a title card, banner, lower third, or scoreboard, including its animations and effects over time, from a standard digital video editing tool (e.g. Adobe Premiere Pro). The videos we consider consist of a single recorded background video, with a single rendered graphical element overlaid on top. Nominally these videos are landscape (horizontal, 16:9) aspect ratio (for viewing on television, YouTube, etc.), and the user goal is to retarget such a video to portrait (vertical, 9:16) aspect ratio (for viewing on Instagram, TikTok, etc.). This is traditionally accomplished by cropping, selecting the region of the video that contains the focal point (often the most salient region). In this scenario, the overlaid graphical element is then cropped, often obscuring the text and design elements, hindering communication (Figure 2).

When faced with the problem we have identified here, a video editor has the following options for retargeting a video with an overlaid graphical element: 1) Use existing reframe tools, 2) Reposition the graphics, 3) Scale the graphics and 4) Redesign the graphical element. To address these limitations, we first devised an automatic retargeting algorithm guided by three themes from prior work: text must remain legible, non-text visuals and their spatial relations should not be degraded, and removing non-informative space should not harm hierarchy. Guided by this body of work, we state our design goals: **DG1** *Maintain text at or above the minimum legible height.* **DG2** *Preserve all non-text visual elements.* **DG3** *Maintain the visual hierarchy.* **DG4** *When preservation of all elements is not possible, prioritize text over non-text visuals.* To explore this problem in a controlled setting, we first built an automatic retargeting pipeline using readily available motion graphic templates (MOGRTs). This gave us a way to study retargeting behavior before dealing with the added complexity of flat videos with baked graphics. Starting from these templates, we devised an early algorithm that automatically recomposed a graphic to fit a target width while attempting to preserve readability and overall layout. Our initial approach focused on operations such as text-aware scaling, preserving the positions of key non-text elements, and removing low-importance empty space when possible.

3.1 Algorithmic Steps

Our automatic retargeting algorithm applies a sequence of increasingly disruptive operations to adapt a graphic to a target width while preserving readability and original design structure. It first identifies the most important regions to protect, then attempts the least invasive transformation that can satisfy the size constraint: uniform scaling, followed by removal of persistently low-importance space, and finally layout reflow when width reduction alone is insufficient. This ordering follows our problem definition directly: preserve readable text, maintain important non-text elements and their spatial relations, and only introduce stronger structural changes when simpler operations fail. Figure 3 shows one example graphic passing through these stages.

1. Compute Energies. We first compute an energy map for each frame to identify regions that should be preserved during retargeting. Rather than using a generic image retargeting objective, we focus on preserving text and strong vertical structures that define the graphic’s shape and layout. OCR is used to detect text

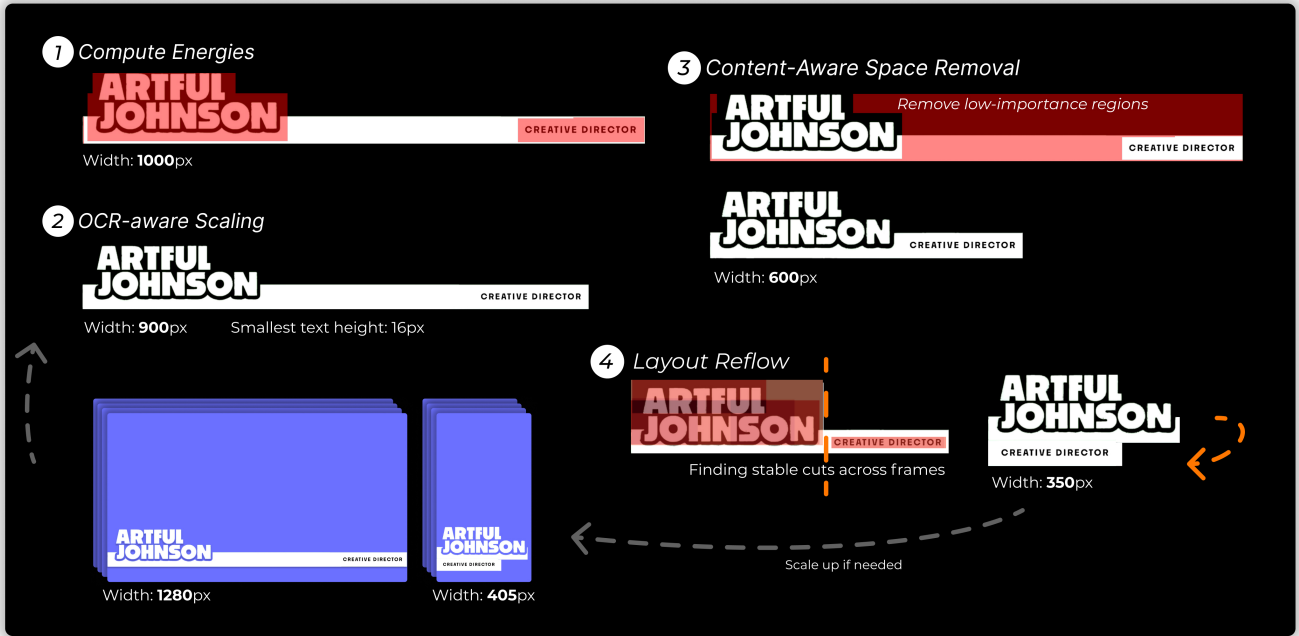


Figure 3: An example graphic going through all stages of our preliminary algorithm pipeline is shown here. Given an input video we 1) Compute energies for all frames to mark regions of importance, 2) Scale the graphics based on minimum text height 3) In this case, the graphic still does not fit, so we remove low importance regions, and finally, 4) If it still does not fit, we reflow the graphic.



Figure 4: Example results from our preliminary automatic retargeting algorithm as shown to our participants. See supplemental video for animations. We are able to preserve many different types of content. *Left*: the bar has a simple in and out animation, and includes multiple vector graphics. *Center*: The three bars animate in at different times and speeds. *Right*: The graphic appears with a build animation and includes a QR code.

regions, which are treated as high-importance areas, while vertical edge responses highlight boundaries around logos, boxes, and other design elements. These frame-wise energies are then aggregated into per-column importance scores, which serve as the basis for both space removal and layout reflow.

2. OCR-Aware Scaling. We next attempt uniform scaling while enforcing a minimum readable text size. The graphic is scaled down only until the smallest detected text reaches the minimum legible height; if the graphic fits the target width at this stage, the result is exported directly. This allows the system to preserve the original design as much as possible before applying any content-aware restructuring.

