VURF: A General-purpose Reasoning and Self-refinement Framework for Video Understanding

Ahmad Mahmood[1] **Ashmal Vayani**[2] **Muzammal Naseer**[3] **Salman Khan**[4,5]
amahmood@ethz.ch

Fahad Shahbaz Khan[4,6]

[1]ETH Zurich [2]University of Central Florida [3]Khalifa University, UAE
[4]Mohamed Bin Zayed University of AI, UAE [5]Australian National University, Australia
[6]Linköping University, Sweden

Abstract

Recent studies have demonstrated the effectiveness of Large Language Models (LLMs) as reasoning modules that can deconstruct complex tasks into more manageable sub-tasks, particularly when applied to visual reasoning tasks for *images*. In contrast, this paper introduces a *Video Understanding and Reasoning Framework* (VURF) based on the reasoning power of LLMs. Ours is a novel approach to extend the utility of LLMs in the context of video tasks, leveraging their capacity to generalize from minimal input and output demonstrations within a contextual framework. We harness their contextual learning capabilities by presenting LLMs with pairs of instructions and their corresponding high-level programs to generate executable visual programs for video understanding. To enhance the program's accuracy and robustness, we implement two important strategies. *Firstly,* we employ a feedback-generation approach, powered by GPT-3.5, to rectify errors in programs utilizing unsupported functions. *Secondly*, taking motivation from recent works on self-refinement of LLM outputs, we introduce an iterative procedure for improving the quality of the in-context examples by aligning the initial outputs to the outputs that would have been generated had the LLM not been bound by the structure of the in-context examples. Our results on several video-specific tasks, including visual QA, video anticipation, pose estimation, and multi-video QA, illustrate these enhancements' efficacy in improving the performance of visual programming approaches for video tasks.

1 Introduction

In recent years, the vision community has developed highly efficient specialized models for various video understanding tasks, including Video Question Answering Antol et al. [2015], Action Anticipation Girdhar and Grauman [2021] and Pose Estimation Toshev and Szegedy [2014], Koprinska and Carrato [2001], Yilmaz et al. [2006], Gammulle et al. [2019]. Despite such advancements, video models usually offer isolated visual comprehension capabilities in the form of narrow task-specific models. Such specialized models limited to individual tasks struggle to offer a comprehensive, adaptable, and scalable understanding of videos for complex reasoning. Although the current off-the-shelf models can perform specific well-defined tasks, they require specialized and comprehensive datasets to effectively train dedicated models for each task, which is not feasible for general-purpose complex reasoning problems. Furthermore, individual off-the-shelf vision models for each video-understanding task necessitate a distinct framework with unique model configurations. This problem

Workshop on Video-Language Models at 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

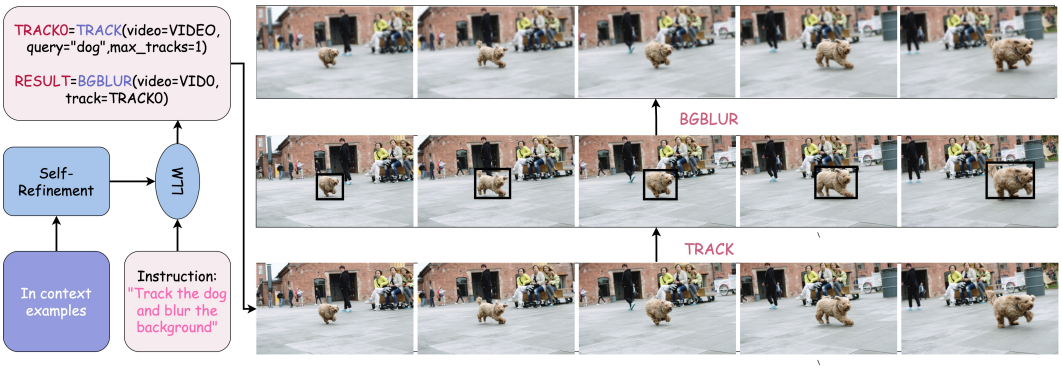


Figure 1: **An overview of the VURF pipeline:** Figure demonstrates how a complex query regarding video editing is broken down in VURF to arrive at the final edited result. *Best viewed in zoom.*

underscores the requirement for a uniform reasoning framework that offers plug-and-play architecture, capable of leveraging any pre-trained computer vision model, enabling seamless execution of a given task.

