VidSTR: Automatic SpatioTemporal Retargeting of Speech-Driven Video Compositions

Joshua Kong Yang
joshua_yang@brown.edu
Brown University
Providence, RI, USA

Mackenzie Leake
leake@adobe.com
Adobe Research
San Francisco, CA, USA

Jeff Huang
jeff_huang@brown.edu
Brown University
Providence, RI, USA

Stephen DiVerdi
diverdi@adobe.com
Adobe Research
San Francisco, CA, USA

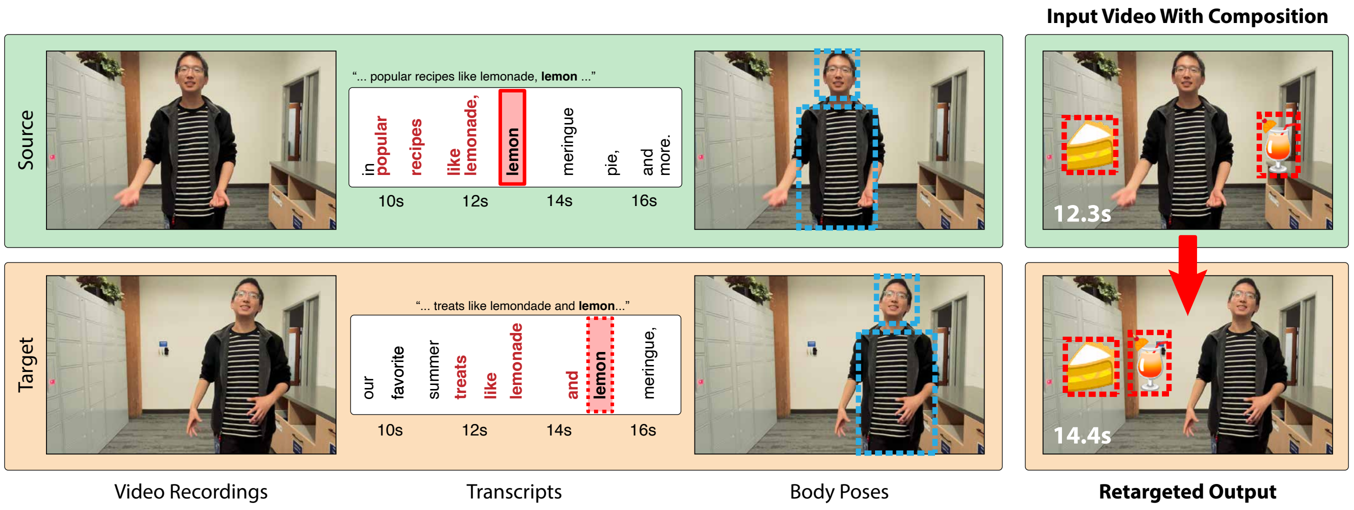

Input Video With Composition
Source
"... popular recipes like lemonade, **lemon** ..."
in popular recipes like lemonade, lemon meringue pie, and more.
10s 12s 14s 16s
12.3s
Target
"... treats like lemondade and **lemon**..."
our favorite summer treats like lemonade and lemon meringue,
10s 12s 14s 16s
14.4s
Video Recordings
Transcripts
Body Poses
Retargeted Output

Figure 1: In video composition retargeting, graphical assets placed on video are updated temporally and spatially to match a new video. Left: The source and target videos are two recordings of similar performances, which are characterized by their transcripts and body poses (used to identify face and torso bounding boxes). Upper right: A user adds graphics to the source video at specific times to emphasize their speech and chooses locations that are aesthetically pleasing. Lower right: When the composition is automatically retargeted to the target video, the timing of graphics is updated based on transcript correspondence, and the location is updated based on optimization with respect to the body poses.

Abstract

Video editors often record multiple versions of a performance with minor differences. When they add graphics atop one video, they may wish to transfer those assets to another recording, but differences in performance, wordings, and timings can cause assets to no longer be aligned with the video content. Fixing this is a time-consuming, manual task. We present a technique which preserves the temporal and spatial alignment of the original composition when automatically retargeting speech-driven video compositions. It can transfer graphics between both similar and dissimilar performances, including those varying in speech and gesture. We use a large language model for transcript-based temporal alignment and integer programming for spatial alignment. Results from retargeting between 51 pairs of performances show that we achieve a temporal alignment success rate of 90% compared to hand-generated ground truth compositions. We demonstrate challenging scenarios, retargeting video compositions across different people, aspect ratios, and languages.

CHI '25, Yokohama, Japan
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1394-1/25/04
https://doi.org/10.1145/3706598.3713857

CCS Concepts

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

Keywords

video, composition, motion graphics, temporal alignment, spatial alignment

ACM Reference Format:
Joshua Kong Yang, Mackenzie Leake, Jeff Huang, and Stephen DiVerdi. 2025. VidSTR: Automatic SpatioTemporal Retargeting of Speech-Driven Video Compositions. In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26-May 1, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3706598.3713857

1 Introduction

Digital video editing is a multi-stage process. For many scripted videos, a video creator records an actor's performance, possibly multiple times to get the best recording. These different performances can capture different phrasings, pacings, gestures, and vocal and visual performance qualities. Once the creator has recorded a good version of their video, they may add additional effects, such as graphical overlays. These overlays are often images and text that illustrate or emphasize the performance, which therefore need to be aligned both temporally and spatially with the content. For example, when an actor describes lemon pie, an image of that object may appear above their hand gesture (Fig. 1). After carefully placing assets, the creator exports this single, final video composition (i.e., video with these overlays).

The linear process of creating a video composition makes it difficult to go back and make changes to the underlying video without having to redo the work of aligning graphical overlays with the performance. In practice, there are many reasons a video creator would want to substitute a different performance recording. For example, they may not be able to fully determine a performance's quality until seeing it in the full composition. They may discover that they should use a different performance or adjust performance cues and re-record the video with the actor. In other cases, they may want to produce multiple versions of the final video. These could be similar video compositions with different performers, possibly even in different languages for different target markets. They may also need to produce multiple video formats (i.e., vertical, horizontal, square) to target different social media channels. Performing any of these edits would require significant time and effort with current digital video editing tools.

We present VidSTR, a video editing tool that supports automatic video composition retargeting, the act of transferring graphical overlays from one video to another, while maintaining temporal and spatial alignment with respect to the video content. To our knowledge, moving such edits *between different video takes* automatically has not been done before. VidSTR can adapt the timing of overlays to non-uniform changes in the performer's speed, while also being robust to differences in the precise speech and gestures performed. We can also adapt the spatial placement of overlays to avoid overlapping the performer as they move to different positions at different times throughout a new video.

We focus on videos in which a single performer gives a speech-centric performance, such as a vlog, video podcast, lecture, or presentation. VidSTR uses a large language model (LLM) for transcript-based temporal alignment and integer linear programming for spatial alignment. To evaluate our algorithm, we measure the error rates for temporal and spatial alignment of transferred graphics to videos with hand-generated ground truth compositions, and show that we achieve error rates significantly lower than baseline approaches, saving time and effort for the video editors.

The main contribution of our work is our algorithm for retargeting video compositions. We implement it in VidSTR through a simple video editing GUI, which we use to demonstrate different applications of the technique, such as retargeting between similar videos (different takes of the same actor performing the same script), dissimilar videos (the same actor performing variations of the script), across different actors performing the same script, across different aspect ratios (e.g., landscape to portrait), and across different languages (e.g., a source video in English and a target video in Brazilian Portuguese). The result is a system that can save video creators significant time and effort while enabling new workflows for video editing.

2 Related Work

Our work builds upon prior work on supporting video editing, time aligning visual assets, and automatic layout. We apply and extend ideas from these three areas to the specific challenges that arise in retargeting video compositions.

2.1 Video editing tools

Many computational tools have been developed for using different modalities to assist with specific parts of the video editing process. Recently, tools have explored the use of natural language for video editing. For example, LAVE uses a LLM-powered agent to provide natural language directions for several common video editing steps, such as brainstorming and storyboarding as well as clip trimming [59]. ExpressEdit provides a natural language and sketch-based interface for specifying edits for informational videos [53]. Tools that focus on the early stages of video editing include ReelFramer, which uses an LLM to translate print news articles into storyboards for short social media videos [61] and ChunkyEdit, which uses LLMs and topic modeling to automatically group interview video clips thematically for pre-editing [34]. More broadly, computational video editing has been applied to various genres of video, such as first person narrative videos [2], 360-degree conversational videos [54], tutorials [12, 56], and informational videos from websites [11, 13]. Prior work has focused on automating the assembly of scripted dialogue-based scenes using a probabilistic editing model to select among repeated takes [33] and of narrated videos using audio-aligned transcripts and spoken annotations [55]. Other prior work has focused on the selection of b-roll videos [24] and other visual assets to display during spoken narration for informational videos [35], travel podcasts [64], live video meetings [37], and presentations [46]. In this work we do not suggest specific visuals or edited video sequences, but rather we realign the user's chosen assets to new performances or takes.

2.2 Automatic video alignment

A variety of approaches for aligning video content via visual features and the transcript have been explored. VideoSnapping automatically temporally aligns multiple videos using a graph-based algorithm for detecting frames with similar visual features [60]. Several tools have leveraged the alignment between the audio dialogue in a video and its text transcript. Rubin et al. [45] extended the Penn Forced Aligner (P2FA) [65], which uses a statistical approach to align audio phonemes with a text transcript. Several computational

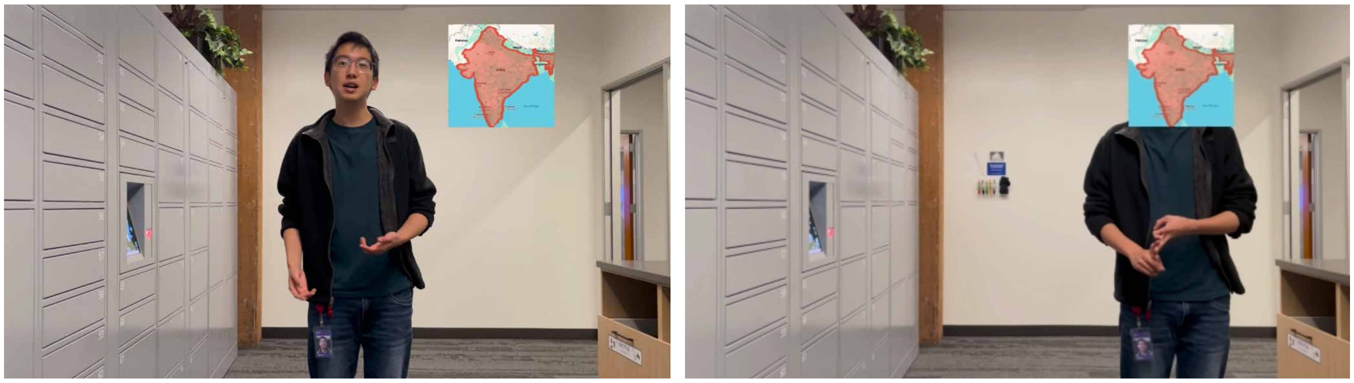


Figure 2: Without effective retargeting, substituting a new video into a composition often yields bad results. On the left we see a video with a user-added map that matches the content and timing of their speech. On the right a new video is substituted with the performer in a different position. The graphical overlay now obscures the face, and the speech timing is different so the graphic appears when a different, unrelated word is spoken. Automatic video composition retargeting addresses this problem.

video editing and review tools (e.g., [33, 42, 43, 55]), build on top of this technology and use string edit distance to support alignment of transcripts with small deviations across performances or mismatches between the audio and script. Our system extends this line of work in finding correspondences between media content and remapping across different performances. In addition to supporting mappings between videos with varied pacing and slight variations in wording, we use an LLM to find mappings between content that varies more broadly between videos, such as when two sections of content have their word order swapped entirely or semantically similar words and phrases are introduced.

2.3 Auto-layout for visual content

Various domain-specific approaches have been developed for the spatial layout of visual content. There has been a long line of work specifically on optimizing GUI and web layouts, such as Cassowary [6] which uses linear programs and constrained optimization derivatives, and preventing overlap between UI components [66]. Notably, GRIDS [15] uses the mixed integer linear program paradigm to make the problem formulation of GUI layout easier to solve. More recent work has explored how to use integer programming to transfer layouts with one set of constraints to another with different constraints [14]. Another line of work has explored using SAT solvers to define optimal packing layouts [49] and applying the more-general SMT solvers to UI layouts [26, 51]. Some AR tools frame object placement as a constraint satisfaction problem [19], while others have been built on top of existing spatial layout solvers, such as MRTK, for placing objects within a scene subject to user-defined constraints [40]. For video, layout guidance has come in the form of story templates to guide capture [31] or pre-defined layout templates to support the creation of a video from a document [10] or desktop to mobile viewing of educational content [30]. In our work, we apply an integer linear programming approach for laying out visual assets on top of a video, rather than using pre-defined templates.