3. Content-Aware Space Removal. If scaling alone is insufficient, we remove low-importance vertical space from the graphic while preserving temporal coherence across frames. Because these graphics are animated, removable columns may change from frame to frame; selecting them independently would introduce jitter and break the animation. Instead, we track temporally stable low importance columns over time, avoiding text, logos, strong borders, and moving content. This step progressively reduces width by sacrificing empty or weakly informative space while maintaining alignment and the original motion structure.

4. Vertical Layout Reflow. When neither scaling nor space removal can achieve the target width, we reflow the graphic into multiple lines. To do this, we compute a composite energy map over sampled frames and search for a stable vertical cut that avoids persistent text and high-importance content while aligning with a natural structural divider in the layout. The cut is then used to split the graphic into two parts, which are rearranged into a narrower multi-line composition. In practice, this is the most aggressive transformation in the pipeline and is only applied when the

earlier stages cannot preserve readability and fit simultaneously. We detail our automatic algorithm in the appendix.

3.2 Preliminary Feedback from Motion Graphics Professionals

We then conducted a formative interviews with four professional motion graphics designers who had experience with retargeting overlaid graphics. In the interview, participants were shown a few result from our initial automatic algorithm which used popularly available MOGRTS as input and produced a 9:16 aspect ratio output, which were rendered on stock video footage. Snapshots of these videos can be found in figure 4. Participants consistently described retargeting as labor-intensive and often close to redesign, since adapting graphics across aspect ratios frequently requires manually re-adjusting text, spacing, layout, and animation while still preserving the original design’s readability, hierarchy, branding, and motion. Rather than treating retargeting as a simple re-size operation, they framed it as a judgment-heavy process that is both tedious and poorly supported by existing workflows. These findings informed three key design goals for CARVid: supporting retargeting from flat or already-published videos without requiring original project files, enabling structured editing of recovered graphic elements, and preserving direct editorial control over retargeting decisions.

Supporting retargeting from flat or already-published videos without requiring original project files. Participants described current retargeting workflows as tedious precisely because they often require going back into the authored motion graphic, reopening the composition, and manually reworking its layers and animation. Rather than treating resizing as a lightweight operation, they described it as painstaking layer-level revision. For example, participants noted going “layer by layer” to shift text, reposition logos, and realign keyframes, and E3 explained, “*I need to redesign everything, readjust the keyframes in After Effects or Premiere Pro to keep the same animation [and reframe the graphics].*” E4 similarly characterized this stage as largely “*mechanical work.*” Taken together, these comments suggest that relying on original layered compositions is both cumbersome and fragile in practice, motivating a system that can begin from an already-rendered video and decompose the overlaid graphic back out of the main footage for downstream retargeting.

Enabling structured editing of recovered graphic elements. Participants did not want the output to remain a single flattened graphic; instead, they wanted to adjust meaningful parts of the composition such as logos, text regions, spacing, and layout parameters. This points directly to the need to recover editable semantic units rather than only pixels. E2 asked for “*settings or configuration that the user could have control over, like, layout, content padding or the scalar resolution,*” and made the desired granularity even clearer by saying, “*Maybe I want that fox logo to be somewhere else, like, not in the lower third.*” E1 similarly imagined a workspace where he could generate multiple versions and “*make global adjustments to each of those if I had to.*” These responses motivate recovering

graphic elements as structured parts that can be independently manipulated, rather than treating the result as a one-shot automatic transformation.

Preserving direct editorial control over retargeting decisions.

Participants consistently framed retargeting as a judgment-heavy design task in which acceptable outcomes depend on context, audience, and the priorities of the editor. They wanted automation to help, but not to fully determine the final tradeoffs. E1, for instance, said, “*I really want [to change] [minimum text] size, because that’s also depending on audience,*” while E2 emphasized the need to repeatedly inspect outputs in context: “*I would also... check pretty often, once I have the portrait version, to see if it looks good on the phone.*” Even trust in the system was framed as something gradual and contingent, with E2 noting, “*I would see how, you know, once I get a hang of the tool, I’ll be like, oh yeah, it seems, trustworthy.*” Together, these comments suggest that the system should support retargeting decisions while keeping the editor in control of which compromises are ultimately made.

Moving beyond an automatic pipeline. Overall, participants felt that this task could not be solved reliably as a fully automatic pipeline. While automatic retargeting could provide a useful starting point, they emphasized that on-screen graphics carry semantic, typographic, and branding constraints that require human judgment. Small changes in crop boundaries, spacing, grouping, or scale could significantly affect readability and design quality, especially for titles, logos, scoreboards, and lower-thirds. Participants also noted that graphics often behave as coordinated layouts rather than independent visual regions, making it difficult for a purely automatic method to decide what should be preserved, removed, or rearranged. These observations led us to move beyond a fully automatic approach and instead design CARVid as a mixed-initiative system, where automatic decomposition and reframing generate an initial proposal that users can inspect and refine. To this end, we designed CARVid around layout-aware segmentation and decomposition

4 CARVID

This section presents CARVid, a human-in-the-loop recomposition tool for on-screen graphics in videos.

4.1 Video Decomposition

We generalize our system to non-layered inputs by introducing a decomposition pipeline that separates overlaid graphics from the underlying footage into clean, temporally stable masks. The process begins by sampling frames and selecting an anchor frame around the region of maximal text density, identified by finding a frame close to the center with the most OCR boxes. We assume, for on-screen graphics, that the representative frame would be the one close to the center and which has significant screen time. The graphic may have in and out or slide-in or moving animations, but to be consumable to the viewer, it has to stabilize and be readable for a significant amount of time over the duration of the sequence. Results of this decomposition process are shown in the appendix in Figure 5, demonstrating that the method can successfully recover graphics layers from flat video inputs.

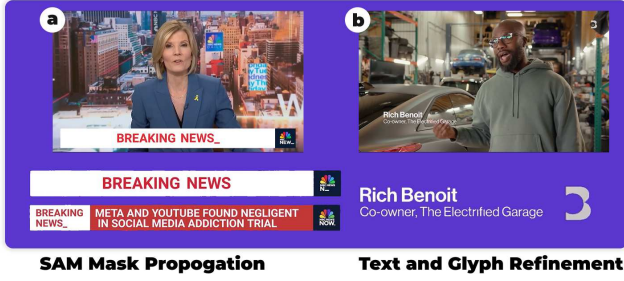


Figure 5: CARVid leverages the foundational Segment Anything model with VLM-guided graphic detection across three reference frames to segment bounded graphics (a). CARVid also supports the extraction of text- or glyph-based graphics (b) by leveraging text-guided SAM and glyph extraction with adaptive image thresholding.