To address these challenges, we adopt a strategy of decomposing broad video-understanding tasks into more manageable sub-tasks, each of which can be solved by executing task-specific models, and subsequently consolidating the results. Our framework is motivated by the observation that complex tasks can be effectively solved by executing an intermediate sequence of sub-tasks, collectively working in sequence to solve the more challenging problem. This process of task decomposition necessitates a reasoning module capable of discriminating the necessary steps for task execution. Large Language Models (LLMs) emerge as promising candidates for this role. Recent works on Visual Programming Gupta and Kembhavi [2023] have demonstrated the effectiveness of LLMs in breaking down complex tasks into smaller, more manageable components that can be tackled by specialized computer vision models Wu et al. [2023], Yang et al. [2023], Ruan et al. [2023]. In our study, we demonstrate the utility of such a reasoning module in tackling specific challenges within the domain of video understanding. We also show that building such an approach on top of the existing off-the-shelf video models can significantly enhance task performance.

While LLMs demonstrate competence in serving as reasoning modules, they are not immune to errors and limitations. One notable deficiency is their vulnerability to hallucinations induced by contextual information, without the means to self-correct based on task-agnostic knowledge Sun et al. [2023]. To address this concern, we draw inspiration from recent research demonstrating the efficacy of self-refinement processes in enabling LLMs to enhance their outputs Madaan et al. [2024], Feng et al. [2024], akin to the way humans engage in self-correction. Specifically, we propose a feedback-generation mechanism that evaluates the LLM's output and employs this feedback to prompt the LLM to refine its output. Our findings substantiate the effectiveness of this approach in elevating the performance of video-based reasoning approaches. Our main contributions are as follows:

- The first generic visual reasoning framework for video understanding consolidates multiple task-specific, domain-specialized video models to answer any video-related user queries.
- Using in-context learning, we align LLM behavior for decomposing a given complex task into multiple sub-tasks easily solvable using existing task-specific video models.
- Our proposed self-refinement strategy helps avoid errors in the programs (outlining subtask decomposition) generated by the LLM and boosts the performance by iterative refining the generated program. The proposed framework is shown to boost performance for tasks such as visual question answering for videos in complex reasoning scenarios.

2 Related Work

Video Understanding: Video understanding focuses on teaching machines to understand and analyze visual content. One crucial task is to recognize and localize different actions in the video Reddy et al.

[2023]. Numerous algorithms have been developed to cater the video understanding tasks such as Video Swin Transformer Liu et al. [2022], VideoMAE Tong et al. [2022], C2D Wang et al. [2018], MViT-V2 Li et al. [2022], STGCN++ Yu et al. [2017], and ViViT Arnab et al. [2021] that achieve high accuracy on SOTA datasets Sigurdsson et al. [2016], Gao et al. [2017], Gu et al. [2018], Deliege et al. [2021], Huang et al. [2020], Girdhar and Grauman [2021], Sadhu et al. [2021].

Video understanding tasks facilitate the efficient handling of diverse information modalities Li et al. [2020] and various tasks have been introduced to test the capabilities of the methods for video understanding such as retrieving temporal and spatial information Zhang et al. [2023] and answering natural language questions from the video i.e., Video Question Answering (VQA) Yang et al. [2003], Lei et al. [2018]. Other methods such as SeViLA Yu et al. [2024] and iVQA Lin et al. [2023b], adapt Localizer and Answerer for both QA and temporal key-frame localization which are then extended to zero-shot VQA Song et al. [2023], Yang et al. [2022], Lin et al. [2023a]. Another challenging task in video understanding is to localize the starting and ending time of the video segment that corresponds to the input query i.e., video grounding Chen et al. [2018], Yang et al. [2022], Zeng et al. [2020]. It can solve various video understanding tasks such as Temporal Action Recognition Chen et al. [2019], spatio-temporal video grounding Zhang et al. [2020], and Action Recognition Carreira and Zisserman [2017]. Our work on video programming provides a pipeline for various video understanding and reasoning tasks leveraging off-the-shelf SOTA models and the reasoning capabilities of LLMs for each sub-task.

Visual Programming: Visual Programming leverages Language Models (LLMs) to break down complex vision-understanding tasks into simpler sub-tasks executed sequentially, improving responses with in-context examples and prompts. Recent advancements, such as VisProg Gupta and Kembhavi [2023], enhance visual task performance by increasing in-context examples, replacing high-error modules with off-the-shelf models, and refining instructions. Our work, VURF, is the first generic reasoning framework for video, utilizing LLMs' self-critique and refining visual programs to address errors. While zero-shot models like Flamingo Alayrac et al. [2022] can adapt to new tasks without fine-tuning, they struggle with generalization, unlike VURF, which leverages SOTA models for downstream tasks.