Another area of prior work has sought to address the challenge of retargeting images and videos to different aspect ratios. A classical technique for images [5] and videos [29] is seam carving, which preserves important visual features while minimizing distortion by removing pixels in lower energy regions of the image or frame to achieve a target aspect ratio. Guo et al. [21] set up image resizing as a constrained image mesh parameterization problem to minimize distortion when resizing images. Other work relies on content detection to intelligently reduce the width of the frame [48] or eye gaze to select the different, key regions of the video to show on different displays [25]. In our work, we instead focus on retargeting the assets from a video composition, deciding where to place the assets in a way that does not occlude important parts of a new underlying video, such as the speaker's face. This process performs aspect ratio retargeting when the new video has different dimensions than the source video.

3 Video Composition Retargeting

We define a "video composition" as a document in any digital editing video tool (e.g., Adobe Premiere or CapCut), comprising a single video with multiple graphical elements placed on top, appearing at times and in positions that are "aligned" with the content of the video. For example, when a performer in a video says the word "lemon" in the phrase "lemon meringue pie," an illustrative image of such a pie briefly appears next to them (Fig. 1).

VidSTR primarily focuses on videos that are speech-driven and has a single performer. There are many such types of videos on social media today, such as vlogs, livestreams, lectures, product or movie reviews, etc. Such video compositions are typically created in standard, commercial digital video editing tools. If the creator wishes to use a different underlying video (e.g., another recording of the performer with clearer speech, fewer distractions, better timing, etc., or even a recording of a different performer) for the composition, they must replace the original video with the new video, which causes all of the graphical elements' times and locations to become misaligned in the new video because the new video's cadence and composition will be different (Fig. 2). Video composition

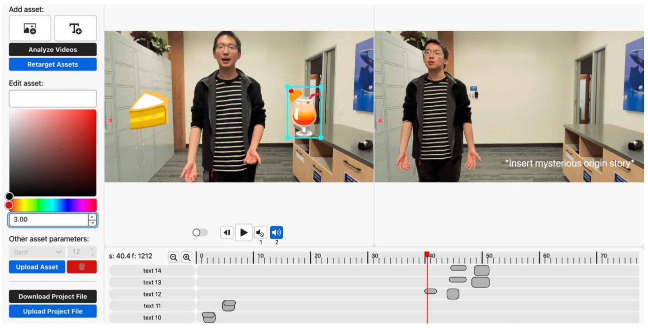


Figure 3: In the VɪᴅSTR UI, the original video is displayed in the left panel. The user adds initial graphics and adjusts their time and position. Timeline tracks show the timing of the graphics (taller, darker gray shapes). A second video take is displayed in the right panel. After retargeting, the time and position of the graphics in the second video are adjusted based on the first video to be aligned with its performance. The shorter gray bar on the timeline tracks show the retargeted asset timings.

"retargeting" is the act of adjusting the times and locations of the graphical elements so they are well-aligned with the new video.

In order to explore video composition retargeting algorithms, we developed VɪᴅSTR (Fig. 3). The user imports a video in the left panel and imports graphics to place on top, using the timeline to temporally align the graphics and the video panel to spatially position them within the frame. Then, the user imports a second video to the right panel, and with a single click, the graphics from the left video are applied to the right video at the appropriate locations and times. This saves the editor from having to manually realign the assets in the right video, as they would typically have to do in a standard video editing tool.

4 Algorithm

A video composition is defined as a video and a number of *events* consisting of when and where graphical assets appear. Retargeting is applying a sequence of events from one video onto another video, updating the times and locations of those events to match the new video. Our algorithm is composed of two distinct steps: *temporal retargeting*, where the times of events are adjusted, and *spatial retargeting*, where the location of events are adjusted.

The source and target videos are characterized entirely by their transcripts and tracked body poses. These are computed once when a video is imported into VɪᴅSTR. Our retargeting algorithm operates only on these data and does not consider video pixels further.

4.1 Temporal Retargeting

We have a set of assets $A = \{a_1, \ldots, a_n\}$ that appear on the screen at times t_i in the source video. Our goal is to find the retargeted times t'_i in the target video, i.e. when a_i should appear in the target video. We align times based on finding matching text in the source and target video transcripts.

A standard approach for transcript alignment is dynamic time warping [17], which addresses a broader problem than ours. It must find alignments for every single word, and is more constrained than our problem, as all words must appear in the same order. Neither applies in our case because we only need to find sparse correspondences for words around graphical events, and they may be ordered differently. If a performance differs enough, the same precise words may not even appear, replaced with semantically similar words. Dynamic time warping and other edit distance based approaches can fail in this common scenario because they rely on having more similar words and syntax. Therefore, we use a large language model (LLM) for alignment.

First, we compute transcripts for the source and target videos, and provide them in full to the LLM as TRANSCRIPT_OLD and TRANSCRIPT_NEW respectively. For each asset a_i we find the closest spoken word $<w_i>$, which is offset from a_i by $o_i = t_i - t_{w_i}$, and its containing phrase $<p_i>$ before prompting the LLM with "*In TRANSCRIPT_OLD, I have an asset <i> that appears on the word $<w_i>$ in the phrase $<p_i>$*" for all i sequentially. The assets a_i should be sorted by their appearance time against TRANSCRIPT_OLD. Then we prompt the LLM to find matches by content similarity: "*Please tell me on which word in TRANSCRIPT_NEW should the asset <i> appear in the new video take, considering the context of where it came from, and the exact phrase it is located in*" for all i sequentially. Finally we further prompt the LLM to post-process its output so we can perform reliable, unambiguous string matching against the raw TRANSCRIPT_NEW to find the final correspondences:

> *Make sure the phrases are at least six words long. Make sure the phrases contain their respective found words so we can find them with string matching in the future. If two identical words appear at different places in the same phrase, please crop the first word's phrase from the end and crop the second word's phrase from the start so we can differentiate them later.*

We set the LLM temperature relatively low, to $T = 0.2$, in order to achieve more consistent results. After prompting, it returns a set of new words w'_i and phrases p'_i. We match p'_i with TRANSCRIPT_NEW to find the unique aligned word w'_i and its timestamp. The final predicted time of the retargeted asset is $t'_i = t'_{w'_i} + o_i$. Repeating for all i yields the new times of all assets in the target video, $\{t'_i\}_{i=1}^{n}$, which is the output of the temporal alignment step.

4.2 Spatial Retargeting

Spatial retargeting is necessary when the underlying visual content of the video changes, namely when regions of interest (ROI) move and avoiding occlusion of these ROIs is desired. We now describe an algorithm to compute new positions (x_a, y_a) of the assets $a \in A$. We define the regions of interest as the bounding boxes of the face and torso from the body tracking data. For each dynamic ROI, we compute the mean bounding box across all on-screen frames. We also compute the static bounding box for each asset that may need repositioning. Our system supports static assets that do not move or change size (unlike the ROIs which do move and change in size).

Our primary goal when repositioning assets is to preserve as many of the placement decisions from the user while preventing overlap between different assets and between assets and certain underlying parts of the video. An asset a requires repositioning if in

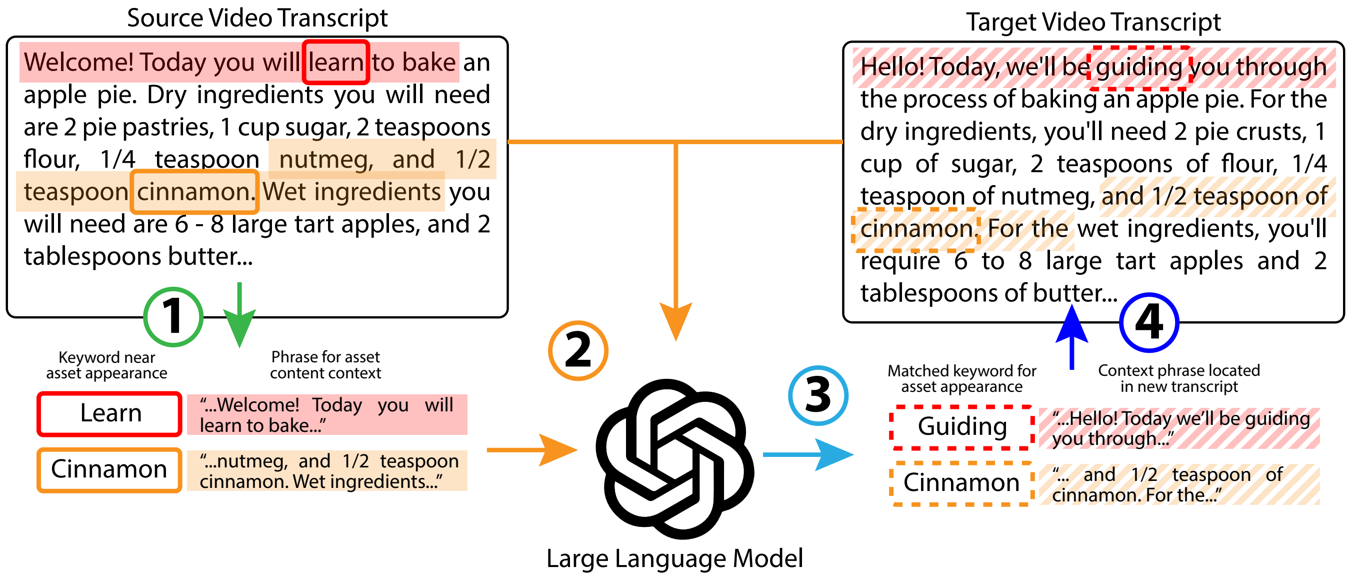

Source Video Transcript
Welcome! Today you will learn to bake an apple pie. Dry ingredients you will need are 2 pie pastries, 1 cup sugar, 2 teaspoons flour, 1/4 teaspoon nutmeg, and 1/2 teaspoon cinnamon. Wet ingredients you will need are 6 - 8 large tart apples, and 2 tablespoons butter...
1
Keyword near asset appearance
Phrase for asset content context
Learn
"...Welcome! Today you will learn to bake..."
Cinnamon
"...nutmeg, and 1/2 teaspoon cinnamon. Wet ingredients..."
2
Large Language Model
3
Target Video Transcript
Hello! Today, we'll be guiding you through the process of baking an apple pie. For the dry ingredients, you'll need 2 pie crusts, 1 cup of sugar, 2 teaspoons of flour, 1/4 teaspoon of nutmeg, and 1/2 teaspoon of cinnamon. For the wet ingredients, you'll require 6 to 8 large tart apples and 2 tablespoons of butter...
4
Matched keyword for asset appearance
Context phrase located in new transcript
Guiding
"...Hello! Today we'll be guiding you through..."
Cinnamon
"... and 1/2 teaspoon of cinnamon. For the..."

Figure 4: Temporal alignment with GPT occurs in four stages. (1) For each asset that appears in the video, we identify the closest lexical word and the phrase location. For example, here assets appear at the words "learn" and "cinnamon." The offset time between the word and asset appearance is also computed using the transcript. (2) The source video transcript and target video transcript are first provided to the LLM. The input word-phrase pairs we obtained are given to the LLM second, sorted by order of appearance in the source video transcript. (3) We prompt the LLM to find the corresponding words and their surrounding phrases in the target video's transcript, giving us output word-phrase pairs. (4) The output word-phrase pairs are used to perform string matching to find the matched lexical word in the target video transcript. The matched word's timestamp recombined with the offset gives the predicted time an asset should appear in the new video.

the target video it intersects any ROI. However, if the source video's composition originally had an overlap, we assume that the user was intentional about this decision and leave it as-is. Video editors employ compositional techniques when arranging visual elements to communicate their vision effectively—whether it be drawing emphasis to the subject, establishing depth, or guiding a viewer's attention [3]. Classical approaches to this in video editing, such as the rule-of-thirds, triangle system, straight-line, and lower-third arrangements [3, 18, 28, 52] are derived from geometric relations that can be captured through ratios between objects in the video and the video frame. We translate these common practices into the rules below to relocate assets.

(1) If a requires repositioning, attempt mirroring it across the y-axis while preserving its ratio of distance to screen-edge over distance to torso-edge, i.e. the ratio of spacing between its two sides.
(2) If a still requires repositioning, attempt aligning it to the left and right edges of the screen with padding.
(3) If a still requires repositioning, center it and...
   (a) ...if a is text insert breaks to get lines with similar length.
   (b) ...otherwise scale its position to the corresponding width or height of the target video.