Identifying a Reference Frame. We define the notion of a reference frame with a simplifying assumption that a frame with the most words/text would serve as our anchor because on-screen graphics are typically used to convey information. We also learn from our interviews that the purpose of these graphics is purely communicative. We use a lightweight open-source OCR [14] library and run it on every frame of the video. From the middle frame, we scan for the frame with the most text as middle frames would be free of in and out animations. We also store a distribution of word count changes every 20 frames to segment between different transition stages of graphics. This reference frame is useful for downstream processing with VLM agents.

VLM Detection and Verification. On the anchor frame, a multi-modal vision model (*gemini-3.1-pro-preview*[8]) identifies a bounding box for all units of on-screen graphics which are close together, while excluding concave free floating text rendered directly on footage, if any (this is done due to the limitations of current foundational segmentation models). Each detected bounding box seeds SAM2’s [31] video segmentation predictor with a set of positive points sampled within the bounding box and negative points outside it, producing an initial preview segmentation over this anchor frame. This mask is composited onto the frame in a high-contrast color and passed to a critic VLM, which inspects the masked preview against the original frame and either accepts the segmentation or returns a set of additional positive prompt points targeting under-segmented regions.

SAM Mask Propagation Detection. These segmentation prompts are injected directly into SAM2’s mask logits to refine the prediction before propagation. Once the critic accepts, the final mask is propagated forward and backward across the full video. To ensure masks are compositable, morphological erosion followed by soft feathering is applied, suppressing visible halos at the boundaries. As the segment-anything model supports prompt refinement across multiple frames in the video we prompt the VLM to sample three detections across three frames: one reference frame, one frame at 25% video length and another at 70% length.

Text and Glyph Refinement. While SAM2 excels at propagating masks for solid, contiguous regions, it is known to struggle with shapes containing interior holes or disjoint components [31], a property that is pervasive in typographic and glyph-based graphics. To address this, we introduce two complementary refinement branches that operate in parallel with the primary SAM2 pipeline.

Text Refinement. During the anchor-frame classification stage, Gemini explicitly identifies whether a detected graphic is purely text-rendered without any opaque background shape. For such cases, we bypass SAM2 and instead invoke HI-SAM [45], a segmentation model fine-tuned from SAM specifically for hierarchical text instance segmentation, which preserves per-glyph fidelity and handles the interior holes inherent to letterforms. Since HI-SAM operates on individual frames rather than video, temporal consistency is achieved by tracking detected text regions with OCR across frames: bounding regions from the anchor are matched forward and backward via OCR correspondence, and HI-SAM inference is applied per-frame within the tracked crops. The resulting per-frame masks are then composited into the final mask sequence, yielding a text-only segmentation that preserves font structure.

Glyph Refinement. For logos and other glyph-type artifacts whose shapes contain interior holes and for which the critic VLM determines the SAM2 mask preview is of insufficient quality, we fall back to a bounding-box-constrained adaptive thresholding approach. The critic VLM localizes the glyph on the reference frame via a tight bounding box, which is held fixed across all frames under the assumption of a static graphic. Within each cropped region, we apply Otsu adaptive thresholding [28] to separate the foreground glyph from the background, producing a clean binary mask per frame. This approach is inherently limited to single-color glyphs, where foreground and background are well-separated in intensity, but provides a reliable last-resort path when learned segmentation models fail on such structured artifacts.

Interactive Refinement. Finally, if the user is not satisfied with any of the automatic results, they could leverage our interactive refinement tool where they can choose which method to use and add point/box prompts themselves. Moreover, they have the ability to scrub through different frames and choose a prompt from there for processing. They also have the option to generate separate segmentations of different parts of the object and combine them.

4.2 Layout-Aware Graphics Segmentation

4.2.1 Cut Candidate Generation. Using the temporal energy map E , we identify candidate cut positions by locating local minima in the column-wise energy profile $\ell(x) = \sum_y E(y, x)$. These minima are pooled with structural boundaries detected via probabilistic Hough transforms—thin designer-placed divider lines that are visually significant but leave little gradient signal. Each candidate is then snapped to the nearest low-energy gap and scored by *structural continuity*: the length of the longest unbroken high-gradient run in that column. Genuine panel boundaries span the full frame height; character strokes do not. The top-ranked candidates form the set C presented to the VLM.



Layout-Aware Graphics Segmentation

Figure 6: CARVis supports automatically segmenting the graphic layout into semantic parts by recursively cutting the graphic using an proposer-critic agent paradigm until it either fits the screen fails.

4.2.2 Proposer-Critic Placement. Selecting the final cut requires semantic judgment that signal processing alone cannot provide—specifically, whether the resulting segments each constitute a complete, self-contained unit of information. Motivated by evidence that a VLM critic reliably catches errors that a VLM proposer makes in isolation [46], we adopt a three-stage protocol. A *gate* call first decides whether the graphic is cuttable at all. A *proposer* then selects the single best candidate from the energy-annotated image. A *critic* receives both the clean graphic and the energy map and makes the final call, rejecting any option that bisects a text region or splits a cohesive content group:

$$p^* = \text{Critic}(\text{graphic}, E, p_{\text{naive}}, p_{\text{proposer}}).$$

Any accepted position is further subject to a hard programmatic check: if p^* falls within 3 pixels of a text mask pixel, it is nudged to the nearest safe column or discarded entirely.

4.2.3 Resultant Cuts. This process can give us cuts in either the vertical direction or the horizontal direction based on the graphic. Moreover, the user can choose to click “layout” reflow which gives them a choice to cut horizontally or vertically. They can do this process on individual graphic segments recursively.

4.3 Interactive Editing

CARVis supports direct-manipulation editing operations that align with common motion graphics workflows. Users can delete unwanted elements, duplicate graphics, group related components, and invoke recomposition operations such as space removal and layout reflow on selected content. These controls are useful when automatic retargeting preserves the wrong elements, fails to maintain intended grouping, or produces layouts that technically fit the new aspect ratio but remain visually unbalanced. For example, branding elements can be selectively removed while preserving the scoreboard, or multiple primitives can be grouped and moved together to maintain their semantic and visual relationship. Moreover, when generating a cut the user can choose to refine the cut

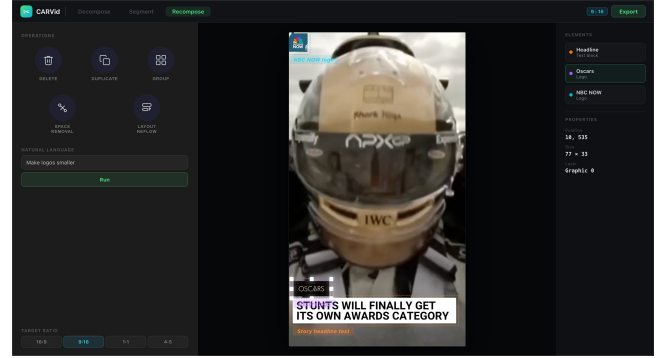


Figure 7: CARVis also supports manual direct manipulation, layout edits, and natural language bulk edits based on the extracted layout structure.

so that the resultant segments are up to their choice. Or they can choose to go with the system recommendation.