Self Refinement: LLMs demonstrate a special ability to enhance their outputs just like how humans re-evaluate, refine, and reiterate the text that they have initially written. The same LLM when used to identify the potential issues with the output and through the generator, refiner, and feedback provider, it even has the potential to improve responses generated by SOTA LLMs like GPT4 Madaan et al. [2023]. Other methods include using external tools like search engines to rectify the output Madaan et al. [2023], detecting hallucinated outputs Gou et al. [2023], Evans et al. [2021], Zhou et al. [2020], Golovneva et al. [2022], using natural language feedback Saunders et al. [2022] to improve the initial generated response.

Leveraging the capabilities of self-debugging, language models have been able to debug their predicted program via few-shot demonstrations Chen et al. [2023], use of relevant in-context examples to generate efficient response Wang et al. [2023], Thawakar et al. [2024], and have contributed in various domains such as code generation and its applications Yu et al. [2019], summarization Campos and Shern [2022], and program synthesis Le et al. [2022], Kim et al. [2023].

While these approaches improve the LLM's response via recursive feedback methods, these methods require continuous refinement of any output generated. Our self-refinement approach focuses on the critique and refinement of in-context examples and we show that just a pre-defined set of in-context examples can boost the performance of a visual programming approach (Table 1).

3 Methodology

In contrast to conventional task-specific models that exhibit limitations in addressing complex reasoning challenges, the Video Understanding and Reasoning Framework (VURF) seeks to utilize the reasoning power of LLMs to deconstruct complex video-related queries into a series of sub-tasks (video programs). By executing these sub-tasks sequentially, we can culminate to arrive at the final response. Moreover, VURF allows seamless integration of new visual models in a plug-and-play manner and also employs self-critique mechanisms to mitigate LLM's hallucinations and judgment errors. An overview of our approach is shown in Fig. 2, and we explain the approach in detail below.

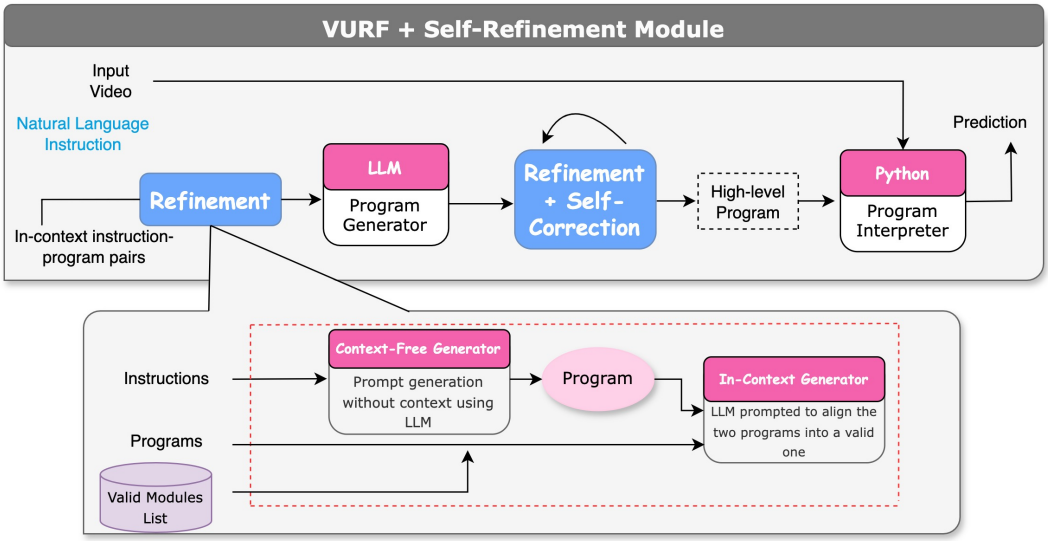


Figure 2: **Video Understanding and Reasoning Framework (VURF) pipeline. Top:** figure shows the main approach of VURF with the added self-correction module. **Bottom:** figure shows the self-refinement module.