We arrive at (1) to preserve these "ratio decisions" using mirroring as a potential solution and included (2) and (3) to fall back on safe, standard positions [8, 16] when (1) doesn't work out. These heuristics alone work well for individual assets, but they are difficult to apply when considering multiple assets that are simultaneously visible. Therefore, we apply the geometric rules independently to each asset a with output target positions (x'_a, y'_a) which may overlap one another (Fig. 5) and fix this through a mixed-integer linear program (MILP) to resolve conflicts and generate the final layout.

Linear programming and related approaches have a long history in automatic UI layout [6, 15, 22]. Mixed integer linear programs can guarantee constraints, such as non-overlap between assets, are met while also maximizing an objective, such as similarity to the initial layout. We solve for the optimal set of x_a, y_a top-left coordinates of each asset in a composition. *Binary* decision variables Π_{ij} and Γ_{ij} are defined to prevent overlap between two assets a_i and a_j with a constrained approach standard in the MILP literature [15, 22]. ROI bounding boxes (i.e., the actor's body) are considered as assets in this process except their positions are fixed. To incorporate the proposed locations from rule-based repositioning, we define alignments to (x'_a, y'_a) through t_{ax}, t_{ay}, which are auxiliary bounding variables [7] on each (x_a, y_a). Finally, we employ a visual weight term and padding constraints to favor more plausible layouts and better constrain the problem. Especially in vertical videos, this discourages placement of large assets near the top of a frame which can look awkward, following the principle of *visual weight* where larger objects should be placed lower in a design [4]. The corresponding cost terms create the objective function to minimize. An overview of our MILP is presented below, and the full version is

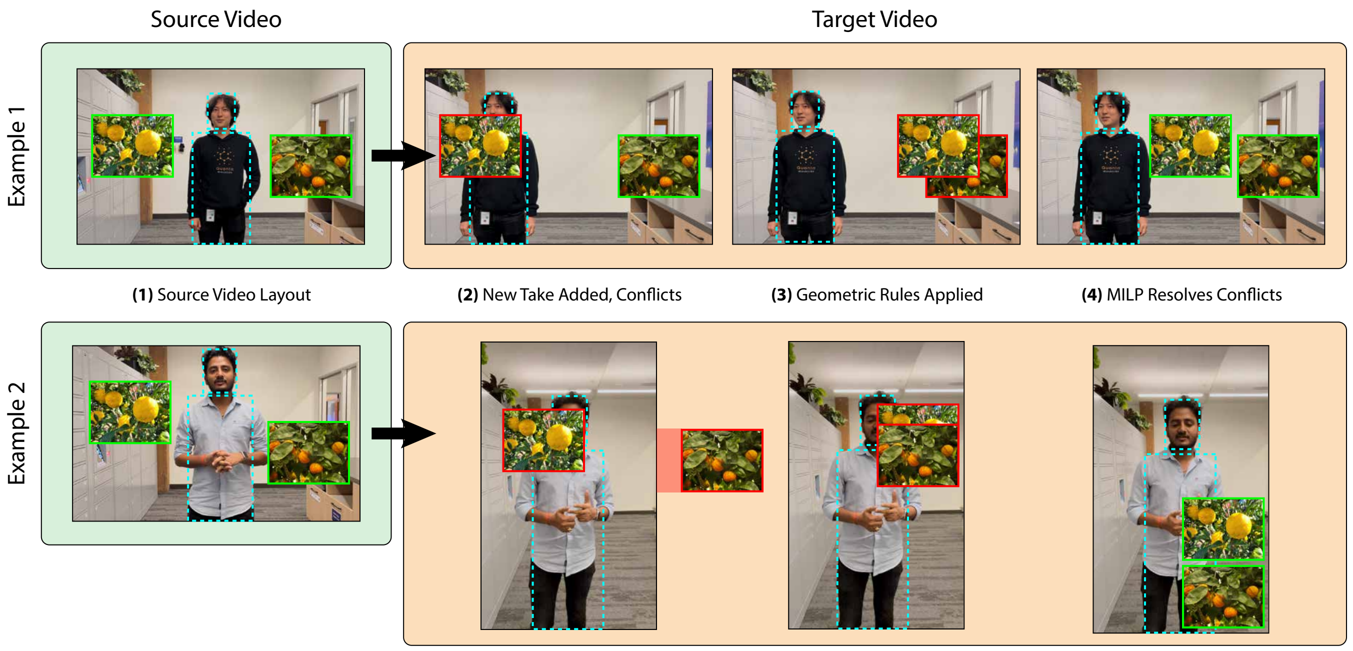

Source Video
Target Video
Example 1
Example 2
(1) Source Video Layout
(2) New Take Added, Conflicts
(3) Geometric Rules Applied
(4) MILP Resolves Conflicts

Figure 5: Spatial retargeting occurs in four stages, shown here for two different videos, on the top in a landscape orientation, and on the bottom in a portrait orientation. (1) We start with the layout composed by the user in the original source video. (2) We directly apply this layout to the target video at the retargeted time. This causes unwanted overlaps and even off-screen clipping when the aspect ratio changes. (3) We apply geometric rules such as mirroring along the torso, preserving margin ratios, and bounding to the screen. This fixes the majority of issues except overlaps. (4) We solve a mixed-integer linear program (MILP) to resolve these overlap conflicts. Assets outlined in green have no issues, and assets outlined in red have conflicts. The actor's head and torso bounding boxes are drawn in teal.

presented in Appendix B.

$$\begin{aligned} \text{min} \quad & c_{\text{non-overlap}} + c_{\text{alignments}} + c_{\text{visual weight}} \\ \text{subject to} \quad & \Pi_{ij}, \Gamma_{ij} \text{ non-overlap constraints} \\ & t_{ax}, t_{ay} \text{ alignment constraints} \\ & p \text{ padding constraints} \\ & \text{ROI fixed constraints} \end{aligned}$$

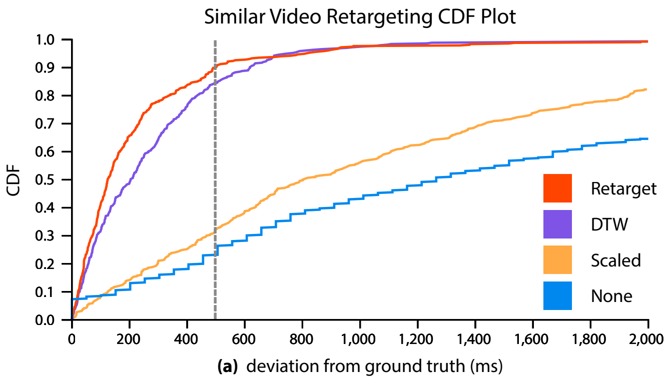
The benefit of using MILPs is the separation of satisfying constraints and minimizing cost terms. However, it is possible that no spatial layout satisfies the constraints (e.g., an asset must overlap the performer to stay within the video bounds) and the MILP is infeasible. In that case, we iteratively remove constraints for ROIs to "relax" the problem until it is satisfiable. We schedule removing constraints by taking out overlapping with the torso first, overlapping with the face second, and then overlapping with other assets and going off-screen. The final satisfying layout yields the asset positions $\{x_a, y_a\}_{a=1}^{n}$, which is the output of the spatial retargeting step.

5 Technical Evaluation

We perform a technical evaluation to help characterize the effects of our retargeting algorithm. We evaluate this algorithm with a number of quantitative measures against hand-annotated ground-truth data. We consider two use cases for evaluation: retargeting between "similar videos" and retargeting between "dissimilar" videos. We define similar videos as recordings of the same performance, where the actor is refining the cadence and speech (e.g., removing pauses or stutters), while dissimilar videos are recordings of a performance that is being improved iteratively (e.g., changing the words for better communication).

5.1 Implementation

VidSTR is implemented as a Typescript application running in Google Chrome, backed by a Python Flask server. We use the Speechmatics API [1] to obtain transcripts of the actors' speaking. We use OpenAI's GPT-4o API [41] for a large language model and a body pose tracker comparable to Google's Mediapipe [20] for pose tracking. We use the open-source HiGHS library [23] to solve mixed integer linear programs. Our videos were all filmed in either 1920 × 1080 (horizontal) or 1080 × 1920 (vertical) resolution on an iPhone 12 Mini. We ran VidSTR on an Apple M1 MacBook Pro. Transcription takes ~1 minute per minute of video and pose tracking takes ~2 minutes per minute of video. Running the entire algorithm, combining both temporal and spatial retargeting, takes on average 5 seconds total for our test compositions with the majority of time spent on the GPT-4o call.



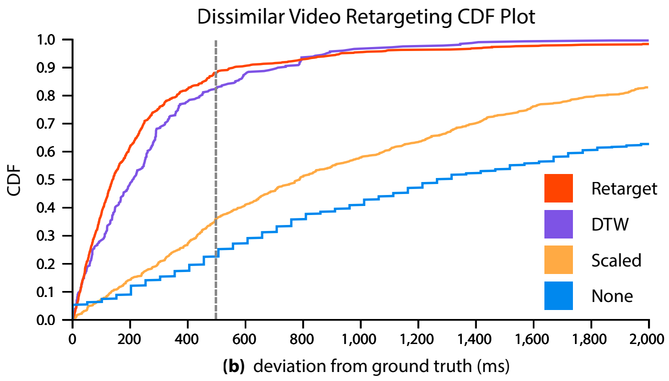


Figure 6: On the left (a), we plot the CDFs of temporal alignment performance between similar videos for our retargeting approach (labeled "Retarget") and three baselines (labeled "None", "Scaled", and "DTW") on a time horizon of two seconds. The grey dashed line represents a 500ms threshold. 90% of "Retarget" is within this threshold compared to 23% for "None", 31% for "Scaled", and 84% for "DTW". On the right (b), we similarly plot the CDFs of temporal alignment performance but with dissimilar performances. 88% of "Retarget" is within the 500ms threshold compared to 22% for "None", 35% for "Scaled", and 83% for "DTW". In both scenarios, retargeting greatly outperforms the no-change and scaling baselines and is comparable to the DTW baseline.

5.2 Procedure

We recorded a total of 66 videos distributed evenly across six different scripts with 11 people (8 male, 2 female, 1 non-binary) recruited through snowball sampling. The duration of each video is approximately one minute (μ=67.4 secs, σ=11.2 secs). Actors were asked to take on the persona of a content creator before video recording. Actors were also encouraged to loosely follow the transcript and to vary both the spoken and gestural cadence of their performances. The scripts were excerpted and modified from public domain text and an informal interview. Their topics, listed below, are intended to span a variety of video genres and content creation applications:

[T1] Wikipedia article about lemons [63] (infotainment)
[T2] *Alice in Wonderland* by C.S. Lewis (video essay)
[T3] Review about 3 Boston bakeries (vlog)
[T4] How to bake apple pie [62] (instructional)
[T5] "Birches" by Robert Frost (lyric and performance)
[T6] Churchill's "We Shall Fight on the Beaches" (interview)

These transcripts are included in the supplementary materials. For T1, T3, and T4, the videos were recorded in sets of three: two where participants followed the same transcript ("similar" videos from doing a video retake) and one where participants followed a reworded transcript ("dissimilar" videos from iterating on a performance). For T2, T5, and T6, the videos were recorded with the same respective transcripts. 3 takes of T5 and 3 takes of T6 were removed from the dataset due to the body tracking library failure. For similar videos pairs, we use take 1 $\longrightarrow$ take 2 for all transcripts and take 2 $\longrightarrow$ take 3 for T2/T5/T6 giving us a total of 29 pairs of videos. For dissimilar video pairs, we use take 1 $\longrightarrow$ take 3 and take 2 $\longrightarrow$ take 3 for T1/T3/T4 giving us a total of 22 pairs of videos. On average, creating the original composition took about 20 minutes for each pair. Manually retargeting them to the new corresponding video (for a ground truth comparison) then took about 15 minutes per pair. To contrast, using VidSTR's automatic, algorithmic retargeting takes less than 5 seconds per pair as described in section 5.1.

Each of the six transcripts is associated with a collection of 15 assets including text and graphics. The corresponding collections are consistently edited onto each video by hand, and we use these asset timings and positions as ground truth for our evaluation. The paper author(s) performed the edits and validated against two other individuals who re-created a subset of the edits. We computed inter-rater reliability with Krippendorff's alpha (bin size = 1 second) at α = 0.62, which suggests relative agreement. The ground truth edits follow a video editing style common on social media and content platforms—such as YouTube, Instagram, and TikTok—where appearance of text and graphics matches the timing of ideas discussed in the video. Therefore, for similar videos we have $29 \times 15 = 435$ total retargetings to compute, and for dissimilar videos we have $22 \times 15 = 330$ total retargetings to compute.