5 TECHNICAL EVALUATION

We conducted a technical evaluation of our decomposition pipeline, layout segmentation technique, and our automatic recomposition algorithm to test the efficacy of CARVis in retargeting videos found on the internet in the real-world.

5.1 In-The-Wild Evaluation Corpus

We constructed our evaluation set from publicly available YouTube videos using seeded snowball sampling. We began with broad seed queries spanning common sources of on-screen graphics, including “sports live,” “sports highlights,” “news live,” “breaking news,” “interview,” “podcast interview,” and “weather report.” From these seeds, we selected videos containing persistent overlaid graphics such as scoreboards, lower thirds, title cards, or corner bugs, and then followed YouTube recommendations to expand the set toward visually similar but channel-diverse examples. We extracted 5-second segments only when a graphic remained visible for most of the clip and was composited over underlying video footage rather than occupying the full screen. We excluded pure subtitles and captions, full-screen slates, and near-duplicate clips from the same graphic package.

Our final corpus contains 60 clips spanning sports (24), news (17), interviews (13), and weather (6). We further categorize clips by the visual support behind their primary information-bearing elements: opaque-backed graphics place text and symbols on visually solid backing panels (49 clips), transparent graphics place them directly over video without a contiguous backing region (5 clips), and translucent-backed graphics place them on bounded semi-transparent panels through which the video remains visible (6 clips). For mixed designs, we assign the label according to the dominant backing behind the main information-bearing elements. The corpus was not class-balanced by design; instead, it reflects the natural distribution of graphics encountered during collection, where opaque-backed graphics were substantially more common. All experiments were conducted on an Apple MacBook Pro equipped with an M3 Max processor and 16GB of RAM.

	Success	Failed	Success Rate
VLM-Guided Segmentation	75	24	75.8%
+ Interactive Editing	81	18	81.8%
+ Text Refinement	85	14	85.9%
+ Combined	91	8	91.9%
Video-Level	53 / 60	7 / 60	88.3%

Figure 8: Segmentation success rates across 99 graphics from 60 in-the-wild video clips (excluding 4 false positive detections and 1 false negative). The baseline pipeline uses VLM-guided detection with Segment-Anything. Each subsequent row shows the effect of applying a single repair strategy: interactive point editing (+6 graphics recovered) or text-aware segmentation refinement (+10). The combined row reflects both strategies applied together, raising the graphic-level success rate from 75.8% to 91.9%. At the video level, 53 of 60 clips (88.3%) are segmented successfully with combined repairs.



Figure 9: CARVid decomposition refinement interface where the user first gets a VLM-assisted segmentation which they can refine or use text or glyph refinements. They could also choose to manually segment with points, bounding boxes, and manual segmentation. They can change reference frame or add additional prompts to other frames.

5.1.1 Decomposition. Figure 8 summarizes the decomposition performance across the evaluation corpus. The pipeline detected 103 graphic elements across the 60 clips. Of these, 4 were false positive detections non-graphic regions incorrectly identified as overlays and 1 clip contained a graphic that the pipeline failed to detect entirely (a false negative), leaving 99 true graphics for segmentation evaluation.

We run the automatic VLM-guided decomposition pipeline on the raw input video from our corpus and store the output as a binary mask and a segmented video with a green-screen background. One author viewed each decomposed graphic side-by-side with the original video and recorded the following for each detected graphic segment: an overall verdict (success, failed, false positive, or false negative), a set of observed failure modes (text with holes, missing information, contains holes, translucent, false positive, did not detect graphic), and free-text comments describing the cause of

any degradation. For cases where failure could be attributed to a specific root cause, the author also assigned one or more semantic tags (e.g., low contrast, encoding artifacts, edge bleed, over-segmentation, non-opaque background). We define success as a downstream usable segmented result where there are no serious issues such as missing content or extreme flickering to the point where the information in the graphic cannot be conveyed. The author then progressively applies text refinements automatically and then proceeds to apply user refinements with our tool.

Results. Without any user intervention, the VLM-guided detection and SAM2 segmentation pipeline successfully decomposed 75 of 99 graphics (75.8%). Successful cases span all overlay categories and visual styles, including scoreboards, lower thirds, title cards, and corner bugs composited over diverse background content.

Repair strategies. Of the 24 baseline failures, 16 are recoverable through one of two complementary repair strategies. *Text-aware segmentation refinement* addresses the most prevalent failure mode: text elements with interior holes produced when SAM2 segments around fine typographic strokes rather than filling them. This strategy alone recovers 10 graphics, raising the success rate to 85.9%. *Interactive point editing*, in which the user adds or removes SAM2 prompt points to correct under- or over-segmentation, recovers an additional 6 graphics that suffered from incomplete mask coverage or extraneous content inclusion, raising the success rate to 81.8% when applied independently. Applying both strategies together raises the graphic-level success rate to 91.9% (91 of 99).

Video-level performance. We consider a clip successfully decomposed when every detected graphic in it is either correctly segmented at baseline or recoverable through a repair strategy. Under this criterion, 53 of 60 clips (88.3%) succeed with combined repairs. The 7 failing clips each contain at least one graphic whose translucency or contrast limitations place it beyond the reach of the current repair mechanisms.

Failure analysis. Among the 24 VLM-guided segmentation failures, the most common was text with interior holes (10/24), where SAM2 failed to produce solid text regions; all 10 were recoverable with HI-SAM text-aware refinement. The second was incomplete or over-extended coverage (6/24), where VLM-guided prompts missed parts of multi-component graphics or included background under low contrast; all 6 were recoverable with interactive point editing, though 2 also required replacing the initial detection. The remaining 8 were unfixable: 5 involved translucent overlays, whose blended pixels violate our binary foreground-background assumption and in 2 cases caused mask holes and temporal flicker; 3 arose from low contrast, compression artifacts, or fragmented text, causing boundary bleed, edge noise, or, in 1 plain-text graphic on a textured background, severe letter fragmentation. Please refer to the supplementary material for results.

5.1.2 Recomposition. To evaluate recomposition, we take the 75 successfully decomposed graphics from our corpus and run each through our automatic VLM-guided recomposition pipeline. For every graphic, the system first crops to the stable content bounding box, then recursively attempts to fit the result to a 605-pixel mobile-width target using four operations in priority order: (1)

scale, subject to a minimum 18-pixel text-height constraint; (2) vertical cut, where an energy-based candidate generator proposes cut positions and a VLM Critic selects or rejects each; (3) horizontal cut, following the same Candidate-Critic protocol; and (4) seam carving, which removes low-energy columns while respecting a text-padded safety mask. If none of these operations succeeds, the system falls back to a forced scale. Each operation and its outcome are logged per graphic, and one author reviews every result to assign a final pass/fail verdict, flagging cases where the pipeline cuts through semantic content, where the VLM misinterprets the layout, or where dynamic text prevents reliable cuts.

Results. We run the 75 decomposed graphics through our cut and recompose technique and perform automated tests if the segments match the target width. Of the 75 graphics, 72 pass recomposition (96.0%). 35 graphics (46.7%) require only a single scale operation to fit the target width, indicating that many broadcast graphics are narrow enough for mobile display after text-safe scaling alone. The remaining 40 graphics (53.3%) require at least one structural cut: vertical cuts account for the majority of layout splits (77 total v-cut invocations across 40 graphics, vs. 14 h-cut invocations across 11), confirming that horizontal graphic bars—the dominant broadcast layout—are most naturally divided along vertical boundaries. Seam carving is invoked rarely (6 calls across 3 graphics), acting as a last resort but a valuable tool when cuts are unsafe. The three failures fall into three categories: VLM misclassification (1 graphic where the VLM treats a patterned region as a solid color block and proposes an invalid cut), dynamic content (1 graphic with frame-to-frame text changes that invalidate static cut positions), and semantic cut errors (1 graphic where the Critic places a cut through foreground content).

6 LIMITATIONS AND FUTURE WORK

Our algorithm currently only outputs a flat mp4 of the graphical element, which limits the editability compared to other formats commonly used for motion graphics such as vectors or layers. Although our method can be made to work with layers, it is unclear how the workflow would look with multiple layers, styles, shapes, and text. In the future, a modification of our algorithm could support layered input and output to give the user direct control of properties such as editing, modifying, and deleting text, shapes, logos, etc. directly after the processing stage. This also allows the user to make edits after the algorithm returns a result. Such an extension would align CARVid more closely with real-world workflows in tools like After Effects, where users expect to iterate quickly and swap assets without re-rendering. Layered export would let designers not only reposition and refine elements but also round-trip assets back into professional pipelines, making CARVid function less like a one-shot filter and more like a co-creative system that preserves the designer’s ability to customize parameters at any stage.

Another limitation of CARVid is that the content-aware space removal technique does not generalize to graphics with rapidly changing text or animated content, since the Candidate-Critic pipeline

reasons over a static reference frame and cannot anticipate layout shifts between frames. Additionally, the VLM Critic can misclassify visually ambiguous regions for example, treating a textured or patterned area as empty space leading to cuts that destroy semantic content. Finally, our recursive decomposition into segments assumes graphics have roughly grid-like or bar-based layouts; graphics with non-rectangular, overlapping, or radially arranged elements may not admit clean vertical or horizontal cuts without information loss.

7 CONCLUSION

CARVid is a content-aware retargeting tool for videos with on-screen graphics that helps editors adapt graphics to new aspect ratios while preserving readability and original design intent. By combining graphics decomposition from flat videos, layout-aware segmentation, text-aware scaling, temporally coherent content-aware space removal, and layout reflow, CARVid recovers usable graphic structure and recomposes it for narrower formats. In our evaluation on a corpus of 60 in-the-wild videos, CARVid successfully decomposed 91% of graphic overlays, and 96% of decomposed graphics fit the target width after recomposition. While current limitations remain around dense multi-line text, translucent overlays, and flat-only exports, future support for layered outputs and parametric customization could better align CARVid with professional motion-graphics workflows.

REFERENCES

- [1] Adobe Systems. 2025. Auto Reframe in Adobe Premiere Pro. <https://helpx.adobe.com/premiere/desktop/add-video-effects/commonly-used-effects/add-auto-reframe-effect-to-a-sequence.html>.
- [2] Apple Inc. 2024. Typography — Apple Human Interface Guidelines. <https://developer.apple.com/design/human-interface-guidelines/typography>. Accessed 2025-08-15.
- [3] Shai Avidan and Ariel Shamir. 2007. Seam carving for content-aware image resizing. In *ACM SIGGRAPH 2007 Papers* (San Diego, California) (SIGGRAPH '07). Association for Computing Machinery, New York, NY, USA, 10–es. <https://doi.org/10.1145/1275808.1276390>
- [4] Jiawang Cao, Yongliang Wu, Weiheng Chi, Wenbo Zhu, Ziyue Su, and Jay Wu. 2024. Reframe Anything: LLM Agent for Open World Video Reframing. arXiv:2403.06070 <https://arxiv.org/abs/2403.06070>
- [5] Dabing Cheng, Haosen Zhan, Xingchen Zhao, Guisheng Liu, Zemin Li, Jinghui Xie, Zhao Song, Weiguo Feng, and Bingyue Peng. 2025. Text-to-Edit: Controllable End-to-End Video Ad Creation via Multimodal LLMs. arXiv:2501.05884 <https://arxiv.org/abs/2501.05884>
- [6] Donghyeon Cho, Yunjae Jung, Francois Rameau, Dahun Kim, Sanghyun Woo, and In So Kweon. 2019. Video Retargeting: trade-off between content preservation and spatio-temporal consistency. In *Proceedings of the 27th ACM international conference on multimedia*. 882–889. <https://doi.org/10.1145/33433031.3350895>
- [7] CNBC. 2026. Uber to invest up to \$1.25 billion in EV maker Rivian in deal to launch 50,000 robotaxis. <https://www.youtube.com/watch?v=nbBpmgRiCTM>. YouTube video, published Mar 19, 2026.
- [8] DeepMind. 2025. Gemini. <https://deepmind.google/models/gemini/>. Accessed: 2025-09-11.
- [9] Nathan Frey and Zheng Sun. 2020. AutoFlip: An Open Source Framework for Intelligent Video Reframing. Google Research Blog. <https://research.google/blog/autoflip-an-open-source-framework-for-intelligent-video-reframing/>. Senior Software Engineers, Google Research, Los Angeles and Mountain View.
- [10] Google. n.d.. Google Material Design Guidelines. <https://m2.material.io/design/typography/>. Accessed: 2025-09-10.
- [11] Google DeepMind. 2025. Veo 3 — Generative Video Model. <https://aistudio.google.com/models/veo-3>. [Accessed 10-09-2025].
- [12] Google LLC. 2026. Google Gemini AI. <https://gemini.google.com/>. Accessed: 2026-04-01.
- [13] Bernd Huber, Hujung Valentina Shin, Bryan Russell, Oliver Wang, and Gautham J. Mysore. 2019. B-Script: Transcript-based B-roll Video Editing with Recommendations. In *Proceedings of the 2019 CHI Conference on Human Factors*