For each video pair, we run our algorithm using the ground truth graphics of the source video as an input composition to be retargeted onto the targeted video, which are then compared to the ground truth graphics of the target video.

5.3 Temporal Alignment Results

To quantify temporal alignment results, we compare the absolute difference between an asset's retargeted time and ground truth time (labeled "Retarget"). We compare VidSTR's retargeting performance against three baselines. For the first baseline, we apply the source video's ground truth times to the target video, which is the current behavior of most commercial video editing tools (labeled "None"). For the second baseline, we naively scale the source video's ground truth by the ratio between the source and target video lengths (labeled "Scaled"). For the third baseline, we employ dynamic time warping (DTW) to find a transcript-to-transcript alignment that maps the source video's ground truth times to the target video (labeled "DTW"). While dynamic time warping is a standard method for time series and sequence alignment [36, 38, 47], no prior work has specifically focused on transcript-to-transcript alignment so we adapt the algorithm ourselves by building the cost matrix based on word matches. The aggregate deviations of these three approaches are plotted in Fig. 6 as cumulative distributions (CDFs).

For "Retarget", the overall mean deviation from ground truth is $\mu = 295$ms with a standard deviation of $\sigma = 477$ms. For the "None" baseline, the mean deviation is $\mu = 2181$ms with a standard deviation of $\sigma = 2410$ms; for the "Scaled" baseline, the mean deviation is $\mu = 1190$ms with a standard deviation of $\sigma = 1224$ms; and for "DTW", the mean deviation is $\mu = 288$ms with a standard deviation of $\sigma = 340$ms. The aggregate results show that LLM and DTW retargeting perform similarly and improve over the no-intervention and naive baselines. For this particular video composition retargeting application, we are concerned with more fine-grained performance data because even just a few poorly timed visuals can contribute negatively to viewers' experience of the video [50].

Analysis of the CDFs reveals that between similar videos, 90% of "Retarget" assets are within 500ms of ground truth followed by 84% for "DTW", compared to "None" and "Scaled" achieving 23% and 31% respectively. As for dissimilar videos, 88% of "Retarget" assets are within the threshold to which "DTW" achieves a close 83%, compared to 22% and 35% for "None" and "Scaled". Prior literature establishes humans are sensitive to lip-syncing latency as low as 100ms [9, 27, 39, 57]. We select a looser threshold because the *semantic* alignment of content is more important and less precise—the window for a "user's flow of thought" is around a second [39]. Nevertheless, we recognize the subjectivity of such choice, and so we provide the CDFs to illustrate performance at shorter and longer thresholds. We find that VidSTR temporally retargets the vast majority of the assets in our videos within a useful margin of error.

5.4 Robustness Compared to DTW

For both similar and dissimilar transcripts, it is somewhat surprising that the LLM and DTW methods have similar success rates at temporal retargeting. We conducted an additional experiment with more challenging retargeting scenarios to gain deeper insight into their capabilities. We took transcripts T1, T3, and T4 (the transcripts used for dissimilar retargeting evaluation) and generated 75 additional versions (25 per each) with ChatGPT at varying levels of dissimilarity measured through word mover's distance (WMD) [32]. We prompted ChatGPT to reword the transcripts while maintaining the same semantic content. Our original dissimilar transcript pairs have WMD between 0.3-0.4, so we generated the new versions to be distributed throughout $[0, 1]$ to test larger differences. We then ran temporal retargeting between the original T1/T3/T4 transcripts and their respective additional versions with both DTW and LLM. Plotted in Fig. 7 are the mean deviations from the ground truth across the levels of WMD dissimilarity.

As expected, for WMD < 0.4, both DTW and LLM consistently achieve high quality temporal alignment. However, for WMD ≥ 0.4, DTW frequently shows significant errors with up to 10 seconds of mean deviation compared to LLM's maximum of 2.5 seconds. Examination of specific error instances reveals DTW struggles when (1) transcript changes correspond with asset order changes, when (2) significant content between assets is added or removed, and when (3) no common words are shared between the transcripts despite containing similar semantic content. It is surprising that the LLM performs as well as DTW in easier cases too, as DTW performs an optimized, dense alignment between the texts while the LLM relies on sparse semantic similarity. Ultimately, both LLMs and DTW can be used in VidSTR for the main use case of retargeting between pairs of similar or dissimilar transcripts. We use a LLM because it

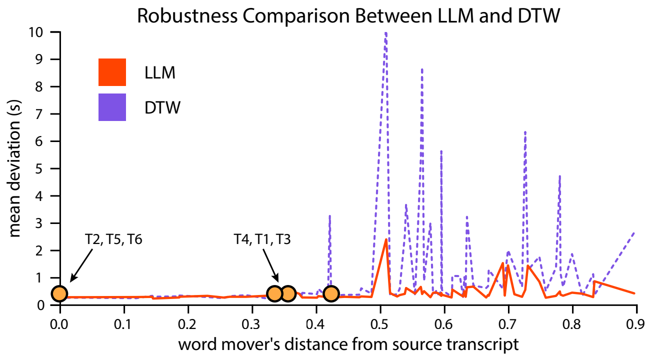


Figure 7: We compute the mean deviation from ground truth when temporally retargeting between the source and additional target transcripts. This is plotted against the word mover's distance (WMD) between the pairs of source and additional target transcripts. We observe that for similar source and target transcripts with WMD < 0.4 that there is no significant difference between using DTW and an LLM for transcript alignment. However once the difference between transcripts is much larger, both algorithms begin to struggle but DTW fails much more catastrophically. The orange dots indicate for similar transcript pairs (e.g. all of T2/T5/T6) in our technical evaluation, WMD ≈ 0, and for dissimilar transcript pairs (e.g., dissimilar T1/T3/T4), WMD ≈ 0.4.

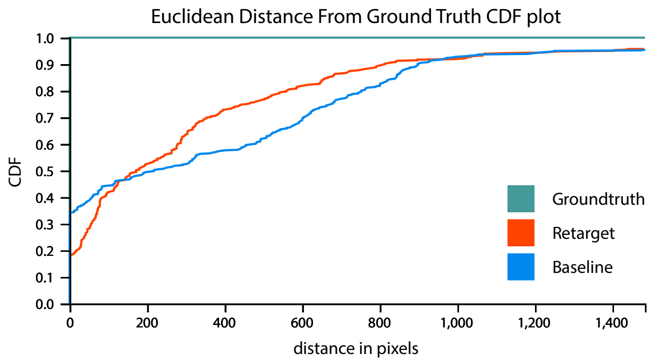


Figure 8: We plot the CDFs of Euclidean distance of an asset's (x, y) position from the manually created ground truth to both auotmatic retargeting and the baseline of the source video placements. There is no significant difference globally between the two with this choice of metric, despite the clear improvements, such those shown in Fig. 5. We argue Euclidean distance is not a good way to evaluate spatial retargeting and we propose percentage overlap with ROIs instead.

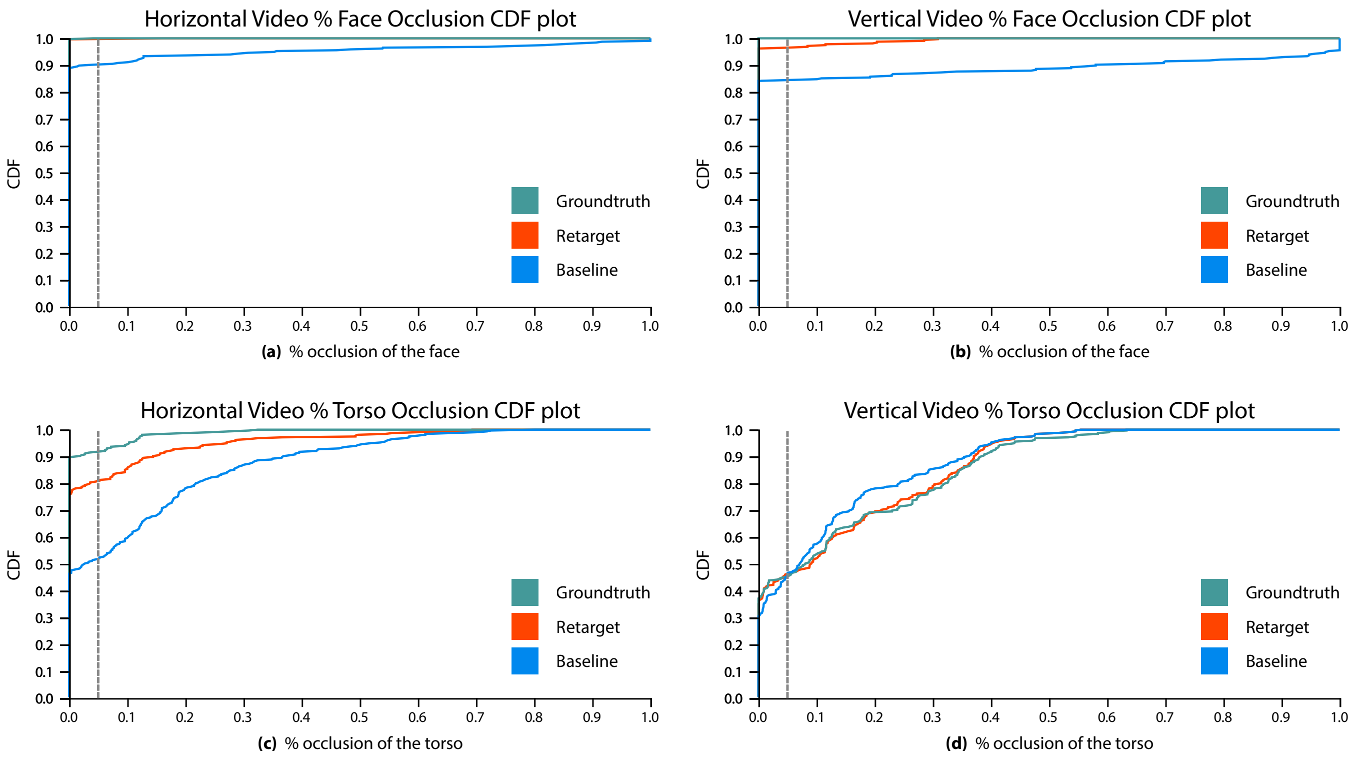


Figure 9: We plot the CDFs of percentage occlusion with the face and torso, separated between horizontal (top row) and vertical (bottom row) videos. (a) For percentage face occlusion in horizontal videos, the ground truth and retargeting are able to avoid the face entirely, while the baseline fails in ∼10% of cases. (b) For percentage face occlusion in vertical videos, retargeting is close to matching ground truth performance, while the baseline does worse. (c) For percentage torso occlusion in horizontal videos, retargeting is again closer to matching ground truth performance. (d) With percentage torso occlusion in vertical videos, performance between all three methods is similarly poor due to the area restrictions of the vertical format.

further enables more challenging applications later demonstrated in our applications.

5.5 Spatial Alignment Results

Unlike temporal alignment where good placements of assets are easier to describe and detect, evaluating good spatial alignment is more challenging. There can be many potential compositions or ways to lay out the assets that are equally acceptable when resolving positioning conflicts. As a test, we compute the euclidean distance of an asset from the ground truth, and find that conflicting layouts without any intervention (labeled "baseline") and retargeted layouts (labeled "retarget") perform similarly (see Fig. 8). However, retargeted layouts are clearly preferable to the baseline of no intervention in many cases, as seen in Figs. 2 and 5.

The primary motivation for spatial retargeting is to avoid assets overlapping the performer's face and torso, which may have moved between the two videos. Therefore, we compute the percentage occlusion of the face and torso as the evaluation metric. Results are separated between horizontal target videos and vertical target videos because the constraints on composition are drastically different between different aspect ratios. We plot the CDFs for these results in Fig. 9.

With regard to occlusion of the face in both horizontal and vertical videos (Fig. 9a, 9b), VidSTR's performance matches the ground truth in being able to avoid face overlap in almost all retargetings. The baseline, on the other hand, occludes the face approximately 10% of the time which is not acceptable. With regard to occlusion of the torso in horizontal videos (Fig. 9c), spatial retargeting is also able to reduce a significant percentage of torso overlap. Interestingly, for *vertical* videos (Fig. 9d), performance in occluding the torso is similar across baseline, retargeting, and ground truth. This may be due to the particular constraints of vertical videos—in the vertical format, actors take up more space with the narrower screen width, and it can be impossible to position an asset without torso overlap. In this case, it can be even desirable to introduce more overlap in order to prevent the asset from running off the screen. VidSTR allows some torso overlap with text in multi-asset compositions to preserve the original layout better and maintain some padding within the frame.