- in *Computing Systems* (Glasgow, Scotland Uk) (CHI '19). Association for Computing Machinery, New York, NY, USA, 1–11. <https://doi.org/10.1145/3290605.3300311>
- [14] Jaied AI. 2020. EasyOCR — Ready-to-use OCR with 80+ Languages Supported. <https://github.com/JaiedAI/EasyOCR>. [Accessed 10-09-2025].
- [15] Johannes Kiess, Benjamin Guthier, Stephan Kopf, and Wolfgang Effelsberg. 2012. SeamCrop: changing the size and aspect ratio of videos. In *Proceedings of the 4th Workshop on Mobile Video* (Chapel Hill, North Carolina) (MoVid '12). Association for Computing Machinery, New York, NY, USA, 13–18. <https://doi.org/10.1145/2151677.2151681>
- [16] Jeongyeon Kim, Yubin Choi, Minsuk Kahng, and Juho Kim. 2022. FitVid: Responsive and Flexible Video Content Adaptation. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) (CHI '22). Association for Computing Machinery, New York, NY, USA, Article 501, 16 pages. <https://doi.org/10.1145/3491102.3501948>
- [17] Jeongyeon Kim, Yubin Choi, Meng Xia, and Juho Kim. 2022. Mobile-friendly content design for MOOCs: challenges, requirements, and design opportunities. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–16. <https://doi.org/10.1145/3491102.3502054>
- [18] Jon Krasner. 2013. *Motion graphic design: applied history and aesthetics*. Routledge. <https://www.routledge.com/Motion-Graphic-Design-Applied-History-and-Aesthetics/Krasner/p/book/9780240821139>
- [19] Mackenzie Leake, Abe Davis, Anh Truong, and Maneesh Agrawala. 2017. Computational video editing for dialogue-driven scenes. *ACM Trans. Graph.* 36, 4, Article 130 (July 2017), 14 pages. <https://doi.org/10.1145/3072959.3073653>
- [20] Mackenzie Leake and Wilmot Li. 2024. ChunkyEdit: Text-first video interview editing via chunking. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) (CHI '24). Association for Computing Machinery, New York, NY, USA, Article 1020, 16 pages. <https://doi.org/10.1145/3613904.3642667>
- [21] Jianan Li, Jimei Yang, Jianming Zhang, Chang Liu, Christina Wang, and Tingfa Xu. 2020. Attribute-Conditioned Layout GAN for Automatic Graphic Design. *IEEE Transactions on Visualization and Computer Graphics* 27 (2020), 4039–4048. <https://doi.org/10.1109/TVCG.2020.2999335>
- [22] Vivian Liu, Rubaiat Habib Kazi, Li-Yi Wei, Matthew Fisher, Timothy Langlois, Seth Walker, and Lydia Chilton. 2025. LogoMotion: Visually-Grounded Code Synthesis for Creating and Editing Animation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems* (CHI '25). Association for Computing Machinery, New York, NY, USA, Article 157, 16 pages. <https://doi.org/10.1145/3706598.3714155>
- [23] Jiaju Ma and Maneesh Agrawala. 2025. MoVer: Motion Verification for Motion Graphics Animations. *ACM Trans. Graph.* 44, 4, Article 33 (July 2025), 17 pages. <https://doi.org/10.1145/3731209>
- [24] NBA. 2025. Scottie Barnes with the tough rejection! <https://www.youtube.com/shorts/6wxQLAo3JA>. YouTube video, published Mar 8, 2025.
- [25] NFL. 2026. And he stayed in bounds! <https://www.youtube.com/watch?v=cgylx1gBxyk>. YouTube video, published Mar 1, 2026.
- [26] OpenAI. 2024. Video generation models as world simulators. <https://openai.com/index/video-generation-models-as-world-simulators/>
- [27] Opusclip Inc. [n. d.]. OpusClip. <https://www.opus.pro/>. Accessed: 2026-04-01.
- [28] Nobuyuki Otsu. 1979. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics* 9, 1 (1979), 62–66. <https://doi.org/10.1109/TSMC.1979.4310076>
- [29] Yael Pritch, Eitam Kav-Venaki, and Shmuel Peleg. 2009. Shift-map image editing. In *2009 IEEE 12th international conference on computer vision*. IEEE, 151–158. <https://doi.org/10.1109/ICCV.2009.5459159>
- [30] Dongjun Qian, Kai Su, Yiming Tan, Qishuai Diaio, Xian Wu, Chang Liu, Bingyue Peng, and Zehuan Yuan. 2025. VC-LLM: Automated Advertisement Video Creation from Raw Footage using Multi-modal LLMs. [arXiv:2504.05673](https://arxiv.org/abs/2504.05673) <https://arxiv.org/abs/2504.05673>
- [31] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. 2024. Sam 2: Segment anything in images and videos. [arXiv:2408.00714](https://arxiv.org/abs/2408.00714) <https://arxiv.org/abs/2408.00714>
- [32] Michael Rubinstein, Diego Gutierrez, Olga Sorkine, and Ariel Shamir. 2010. A comparative study of image retargeting. *ACM Trans. Graph.* 29, 6, Article 160 (Dec. 2010), 10 pages. <https://doi.org/10.1145/1882261.1866186>
- [33] Michael Rubinstein, Ariel Shamir, and Shai Avidan. 2008. Improved seam carving for video retargeting. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 1–9. <https://doi.org/10.1145/1360612.1360615>
- [34] Michael Rubinstein, Ariel Shamir, and Shai Avidan. 2009. Multi-operator media retargeting. *ACM Transactions on graphics (TOG)* 28, 3 (2009), 1–11. <https://doi.org/10.1145/1531326.1531329>
- [35] Wenda Shi, Yiren Song, Dengming Zhang, Jiaming Liu, and Xingxing Zou. 2025. FontTS: Text Rendering With Typography and Style Controls. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 18463–18474.
- [36] Than Htut Soe and Marija Slavkovic. 2022. A content-aware tool for converting videos to narrower aspect ratios. In *Proceedings of the 2022 ACM International Conference on Interactive Media Experiences*. 109–120. <https://doi.org/10.1145/3505284.3529970>
- [37] Bekzat Tilekbay, Saelyne Yang, Michal Adam Lewkowicz, Alex Suryapranata, and Juho Kim. 2024. ExpressEdit: Video Editing with Natural Language and Sketching. In *Proceedings of the 29th International Conference on Intelligent User Interfaces* (Greenville, SC, USA) (IUI '24). Association for Computing Machinery, New York, NY, USA, 515–536. <https://doi.org/10.1145/3640543.3645164>
- [38] Bryan Wang, Yuliang Li, Zhaoyang Lv, Haijun Xia, Yan Xu, and Raj Sodhi. 2024. LAVE: LLM-Powered Agent Assistance and Language Augmentation for Video Editing. In *Proceedings of the 29th International Conference on Intelligent User Interfaces* (Greenville, SC, USA) (IUI '24). Association for Computing Machinery, New York, NY, USA, 699–714. <https://doi.org/10.1145/3640543.3645143>
- [39] Sitong Wang, Zheng Ning, Anh Truong, Mira Dontcheva, Dingzeyu Li, and Lydia B Chilton. 2024. PodReels: Human-AI Co-Creation of Video Podcast Teasers. In *Proceedings of the 2024 ACM Designing Interactive Systems Conference* (Copenhagen, Denmark) (DIS '24). Association for Computing Machinery, New York, NY, USA, 958–974. <https://doi.org/10.1145/3643834.3661591>
- [40] Yu-Shuen Wang, Jen-Hung Hsiao, Olga Sorkine, and Tong-Yee Lee. 2011. Scalable and coherent video resizing with per-frame optimization. In *ACM SIGGRAPH 2011 Papers* (Vancouver, British Columbia, Canada) (SIGGRAPH '11). Association for Computing Machinery, New York, NY, USA, Article 88, 8 pages. <https://doi.org/10.1145/1964921.1964983>
- [41] Yu-Shuen Wang, Hui-Chih Lin, Olga Sorkine, and Tong-Yee Lee. 2010. Motion-based video retargeting with optimized crop-and-warp. *ACM Trans. Graph.* 29, 4, Article 90 (July 2010), 9 pages. <https://doi.org/10.1145/1778765.1778827>
- [42] Yu-Shuen Wang, Chiew-Lan Tai, Olga Sorkine, and Tong-Yee Lee. 2008. Optimized scale-and-stretch for image resizing. In *ACM SIGGRAPH Asia 2008 papers*. 1–8. <https://doi.org/10.1145/1409060.1409071>
- [43] Lior Wolf, Moshe Guttman, and Daniel Cohen-Or. 2007. Non-homogeneous content-driven video-retargeting. In *2007 IEEE 11th international conference on computer vision*. IEEE, 1–6. <https://doi.org/10.1109/ICCV.2007.4409010>
- [44] Joshua Kong Yang, Mackenzie Leake, Jeff Huang, and Stephen DiVerdi. 2025. VidSTR: Automatic Spatiotemporal Retargeting of Speech-Driven Video Compositions. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–17. <https://doi.org/10.1145/3706598.3713857>
- [45] Maoyuan Ye, Jing Zhang, Juhua Liu, Chenyu Liu, Baocai Yin, Cong Liu, Bo Du, and Dacheng Tao. 2024. Hi-SAM: Marrying Segment Anything Model for Hierarchical Text Segmentation. [arXiv:2401.17904](https://arxiv.org/abs/2401.17904) [cs.CV] <https://arxiv.org/abs/2401.17904>
- [46] Di Zhang, Junxian Li, Jingdi Lei, Xunzhi Wang, Yujie Liu, Zonglin Yang, Jiaotong Li, Weida Wang, Suorong Yang, Jianbo Wu, Peng Ye, Wanli Ouyang, and Dongzhan Zhou. 2025. Critic-V: VLM Critics Help Catch VLM Errors in Multimodal Reasoning. [arXiv:2411.18203](https://arxiv.org/abs/2411.18203) [cs.CV] <https://arxiv.org/abs/2411.18203>
- [47] Peirong Zhang, Haowei Xu, Jiaxin Zhang, Guitao Xu, Xuhan Zheng, Zhenhua Yang, Junle Liu, Yuyi Zhang, and Lianwen Jin. 2025. Aesthetics is Cheap. Show me the Text: An Empirical Evaluation of State-of-the-Art Generative Models for OCR. [arXiv preprint arXiv:2507.15085](https://arxiv.org/abs/2507.15085) (2025). <https://arxiv.org/abs/2507.15085>
- [48] Sharon Zhang, Jiaju Ma, Jiajun Wu, Daniel Ritchie, and Maneesh Agrawala. 2023. Editing Motion Graphics Video via Motion Vectorization and Transformation. *ACM Trans. Graph.* 42, 6, Article 229 (Dec. 2023), 13 pages. <https://doi.org/10.1145/3618316>
- [49] Xinru Zheng, Xiaotian Qiao, Ying Cao, and Rynson W. H. Lau. 2019. Content-aware generative modeling of graphic design layouts. *ACM Trans. Graph.* 38, 4, Article 133 (July 2019), 15 pages. <https://doi.org/10.1145/3306346.3322971>
- [50] Xingxing Zou, Wen Zhang, and Nanxuan Zhao. 2025. From Fragment to One Piece: A Review on AI-Driven Graphic Design. *Journal of Imaging* 11, 9 (2025). <https://doi.org/10.3390/jimaging11090289>