6 Applications

VidSTR's retargeting enables a number of different application scenarios, which we explore here.

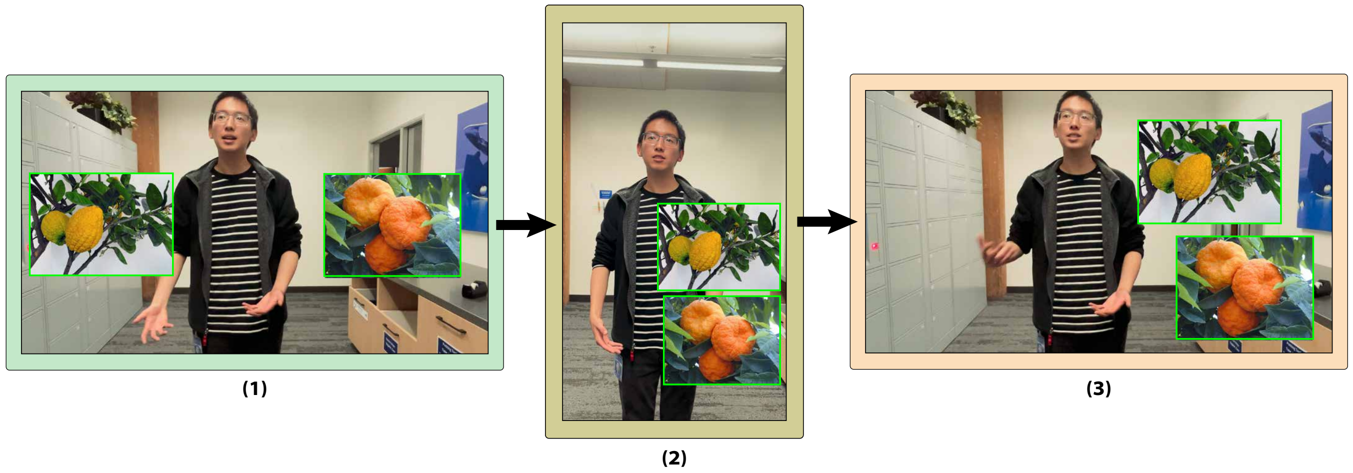


Figure 10: (1) The source landscape video has two assets, one on each side of the speaker. (2) A target portrait orientation video has the same two assets automatically re-positioned to fit the frame while avoiding the face. (3) Retargeting the vertical video composition to a different video (which happens to also be landscape-orientation but is a different video performance than (1)) leads to a different placement of assets. Note that spatial retargeting does not find a bijective mapping, as seen in the difference between the first and third pictures, i.e., going from horizontal to vertical and back to horizontal may not give back the starting layout.

6.1 Aspect Ratio Retargeting

Video editors often wish to perform aspect ratio retargeting in their projects, such as to export their video creations to different platforms. Desktop viewers (e.g., YouTube, Facebook) typically require landscape viewing while mobile viewers (e.g., Instagram, TikTok) require portrait viewing. While our spatial retargeting algorithm was designed to resolve overlapping conflicts between a source and target video, the width and height of the target video are set as arbitrary constants $\mathbb{W}$ and $\mathbb{H}$ in the MILP. There is no restriction on the width and height of the source video, and we find that VidSTR retargeting between different aspect ratios generates acceptable compositions, as seen in Fig. 10. Also, the two videos can be either be same recording—same as modifying the source video—or they could be totally different recordings. For instance, the user can film a recording in landscape mode first and film a recording in portrait mode second, and then VidSTR can seamlessly retarget assets between these two different takes. Aspect ratio retargeting can involve going from horizontal (landscape) to vertical (portrait) videos or from vertical to horizontal videos. Each case has specific considerations that can improve the final re-layout.

For instance, when changing aspect ratio from horizontal to vertical videos, the amount of horizontal space around the performer decreases substantially, or even completely if the performer now takes up the entire frame. Large assets positioned to the left and right of the performer have less space and overlapping the torso can be inevitable. The video also has more vertical space to position assets. We experimentally find that prioritizing visual weight during spatial retargeting improves results in this scenario. VidSTR is able to rearrange assets originally placed to the left and right (Fig. 10.1) and stack them to the bottom right side of the target video (Fig. 10.2). While the torso overlap is not ideal, this is an acceptable layout compared to the alternatives.

6.2 Retargeting Between Different People

We also find that VidSTR can retarget between different people performing the same scenes (Fig. 11). We tested video composition retargeting with swapping performers across the same scripts and found only a negligible drop in performance. The usage of high-level features, such as spoken transcripts, over low-level features, such as the audio signal, allows for greater flexibility in retargeting across different people. An organization may have a diverse range of performers they wish to appear in different versions of a production, such as a commercial, and adding graphics to each performer's take would be a tedious process. With video composition retargeting, the edited assets on one initial video can be automatically retargeted onto all of the different performances.

6.3 Robustness Against Transcript Variations

In our technical evaluation, we considered two scenarios of retargeting between similar and dissimilar pairs of video takes. To consider similarity more generally, video pairs can have utterance-level (micro) variances with added disfluencies (e.g., "um's"), sentence-level variances such as rewordings and syntax changes, and outline-level (macro) variances such as swapping the order in which content is presented in a video. Any level of these variations can appear during the iteration process of video creation when replacing one video take with another. For instance, macro-level variations could appear during the initial script refinement stage, while micro-level variations could appear later in the process (i.e., "iterating on an idea" versus "getting the perfect take").

In similar takes, transcripts almost match word-for-word, and it is no surprise that retargeting works. For dissimilar takes, however, sentence-level variances of the spoken content described in Fig. 12 are overcome by the large language model's semantic reasoning abilities [44]. VidSTR is notably robust to the macro-level variations

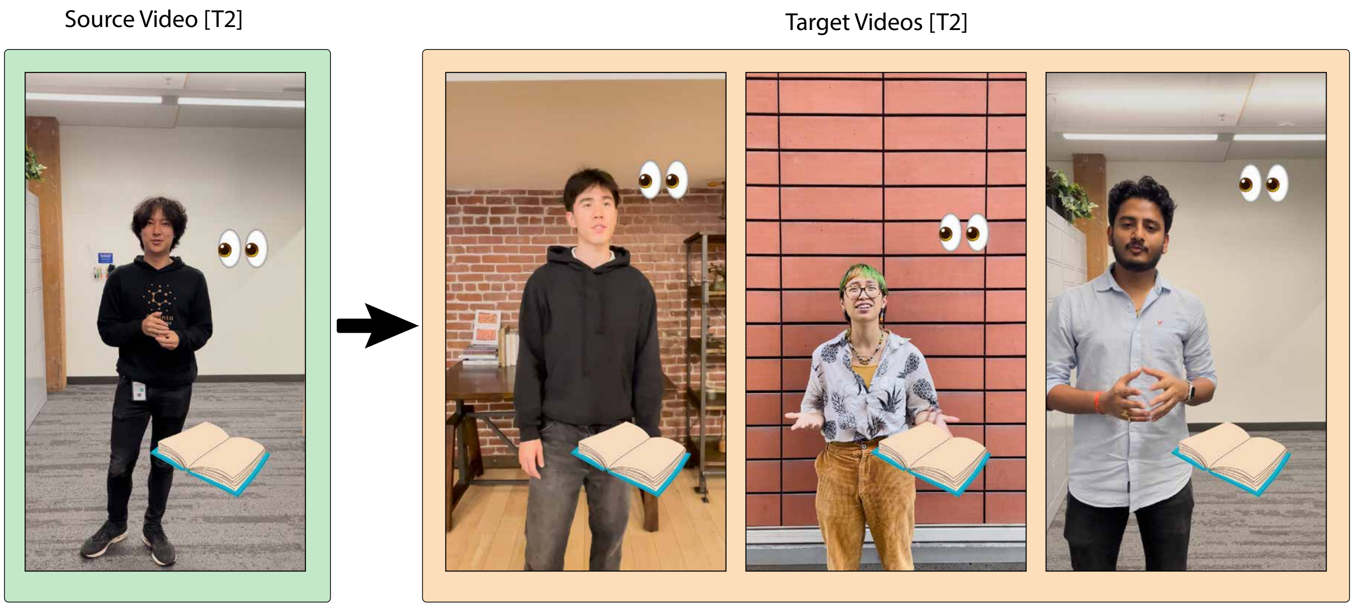


Figure 11: Retargeting between people. Left: the source video composition for transcript [T2] at 7.5 sec. Right: the target videos with different performers that have the retargeted source video composition applied. Note that the "eyes" asset dynamically adjusts to various plausible heights between takes while the "book" asset remains in a similar position across different performers.

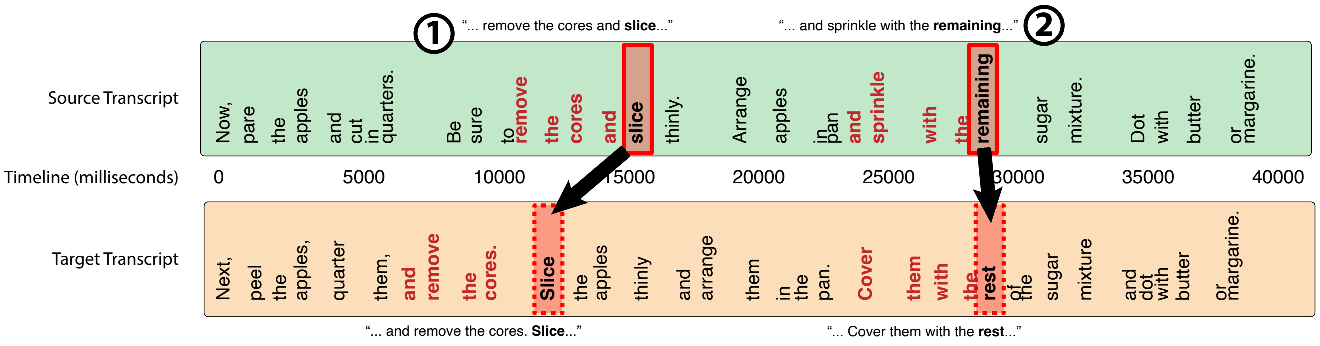

1
"... remove the cores and slice..."
"... and sprinkle with the remaining..."
2
Source Transcript
Now, pare the apples and cut in quarters. Be sure to remove the cores and slice thinly. Arrange apples in pan and sprinkle with the remaining sugar mixture. Dot with butter or margarine.
Timeline (milliseconds)
0
5000
10000
15000
20000
25000
30000
35000
40000
Target Transcript
Next, peel the apples, quarter them, and remove the cores. Slice the apples thinly and arrange them in the pan. Cover them with the rest of the sugar mixture and dot with butter or margarine.
"... and remove the cores. Slice..."
"... Cover them with the rest..."

Figure 12: In this pair of videos, the two transcripts have varying wordings but similar semantics. The lexical word an asset is tied to is bolded. At example (1), the algorithm retargets from "...remove the cores and slice..." to "...and remove the cores. Slice..." which is a relatively straightforward case because the words mostly match. At example (2), the algorithm retargets from "...and sprinkle with the remaining..." to "...Cover them with the rest..." which is a more difficult case because the target words and context do not match syntactically, only semantically. The large language model can handle both alignment cases.

between takes as well, *such as when we swap entire paragraphs in the transcript*, as illustrated in Fig. 13. We give the large language model entire transcripts of the source and target videos at once, so it reasons on the full video content stored in its context window. We tested video composition retargeting by swapping sections of our transcripts and found only a negligible drop in performance. The flexibility and reliability of retargeting across different granularities means video editors can safely edit assets onto their videos early, knowing they can swap in new takes without hassle.

6.4 Localization: Retargeting Across Languages

Localization of video content helps cater media to different viewers in an increasingly globalized world. For example, pre-flight safety videos are a type of content that needs to be performed in different languages yet have consistent edits—with a different actor for each language. Assuming the English performance has had graphical assets added first, the task of transferring them to the various language performances is extremely laborious with current tools. The

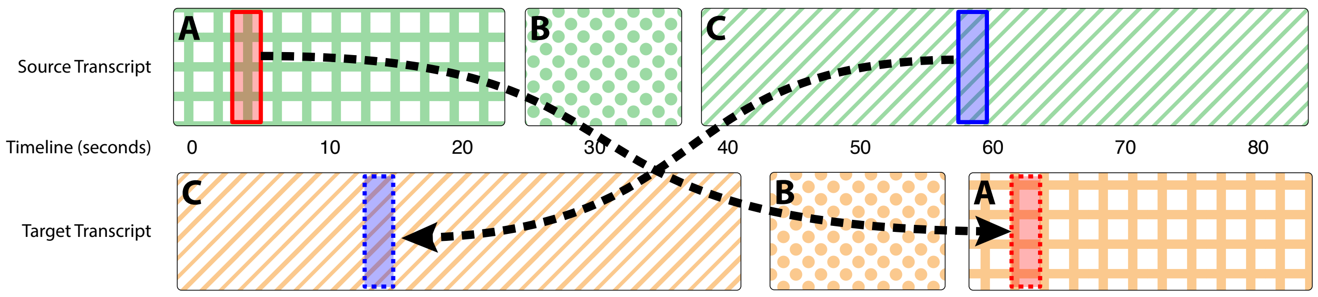


Figure 13: We present a retargeting scenario where the video content is separated into ordered sections A, B, and C in the source video. When the target video has macro-level reordering of content, such as when entire sections A and C are swapped, VidSTR generates correct mappings that respect the re-ordering. The large language model takes the entire transcript at once and can reason about the semantic content at both a global (inter-paragraph) and local (inter-word) level.

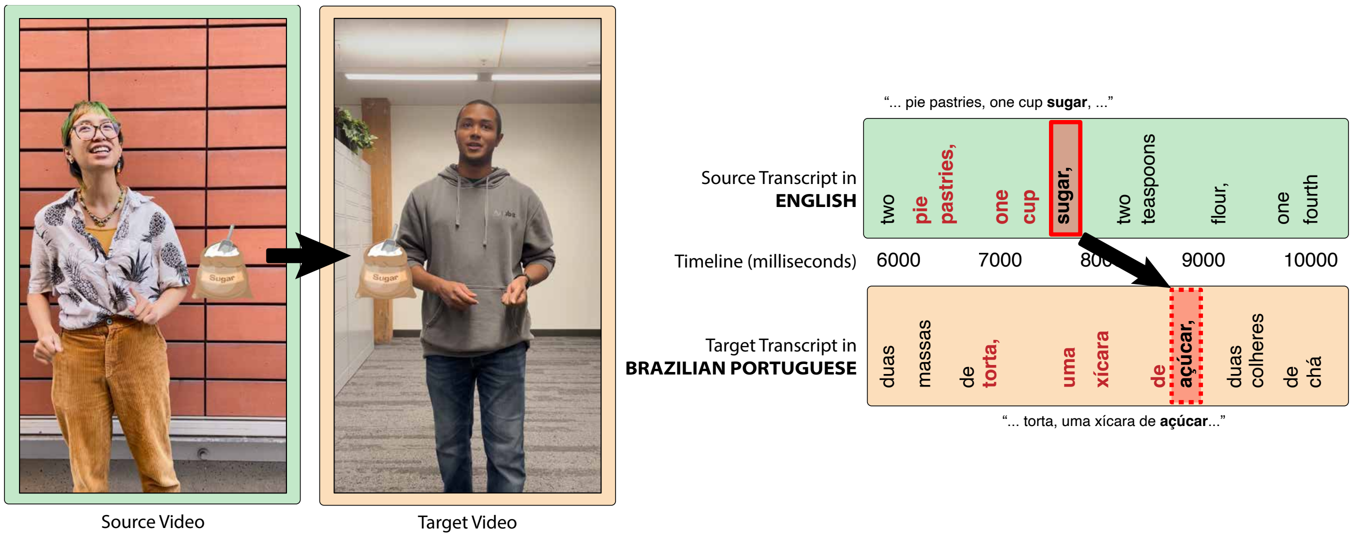

Source Video
Target Video
"... pie pastries, one cup sugar, ..."
Source Transcript in ENGLISH
two pie pastries, one cup sugar, two teaspoons flour, one fourth
Timeline (milliseconds)
6000
7000
8000
9000
10000
Target Transcript in BRAZILIAN PORTUGUESE
duas massas de torta, uma xícara de açúcar, duas colheres de chá
"... torta, uma xícara de açúcar..."

Figure 14: Localization. Left: The source video is in English, with an asset appearing with the spoken word "sugar." Center: The target video is in Brazilian Portuguese, with the same asset retargeted to appear with the word "açúcar," which means sugar. Right: Our automatic retargeting technique can find correspondences across languages with words spoken at different times.

video editor must work with a different translator for each language to ensure assets are translated to the correct spatiotemporal position in the new video.

We find our technique can take in a source video in English and a target video in a foreign language and automatically perform the retargeting without any additional modifications. We took the transcript [T4] about baking apple pie and recruited native speakers to perform it in Brazilian Portuguese and Italian. This experiment produced results on-par with English-to-English retargetings. The Speechmatics API we used auto-detects the language spoken in an audio file and accurately transcribes the spoken, say, Brazilian Portuguese. The GPT-4o API we used is a multilingual large language model that can semantically reason across languages, and no additional prompting was used. Video composition retargeting opens the door to more easily creating multiple versions of videos in different languages.

This application is one example of how large language models can be used to support localization. The majority of artifacts and prior work in this space assume English language as the de-facto input and output, and we hope VidSTR shows the potential of large language models to offer greater diversity in output content for HCI systems research.

7 Qualitative Evaluation

To complement our technical evaluation, we also conducted a preliminary expert review to gain qualitative feedback on VidSTR. We recruited three video professionals (one male, two female, ages 20–25) and conducted semi-structured interviews to understand VidSTR's potential in real-world workflows and usage scenarios. E1 is a content creator on Instagram with over seventy million views and has used CapCut for video editing for three years. E2 is an animation student with three years of commercial experience and has used After Effects for four years. E3 is a video editor and cinematographer with five years of industry experience and has used Premiere Pro for ten years. Interviews were conducted through one-on-one sessions lasting forty-five minutes. Participants were shown a video overview of the tool, and three retargeting video pairs (between dissimilar transcripts, aspect ratios, and persons/languages) included in our supplemental materials before discussing results and system functionality.

7.1 VidSTR saves video editors significant time

Our experts responded very positively overall to VidSTR as a one-click solution to re-aligning edits when swapping out video takes. E1, E2, and E3 all agreed that VidSTR's outputs were helpful in the retargeting problem and would like to see it integrated as a one-click button into existing video editing apps. When asked about their current workflow for retargeting, E1 shared *"it's especially annoying when I have to reposition, re-listen to my own clips and then move the text to what I'm saying."* For a one-minute reel, E1 said this process takes them at least 30 minutes. E3 mentioned they *"use a lot of one button things in video editing all the time ... as a starting place"*, and they thought that VidSTR would *"definitely help [with] initial layout and getting you started."* E3 said that retargeting a one-minute clip manually without VidSTR would take fifteen to thirty minutes, whereas with VidSTR, it would take around five minutes. Furthermore, participants were generally satisfied with the retargeting outputs. E2 said VidSTR would be particularly useful for productions that require *"quick making of a high volume of short form content."* E3 mentioned VidSTR works well for an *"editing style very common in social media videos,"* but they would occasionally *"be worried about quality if ... the client would be very specific about exactly where they want [key assets] in the frame,"* such as a logo. Nevertheless, all three experts agreed that VidSTR would save them time and provide a good starting point, which could be manually tweaked in some cases.

7.2 VidSTR alleviates the burden of strict ordering between recording and editing in video workflows

In their various professional backgrounds, our experts expressed frustration over the fact that in commercial video workflows, swapping out old video takes for new ones is a burden they must work around. E2 shares that currently *"when I make videos at first, I know I'm going to be doing a lot of additional editing on top of it. So I spend a super huge amount of time making sure that initial core video is perfect."* E2 said that if VidSTR gives the assurance that *"assets on top could be shifted around willy nilly, then I probably would spend less time making sure the initial take is absolutely perfect and kind of jump into editing earlier on in the workflow."* This is important in commercial work because *"it makes that less stressful because clients change their mind, like every two hours. And if you can, if you know that things are more flexible on your side, that becomes less stressful."* E1 shared similar experiences in content creation and how VidSTR would help: *"I think I wouldn't worry about editing too much or getting the perfect first take. Especially because when [sponsoring] brands reach out to you and you have to get their feedback and kind of readjust based on everything, it'd be a lot easier."* They recalled working with one client where *"they gave me feedback with certain timestamps ... I did have to refilm and then I had to readjust everything."* These scenarios are representative of the major problem VidSTR addresses; VidSTR allows for much faster readjustment and gives editors flexibility on when and where they want to do editing in their workflow.

7.3 VidSTR reduces retargeting effort for different platforms and languages

Our experts highlighted that a significant part of the video editing workflow is spent on supporting various export destinations of their content across viewing platforms and locales. Within their professional work, E3 said *"I have to export a lot of different versions, but I usually have one main one that's the longest one in horizontal. And [for social media] then I'll cut that into the vertical version that's sixty seconds."* VidSTR assists with retargeting the assets to different aspect ratios and formats.

As for working with different languages, E3 shared that for one project, *"I actually I edited a documentary in French that I couldn't understand at all, but Adobe Premiere had a transcription thing, and French is one of the languages it transcribed. So it transcribed everything in French first and then I put that in Google Translate. But that wasn't the most accurate thing. So I also had two translators helping me out."* E3's experience reveals working across language requires an intricate, suboptimal pipeline of software and translators that can be replaced with an LLM as demonstrated in VidSTR. And while E1's content is currently directed towards an English-speaking audience, they imagined VidSTR would help if the *"content required the creator to talk in different languages or something like that, or create different versions of kind of the same product."* E3 summed it up with the following: *"It's always annoying when you have to create like ten different versions of the same video or one for YouTube, one for Instagram, one in a different language. But, if you could do that all in one click of a button, and be super nice because then you would just edit your one main video."*

7.4 Expert-suggested extensions for composition retargeting in VidSTR

Throughout the semi-structured interviews, our experts asked for various capabilities in the retargeting process related to their practice. E3 emphasized their desire to see realignment of assets aside from graphical overlays such as audio clips and color gradings used extensively in film: *"Video is not just images. It's a lot of audio too and it's often overlooked."* In VidSTR's current form, these components can be injected into the transcript as screenplay directions to be processed by an LLM for temporal alignment. Meanwhile, E2 discussed the dynamic effects and transitions they employ in their work: for instance, when an asset appears *"there's like a some sort of wiggle effect or it swings in from the side or it fades in or something."* VidSTR only supports static assets, so recognizing and representing patterns of dynamism is integral to faithfully translating edits. E2 said for a fully automated tool to do this, they'd need the output to be *"at the standard quality that I've been making for however long I've been making that content."* In the meantime when automatic systems don't quite reach that bar, E2 would be happy to *"just go in and just tweak it back to match."* Echoing earlier sentiments, the algorithm saves the editor time through a good initialization for retargeting—it does not need to entirely replace the editor when it comes to stylization.

8 Limitations and Future Work

In our technical evaluation, our videos were speech-centric, featured a single person, and had a static camera. Because temporal alignment relies on a transcript, VidSTR only works with speech-driven videos, which are a common form media content. We do not make any assumptions about the camera motion or number of speakers, which are handled by the body tracker and transcription respectively. Spatial alignment assumes a single person, but it would be straightforward to add constraints for additional performers in the MILP. Since VidSTR uses the mean bounding box for a performer's ROI, it implicitly assumes the performer stays relatively still in the camera frame; if this is not true, some assets may overlap with the performer for portions of their duration.

VidSTR supports a variety of use cases rooted in video composition retargeting, but there are still many other directions for additional exploration. Our algorithm is strictly limited to find a one-to-one mapping between assets in the source and target videos. It could be interesting to relax this requirement, adding support for 1) one-to-many mappings where an asset appears multiple times in the target video, 2) one-to-none mappings where source video assets are removed from the target video, and 3) zero-to-one mappings which introduce new content into the target video. For example, an LLM could recommend new assets to add to the composition.

There are further opportunities to consider expanding the results retargeting can produce. We do not adjust the duration of an asset on the timeline, nor to we adjust an asset's size. We also do not consider assets that move over time (e.g., with keyframed animation). More fundamentally, we restrict ourselves to videos with one performer, mostly speaking, but we could expand VidSTR to work on videos with no on-screen performers (e.g., a nature documentary), multiple performers (e.g., an interview), or without speech (e.g., a dance video). These extensions could help achieve better compositions across a broader range of content.

Finally, our algorithm provides a one-click solution to video composition retargeting. Particularly with spatial retargeting, the desired output composition is subjective, but our tool has no means for a user to input stylistic preferences, beyond those that they have made in the input composition. Incorporating user interaction into an iterative retargeting process would increase the user effort needed, but could yield results that better match the user's artistic intent across a wider variety of compositions.

While we conducted expert interviews to evaluate VidSTR's results and implications, we acknowledge performing user studies in real-world scenarios would better validate the system's effectiveness at assisting video composition retargeting tasks. Furthermore, future work should incorporate structured workflow studies to comprehensively assess VidSTR's impact and potential benefits in video editing practices.