APPENDIX A

8 AUTOMATIC RETARGETING

8.0.1 Graphics Pre-Preprocessing. The first step in our pipeline, given an input video of graphics that need to be retargeted, is to compute an *energy map* E_f for each frame, which is a per-pixel score on the frame luminance (Figure 10). We use it to rank columns for deletion. This is a common practice in image/video processing and resizing methods like seam carving where the energy is the orientation-agnostic Sobel gradient magnitude $e(I_f) = \left| \frac{\partial I_f}{\partial x} \right| + \left| \frac{\partial I_f}{\partial y} \right|$, where I_f is the luminance of frame f . Classic retargeting methods (e.g., seam carving) score pixels with an orientation-agnostic gradient and remove pixels around *all* edges [3, 33]. This works for natural images where uneven removal is masked by texture. On-screen

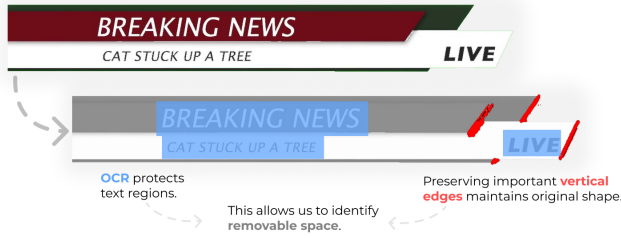


Figure 10: A frame of a graphical element annotated with high energy regions, which are preserved during reframing. Optical character recognition (OCR) is used to identify text bounding boxes (blue). Gradient filtering marks pixels that are part of strong edges (red) that are important for maintaining the visual identity.

graphics, however, sit over simple, non-salient backgrounds with little texture; design preservation (shapes, content, and the overall structure) is paramount and our goal is to sacrifice space while preserving the most important design elements [18].