9 Conclusion

VidSTR demonstrates that assets can be transferred from one video to another automatically by retargeting on temporal and spatial dimensions, so the effort of placing assets the first time can be leveraged for other creative endeavors. Our technical evaluation shows that temporal retargeting using LLMs for transcript-based alignment and spatial retargeting using mixed integer linear programming are effective across different topics of videos and different performers. In specific cases we found that gesture fine tuning led to crisper alignment between a gesture and an overlaid asset.

Our technique automates one time-consuming step in video composition creation. This enables a high-fidelity prototype to be created earlier in the video editing process. By allowing critical assets to be placed in videos without worry that they will all have to be placed again each time the video is edited, VidSTR enables a more iterative workflow for the creator who can see the composition before different versions, edits, or actors are finalized. This can make the overall process less like a rigid "waterfall" where each step must be completed in order before moving to the next. Overall, the VidSTR system contributes to the broader landscape of automated and semi-automated tools that enable new video editing prototyping, workflows, and outputs.

References

[1] 2024. *Speechmatics.* https://www.speechmatics.com/ Accessed: 2024-08.

[2] Ido Arev, Hyun Soo Park, Yaser Sheikh, Jessica Hodgins, and Ariel Shamir. 2014. Automatic editing of footage from multiple social cameras. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–11.

[3] Daniel Arijon. 1976. *Grammar of the film language.* Silman James Press.

[4] Rudolf Arnheim. 1954. *Art and visual perception: A psychology of the creative eye.* Univ of California Press.

[5] Shai Avidan and Ariel Shamir. 2007. *Seam Carving for Content-Aware Image Resizing* (1 ed.). Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3596711.3596776

[6] Greg J. Badros, Alan Borning, and Peter J. Stuckey. 2001. The Cassowary linear arithmetic constraint solving algorithm. *ACM Trans. Comput.-Hum. Interact.* 8, 4 (dec 2001), 267–306. https://doi.org/10.1145/504704.504705

[7] Dimitris Bertsimas and John N Tsitsiklis. 1997. *Introduction to linear optimization.* Vol. 6. Athena Scientific Belmont, MA.

[8] Y. Braha and B. Byrne. 2011. *Creative Motion Graphic Titling for Film, Video, and the Web.* Focal Press. https://books.google.com/books?id=cP398FPOy60C

[9] Stuart K Card, Jock D Mackinlay, and George G Robertson. 1991. A morphological analysis of the design space of input devices. *ACM Transactions on Information Systems (TOIS)* 9, 2 (1991), 99–122.

[10] Peggy Chi, Tao Dong, Christian Frueh, Brian Colonna, Vivek Kwatra, and Irfan Essa. 2022. Synthesis-Assisted Video Prototyping From a Document. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology* (Bend, OR, USA) *(UIST '22).* Association for Computing Machinery, New York, NY, USA, Article 16, 10 pages. https://doi.org/10.1145/3526113.3545676

[11] Peggy Chi, Nathan Frey, Katrina Panovich, and Irfan Essa. 2021. Automatic Instructional Video Creation from a Markdown-Formatted Tutorial. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '21).* Association for Computing Machinery, New York, NY, USA, 677–690. https://doi.org/10.1145/3472749.3474778

[12] Peggy Chi, Joyce Liu, Jason Linder, Mira Dontcheva, Wilmot Li, and Bjoern Hartmann. 2013. Democut: generating concise instructional videos for physical demonstrations. In *Proceedings of the 26th annual ACM symposium on User interface software and technology.* 141–150.

[13] Peggy Chi, Zheng Sun, Katrina Panovich, and Irfan Essa. 2020. Automatic video creation from a web page. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology.* 279–292.

[14] Niraj Ramesh Dayama, Simo Santala, Lukas Brückner, Kashyap Todi, Jingzhou Du, and Antti Oulasvirta. 2021. Interactive Layout Transfer. In *Proceedings of the 26th International Conference on Intelligent User Interfaces* (College Station, TX, USA) *(IUI '21).* Association for Computing Machinery, New York, NY, USA, 70–80. https://doi.org/10.1145/3397481.3450652

[15] Niraj Ramesh Dayama, Kashyap Todi, Taru Saarelainen, and Antti Oulasvirta. 2020. GRIDS: Interactive Layout Design with Integer Programming. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (<conf-loc>, <city>Honolulu</city>, <state>HI</state>, <country>USA</country>, </conf-loc>) *(CHI '20).* Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376553

[16] FX Elements. 2024. Getting Started with Video Overlays. https://www.fxelements.com/guides/getting-started-with-video-overlays Accessed: 2024-12.

[17] Mark Everingham, Josef Sivic, and Andrew Zisserman. 2006. "Hello! My name is... Buffy"–Automatic Naming of Characters in TV Video.. In *BMVC*, Vol. 2. Citeseer, 6.

[18] Marshal Carper. Social Media Examiner. 2019. A 6-Step Workflow to Create Video for Multiple Platforms. https://www.socialmediaexaminer.com/6-step-workflow-create-video-multiple-platforms/ Accessed: 2024-12.

[19] Ran Gal, Lior Shapira, Eyal Ofek, and Pushmeet Kohli. 2014. FLARE: Fast layout for augmented reality applications. In *2014 IEEE international symposium on mixed and augmented reality (ISMAR)*. IEEE, 207–212.

[20] Google. 2024. *MediaPipe.* https://github.com/google-ai-edge/mediapipe Accessed: 2024-09.

[21] Yandong Guo, Zhongliang Deng, Xiaodong Gu, Zhibo Chen, Quqing Chen, and Charles Wang. 2008. Aspect Ratio Conversion Based on Saliency Model. In *2008 Congress on Image and Signal Processing*, Vol. 4. 92–96. https://doi.org/10.1109/CISP.2008.714

[22] Stephen M Hart and Liu Yi-Hsin. 1995. The application of integer linear programming to the implementation of a graphical user interface: a new rectangular packing problem. *Applied mathematical modelling* 19, 4 (1995), 244–254.

[23] Qi Huangfu, Lukas Schork, Michael Feldmeier, Leona Gottwald, Julian Hall, and Ivet Galabova. 2024. *HiGHS: High-performance parallel linear optimization software.* University of Edinburgh. https://highs.dev Version 1.5.3.

[24] Bernd Huber, Hijung 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.* 1–11.

[25] Eakta Jain, Yaser Sheikh, Ariel Shamir, and Jessica Hodgins. 2015. Gaze-Driven Video Re-Editing. *ACM Trans. Graph.* 34, 2, Article 21 (mar 2015), 12 pages. https://doi.org/10.1145/2699644

[26] Yue Jiang, Ruofei Du, Christof Lutteroth, and Wolfgang Stuerzlinger. 2019. ORC Layout: Adaptive GUI Layout with OR-Constraints. 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, 1–12. https://doi.org/10.1145/3290605.3300643

[27] Topi Kaaresoja, Stephen Brewster, and Vuokko Lantz. 2014. Towards the Temporally Perfect Virtual Button: Touch-Feedback Simultaneity and Perceived Quality in Mobile Touchscreen Press Interactions. *ACM Trans. Appl. Percept.* 11, 2, Article 9 (jun 2014), 25 pages. https://doi.org/10.1145/2611387

[28] Steven D Katz. 1991. *Film Directing Shot by Shot: Visualizing from Concept to Screen.* Michael Wiese Productions.

[29] Johannes Kiess, Benjamin Guthier, Stephan Kopf, and Wolfgang Effelsberg. 2012. SeamCrop: changing the size and aspect ratio of videos *(MoVid '12)*. Association for Computing Machinery, New York, NY, USA, 13–18. https://doi.org/10.1145/2151677.2151681

[30] Jeongyeon Kim, Yubin Choi, Minsuk Kahng, and Juho Kim. 2022. FitVid: Responsive and Flexible Video Content Adaptation *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 501, 16 pages. https://doi.org/10.1145/3491102.3501948

[31] Joy Kim, Mira Dontcheva, Wilmot Li, Michael S. Bernstein, and Daniela Steinsapir. 2015. Motif: Supporting Novice Creativity through Expert Patterns. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 1211–1220. https://doi.org/10.1145/2702123.2702507

[32] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning.* PMLR, 957–966.

[33] Mackenzie Leake, Abe Davis, Anh Truong, and Maneesh Agrawala. 2017. Computational video editing for dialogue-driven scenes. *ACM Trans. Graph.* 36, 4, Article 130 (jul 2017), 14 pages. https://doi.org/10.1145/3072959.3073653

[34] Mackenzie Leake and Wilmot Li. 2024. ChunkyEdit: Text-first video interview editing via chunking. In *Proceedings of the 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

[35] Mackenzie Leake, Hijung Valentina Shin, Joy O Kim, and Maneesh Agrawala. 2020. Generating Audio-Visual Slideshows from Text Articles Using Word Concreteness. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), 1–11.

[36] Vladimir Likic. 2008. The Needleman-Wunsch algorithm for sequence alignment. *Lecture given at the 7th Melbourne Bioinformatics Course, Bi021 Molecular Science and Biotechnology Institute, University of Melbourne* (2008), 1–46.

[37] Xingyu" Bruce" Liu, Vladimir Kirilyuk, Xiuxiu Yuan, Alex Olwal, Peggy Chi, Xiang" Anthony" Chen, and Ruofei Du. 2023. Visual captions: augmenting verbal communication with on-the-fly visuals. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems.* 1–20.

[38] Meinard Müller. 2007. Dynamic time warping. *Information retrieval for music and motion* (2007), 69–84.

[39] Jakob Nielsen. 1994. *Usability engineering.* Morgan Kaufmann.

[40] Aziz Niyazov, Barrett Ens, Kadek Ananta Satriadi, Nicolas Mellado, Loïc Barthe, Tim Dwyer, and Marcos Serrano. 2023. User-driven constraints for layout optimisation in augmented reality. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems.* 1–16.

[41] OpenAI. 2023. *GPT-4.* https://openai.com/gpt-4 Accessed: 2024-08.

[42] Amy Pavel, Dan B Goldman, Björn Hartmann, and Maneesh Agrawala. 2015. Sceneskim: Searching and browsing movies using synchronized captions, scripts and plot summaries. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology.* 181–190.

[43] Amy Pavel, Colorado Reed, Björn Hartmann, and Maneesh Agrawala. 2014. Video digests: a browsable, skimmable format for informational lecture videos. In *UIST*, Vol. 10. Citeseer, 2642918–2647400.

[44] Steven Piantadosi. 2023. Modern language models refute Chomsky's approach to language. *Lingbuzz Preprint, lingbuzz* 7180 (2023).

[45] Steve Rubin, Floraine Berthouzoz, Gautham J Mysore, Wilmot Li, and Maneesh Agrawala. 2013. Content-based tools for editing audio stories. In *Proceedings of the 26th annual ACM symposium on User interface software and technology.* 113–122.

[46] Nazmus Saquib, Rubaiat Habib Kazi, Li-Yi Wei, and Wilmot Li. 2019. Interactive body-driven graphics for augmented video performance. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems.* 1–12.

[47] Pavel Senin. 2008. Dynamic time warping algorithm review. *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA* 855, 1-23 (2008), 40.

[48] Than Htut Soe and Marija Slavkovik. 2022. A content-aware tool for converting videos to narrower aspect ratios. In *Proceedings of the 2022 ACM International Conference on Interactive Media Experiences* (Aveiro, JB, Portugal) *(IMX '22)*. Association for Computing Machinery, New York, NY, USA, 109–120. https://doi.org/10.1145/3505284.3529970

[49] Takehide Soh, Katsumi Inoue, Naoyuki Tamura, Mutsunori Banbara, and Hidetomo Nabeshima. 2010. A SAT-based method for solving the two-dimensional strip packing problem. *Fundamenta Informaticae* 102, 3-4 (2010), 467–487.

[50] Ralf Steinmetz. 1996. Human perception of jitter and media synchronization. *IEEE Journal on selected Areas in Communications* 14, 1 (1996), 61–72.

[51] Amanda Swearngin, Chenglong Wang, Alannah Oleson, James Fogarty, and Amy J Ko. 2020. Scout: Rapid exploration of interface layout alternatives through high-level design constraints. In *Proceedings of the 2020 CHI conference on human factors in computing systems.* 1–13.

[52] Roy Thompson and Christopher J Bowen. 2009. *Grammar of the Edit.* Taylor & Francis.

[53] 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

[54] Anh Truong and Maneesh Agrawala. 2019. A Tool for Navigating and Editing 360 Video of Social Conversations into Shareable Highlights.. In *Graphics Interface.* 14–1.

[55] Anh Truong, Floraine Berthouzoz, Wilmot Li, and Maneesh Agrawala. 2016. Quickcut: An interactive tool for editing narrated video. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology.* 497–507.

[56] Anh Truong, Peggy Chi, David Salesin, Irfan Essa, and Maneesh Agrawala. 2021. Automatic generation of two-level hierarchical tutorials from instructional makeup videos. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems.* 1–16.

[57] International Telecommunication Union. 2004. *Tolerable round-trip time delay for sound-programme and television broadcast programme inserts.* Technical Report. United Nations Economic and Social Council.

[58] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O Wobbrock. 2018. $ Q: A super-quick, articulation-invariant stroke-gesture recognizer for low-resource devices. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services.* 1–12.

[59] 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

[60] Oliver Wang, Christopher Schroers, Henning Zimmer, Markus Gross, and Alexander Sorkine-Hornung. 2014. VideoSnapping: interactive synchronization of multiple videos. *ACM Trans. Graph.* 33, 4, Article 77 (jul 2014), 10 pages. https://doi.org/10.1145/2601097.2601208

[61] Sitong Wang, Samia Menon, Tao Long, Keren Henderson, Dingzeyu Li, Kevin Crowston, Mark Hansen, Jeffrey V Nickerson, and Lydia B Chilton. 2024. ReelFramer: Human-AI co-creation for news-to-video translation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems.* 1–20.

[62] Wikipedia. 1964. *Old-Fashioned Apple Pie.* Public Domain Recepies.

[63] Wikipedia. 2024. *Lemon.* Wikimedia Foundation.
[64] Haijun Xia, Jennifer Jacobs, and Maneesh Agrawala. 2020. Crosscast: adding visuals to audio travel podcasts. In *Proceedings of the 33rd annual ACM symposium on user interface software and technology.* 735–746.
[65] Jiahong Yuan and Mark Liberman. 2008. Speaker Identification on the SCOTUS Corpus. In *Proceedings of Acoustics '08*, Vol. 123. Paris, France. http://www.acoustics08.org
[66] Clemens Zeidler, Christof Lutteroth, Wolfgang Sturzlinger, and Gerald Weber. 2013. The auckland layout editor: An improved gui layout specification process. In *Proceedings of the 26th annual ACM symposium on User interface software and technology.* 343–352.

Appendix A: Considering Gestures in Temporal Alignment

During the process of recording the video takes during our evaluation and watching the output retargeted videos, we notice that a subset of our performers used gestures to emphasize certain parts of speech, particularly during important content where an asset is likely to appear. An example scenario could be pointing to the head during a "light-bulb" moment corresponding with a light-bulb emoji. We decide to use hand gesture data to fine-tune the temporal alignment of assets and discuss its implications after the algorithm extension.

A.1 Algorithm Extension: Gesture Alignment

Body tracking yields per-frame positions of the head and body of the performer in our videos. For gesture alignment, we use the hand positions as points of interest (POI) for gesture detection. Established algorithms such as the $Q Super-Quick Recognizer [58] are commonly used for gesture recognition, but this class of algorithms intentionally discards velocity data, which we found was important for identifying hand gestures. Instead, we consider gesture alignment to be a similar problem to transcript alignment.

We re-sample the hand position trajectory with constant arc length and filter it with a discrete Gaussian kernel. Then, for each trajectory sample, we *tokenize* the sample by converting it into one of a discrete set of tokens based on its trajectory: S (stopped), O (moving forward), L (turning left), R (turning right), and X (off-screen). The statistics of the tokens are very different from those of words, and they lack semantic meaning. We found a similar LLM-based approach to the one we used for transcript alignment had limited success for this problem. We instead use a voting scheme where each token in the target video "votes" for the location of an asset appearance based on its offset to an asset appearance in the source video.

Because gestures are often re-used throughout a performance, they are not as uniquely localized as speech. Therefore we do not search for the best gesture match throughout the entire video. Instead we use transcript based temporal alignment first to identify a 2 second window in which gesture alignment is performed to refine the alignment.

A.2 Gesture Alignment Results

We observed that the majority of asset appearances in the dataset collected for our technical evaluation did not correspond with a major gestural movement because we did not explicitly tell participants to use gestures. In fact, several participants did not gesture at all with their hands during their performances. However, as shown

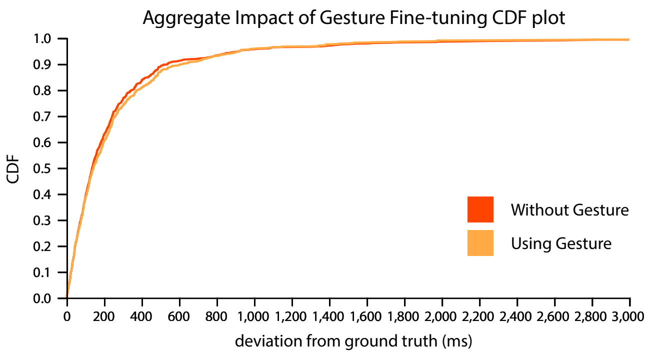


Figure 15: We plot the CDFs of temporal alignment computed with and without gesture data and find no significant impact in aggregate. While there is no improvement, there is no meaningful degradation of performance either. Fine-tuning with gesture data only impacts a small subset of retargeting events, which do not show up in this large-scale analysis. However, for cases in which actors carefully use gestures to emphasize speech, we see a significant improvement in alignment quality with gesture fine tuning (Fig. 16).

in Fig. 16, when the speaker does use intentional gestures, gesture fine tuning can help time the appearance of an asset to these gestures. This contributes to a "snappy" effect where the appearance of an asset looks coordinated with a gesture. We plot the CDF of overall deviation (all 765 retargeting events combined) from the ground truth computed with and without the gesture fine-tuning in Fig. 15. In aggregate, we see little effect of this fine tuning, but it leads to better, snappier retargeting for specific performances (Fig. 16).

Appendix B: Derivation of MILP for Spatial Alignment

We define decision variables x_a, y_a as the top-left coordinate of each asset's $w_a \times h_a$ bounding box inside the $\mathbb{W} \times \mathbb{H}$ video frame. We now explain each component of the MILP outline presented in subsection 4.2:

$$\begin{aligned}
\text{min} \quad & c_{\text{non-overlap}} + c_{\text{alignments}} + c_{\text{visual weight}} \\
\text{subject to} \quad & \Pi_{ij}, \Gamma_{ij} \text{ non-overlap constraints} \\
& t_{ax}, t_{ay} \text{ alignment constraints} \\
& p \text{ padding constraints} \\
& p \text{ ROI fixed constraints}
\end{aligned}$$

For the non-overlap problem, two rectangles that do not overlap have a partition line on the x- and/or y-axis. We introduce *binary* decision variables $\Pi_{ij} \in 0, 1$ to reflect that a_i is completely to the left of a_j with padding p through the following constraints

$$\begin{aligned}
\Pi_{ij} + \Pi_{ji} &\leq 1 \\
x_j &\geq p + x_i + w_i + \mathbb{W} \cdot (\Pi_{ij} - 1) \\
x_i &\geq p + x_j + w_j + \mathbb{W} \cdot (\Pi_{ji} - 1)
\end{aligned}$$

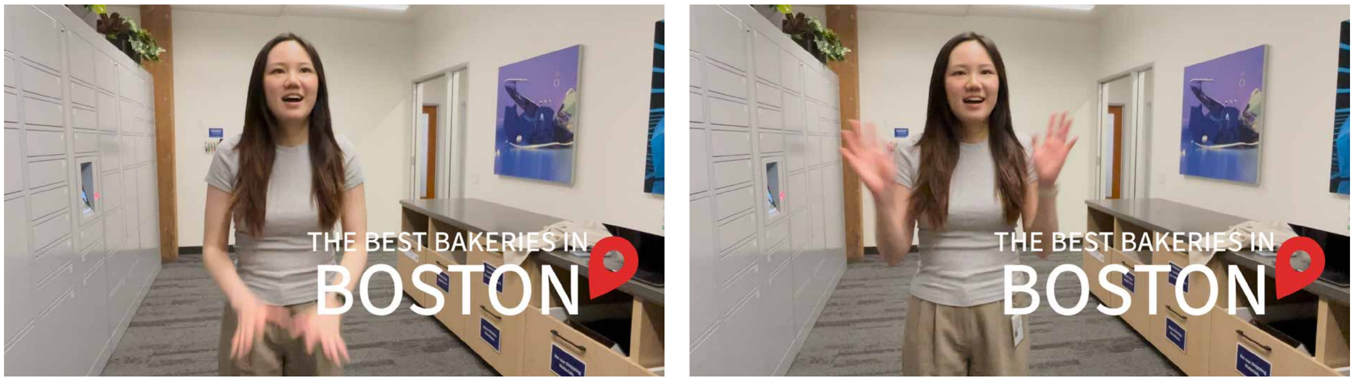


Figure 16: Gestural fine tuning: On the left, the first frame of a "Best Bakeries in Boston" graphic is retargeted onto a target video take *without gestural fine-tuning*. The actor is just about to raise their hands to wave, but the graphic appears early. On the right, the first frame of the same graphic is retargeted *with gestural fine-tuning*. The actor has raised their hands to wave, and the graphic appears in coordination with the gesture. The difference between the two appearance times in this example was ~200ms. The gestural fine tuning helps the asset better align with the performance in this case.

To reflect that a_i is completely above of a_j with padding p, we define y-coordinate/height variables the same way we did with x-coordinate/width variables through the following constraints:

$$\Gamma_j^i + \Gamma_i^j \le 1$$
$$y_j \ge p + y_i + h_i + \mathbb{H} \cdot (\Gamma_i^j - 1)$$
$$y_i \ge p + y_j + h_j + \mathbb{H} \cdot (\Gamma_j^i - 1)$$

And to guarantee a horizontal and/or vertical partition exists, thereby preventing overlap, we bound our Π's and Γ's with the constraint $1 \le \Pi_{ij} + \Pi_{ji} + \Gamma_j^i + \Gamma_i^j \le 2$. As mentioned in the main paper, this approach to preventing overlap is standard in the MILP literature [15, 22]. We add these constraints and decision variables for every asset with every ROI, and for every pair of assets that are simultaneously visible. However like with the geometric rules, if two bounding boxes were originally overlapping we interpret the user was intentional and skip adding constraints between them. We also prevent assets from going off-screen with additional padding around the border via constraints $p \le x_a \le \mathbb{W} - w_a - p$ and $p \le y_a \le \mathbb{H} - h_a - p$. This approach is purely feasibility-driven as it only enforces the existence of an arrangement where no overlap exists. Therefore our objective function's coefficients are 0 for all Π and Γ, i.e. $c_{\text{non-overlap}} = 0$, since cost is not needed in this construction.

Next, we minimize distance of each x_a, y_a to the geometric rule's (x'_a, y'_a) with an additional objective min $|x_a - x'_a| + |y_a - y'_a|$ by bounding these terms with auxiliary decision variables t_{ax}, t_{ay} with the following setup:

$$\begin{aligned} &\min |x_a - x'_a| + |y_a - y'_a| \\ &\Downarrow \\ &\min t_{ax} + t_{ay} \\ \text{subject to } &x_a - x'_a \le t_{ax}, \quad -x_a + x'_a \le t_{ax} \\ &y_a - y'_a \le t_{ay}, \quad -y_a + y'_a \le t_{ay} \end{aligned}$$

Note that x'_a and y'_a are constants from the initial rule-based spatial repositioning output. All auxiliary variables have objective function coefficient 1 to minimize total position changed from the geometric output step *across* a group of assets. Therefore $c_{\text{alignments}} = \sum_a 1 \cdot t_{ax} + 1 \cdot t_{ay}$. Combined with the prior Π and Γ constraints, the MILP finds the layout closest to the rule-based output that has no overlap.

Finally, coefficients c_{ax}, c_{ay} on variables x_a and y_a dictate a cost directly on the asset positions. We use $c_{ax} = 0$ and $c_{ay} = -\log(w_a \cdot h_a) \cdot \alpha$ to encourage moving larger assets towards the bottom of the video with $\alpha = 0.01$, following the metaphor of visual weight. Therefore, $c_{\text{visual weight}} = \sum_a -\log(w_a \cdot h_a) \cdot \alpha$. However, the procedure to fix a ROI to a position (f_x, f_y) is still achieved through constraints—we convert $f_x = x_a, f_y = y_a$ equalities into double inequalities $f_x \le x_a \le f_x$ and $f_y \le y_a \le f_y$ that can be easily encoded into the $Ax \le b$ canonical LP form. In the case that a MILP is infeasible, we relax the program by sequentially removing categories of constraints as described in subsection 4.2.