Uneven removal or trimming near vertical edges visibly distorts a design layout. As a result, our method operates on the principle of achieving width reduction by deleting vertical columns, we ignore horizontal edges and never incentivize cutting through vertical edges by modifying the energy computation to the horizontal image derivative only, i.e., $e(I) = \left| \frac{\partial I}{\partial x} \right|$. Moreover, elements like logos and QR codes should neither be removed nor altered; by focusing on vertical edges and protecting the space around them, we can identify these high-importance regions while still leaving low-importance areas elsewhere for removal. For each frame I_f , we compute $\hat{E}_f = e(I_f)$, and retain only the pixels with energy in the top 90 percentile $\tau_f = \text{quantile}(\hat{E}_f, 0.9)$ and zero out values below τ_f , obtaining $\hat{V}_f$ to suppress weak texture. We use OCR to identify text region bounding boxes. These regions are incorporated into our energy function as high-energy regions by adding the temporally stabilized OCR mask O_f with a large constant M , so those pixels are given a high energy ($E_f = \hat{V}_f + M O_f$). Finally, we compute per-column scores by vertical summation $\phi_f(x) = \sum_y E_f(x, y)$, which downstream stages use to decide which regions can be safely removed. We provide an example of this computation in figure 10.

8.1 Text-Aware Scaling

We first attempt uniform scaling while maintaining text readability. We operationalize a minimum legible height $h_{\min}$ as 16px following prior literature on readability [2, 10, 17] and compute the smallest legible text region in the video h_{text} . The graphic is scaled uniformly until $h_{\text{text}} \geq h_{\min}$ is no longer satisfied; if the target size is reached first, we export, otherwise we proceed to content-aware space removal.

8.2 Content-Aware Space Removal

While computing energy maps provides a rough and wide estimate of where empty or low-importance regions exist, the animated nature of graphics videos means that these regions fluctuate across

time: a column that looks available in one frame may be crossed by text or a moving logo in the next. Because our operation is width reduction by removing vertical columns, selecting columns independently in each frame would introduce jitter and break the animation. To preserve temporal coherence, we instead track one column consistently across frames so that removal preserves the animation and keeps it stable and aligned with the motion of the video. We achieve this by defining a temporal consistency objective with three terms: an energy avoidance objective, a lookahead objective with bandwidth, and a motion (x-direction) objective.

The **energy avoidance objective** penalizes columns near and around high-energy content by summing the per-column scores, so that columns containing or close to text, logos, or vertical borders are never removed. The **lookahead objective with bandwidth constraint** enforces stability by restricting the next column choice to a narrow band around the previous one and adding a penalty proportional to displacement, which discourages sudden horizontal jumps. The **motion (x-direction) objective** uses optical flow to measure horizontal displacement along candidate columns; if $u_f(x)$ is the average optical flow magnitude near column x in frame f , then columns intersecting scrolling text or sliding graphics are penalized and only persistently empty areas are removed. Together these terms define the temporal consistency optimization across the column path $x_{1:F}$, with eligible low-importance regions S_f and protected no-go regions N_f identified per frame, constrained by bandwidth parameter B :

$$\begin{aligned} \min_{x_{1:F}} \quad & C_{\text{energy}} + C_{\text{lookahead}} + C_{\text{motion}} \\ C_{\text{energy}} = \quad & \sum_f \phi_f(x_f) \\ C_{\text{lookahead}} = \quad & \sum_f |x_f - x_{f-1}| \\ C_{\text{motion}} = \quad & \sum_f u_f(x_f) \end{aligned} \quad (1)$$

$$\text{subject to } x_f \in S_f, \quad x_f \notin N_f, \quad |x_f - x_{f-1}| \leq B$$

This tracks a single coherent vertical column across the all frames. After one column is removed (from all frames), we update the energy maps and eligibility masks and repeat until the target width is reached or no consistent path remains. This process eliminates only regions that remain persistently low-importance while preserving straight borders, alignment, and the temporal coherence of on-screen graphics.

8.3 Layout Reflow

When neither scaling nor seam carving suffice, we reflow the graphic into multiple lines. We compute a composite energy map across sampled frames. A candidate cut position is then chosen as the column with low average energy but near a high-energy vertical edge, which typically aligns with a natural divider. If no cut satisfies this constraint (e.g., all candidates intersect dense text), reflowing is not performed.

To reflow the layout, we search for a vertical cut that avoids important regions while still reducing width. We first use the frame energies to build a composite energy map $\bar{E} = \frac{1}{|S|} \sum_f E_f$, which

smooths temporal fluctuations and highlights persistently important areas. Columns with high composite values typically contain vertical edges, logos, or text, while columns with low values represent candidate cut sites.

The search space excludes trivial edges near the boundaries, columns above a high-percentile energy threshold, and columns where OCR masks indicate persistent text across frames. This yields a reduced set of eligible cut positions.

Each candidate column is then scored according to four weighted terms: the **text avoidance score** $s_{\text{text}}(x)$, which penalizes overlaps with persistent text and rewards text-free regions; the **energy**

score $s_{\text{energy}}(x)$, which prefers low-energy columns; the **edge alignment score** $s_{\text{edge}}(x)$, which rewards alignment with existing vertical structures; and the **center preference score** $s_{\text{center}}(x)$, which mildly favors balanced left-right sections. These combine into a single ranking where the optimal cut position x^* is defined with custom weights for each:

$$x^* = \arg \max (w_t s_{\text{text}}(x) + w_e s_{\text{energy}}(x) + w_a s_{\text{edge}}(x) + w_c s_{\text{center}}(x)) \quad (2)$$

Once the best column is chosen, the frame is cut at this position and the right portion is shifted leftward so it sits directly below the left side of the cut, producing a seamless reflow.