3.1 Video Reasoning LLMs

Role of Large Language Models. Large language models (LLMs), exemplified by GPT-3 Brown et al. [2020] and GPT-4 Achiam et al. [2023], have demonstrated an impressive in-context learning capability (ICL) that does not necessitate fine-tuning. ICL aims to extend the understanding of LLMs to novel scenarios using a restricted set of input and output demonstrations within the relevant context. In this work, we leverage GPT-3.5 to generate visual programs that solve video reasoning tasks involving natural language instructions. The LLMs efficiently perform video reasoning tasks by avoiding direct video processing. Instead, it formulates logical sequence flows that decompose complex tasks into simpler sub-tasks. This approach enhances efficiency and allows the model to navigate the intricacies of video-related challenges.

Prompting. We prompt GPT-3.5 with in-context examples which consist of pairs of instruction and the associated programs that the LLM is expected to generate. The programs follow a generic structure where each line of the program includes the name of a module, the module's input argument names and their values, and an output variable name. As output variables in a specific step are used later for another step they follow a general structure which the LLM learns:

```
OUTPUT0=FUNC0(video=VIDEO,...)
OUTPUT1=FUNC1(arg0=OUTPUT0,...)
...
```

Given a set of these pairs and a new instruction, the LLM can generate a new program that follows the same structure and can thus be executed via our program interpreter.

3.2 Self-Refinement

One prominent limitation of a naive visual programming technique like Gupta and Kembhavi [2023] is the proneness of generating inaccurate information influenced by contextual cues, as well as lacking an inherent capacity for autonomous self-correction through task-agnostic knowledge. Two major issues arise with the LLM-generated program. Firstly, due to LLM hallucinations, the program might be using some function that is not supported by our interpreter. Secondly, the program may not break down the instruction into sub-tasks in an optimal fashion. We resolve these limitations by the below steps.

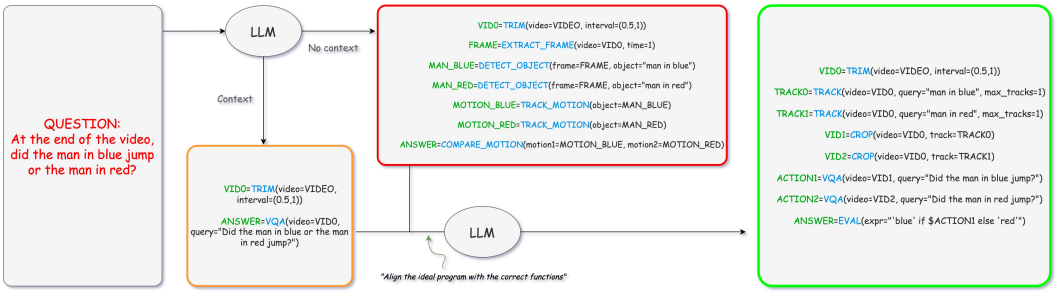


Figure 3: **Auto Self-Refinement** example. Two programs are generated: one with contextual examples and one without, but with added information for structural integrity. Both are then input into the Language Model (LLM) to generate a new program that aligns with the ideal while avoiding invalid functions.

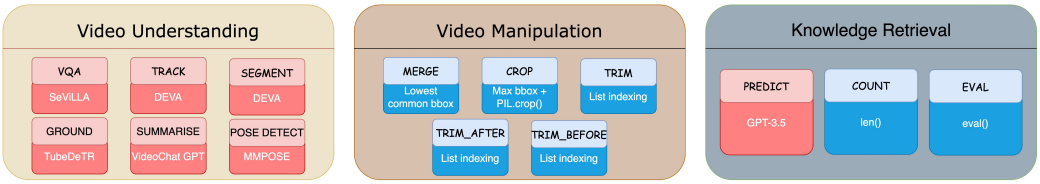


Figure 4: **Main Modules used by VURF.** The red boxes show modules that require a pre-trained model whereas the boxes are modules that require trivial functions.

Error Correction: To address the issue of a program utilizing a function unsupported by our interpreter, we employ a feedback generation approach, which notably leverages the power of GPT-3.5. We present the program to this module, alongside the available list of functions and their general usage, and inquire if the given program violates these constraints. If discrepancies are identified, the program is subsequently regenerated, incorporating the provided feedback as contextual information. This iterative process ensures error-free execution of the given instruction.

Auto-Refinement of In-Context Examples: The effectiveness of our approach hinges significantly on the quality of in-context examples. Therefore, it is imperative to refine these examples to enhance the module's performance Lu et al. [2023], Madaan et al. [2023]. To accomplish this, we implement a self-refinement procedure. Initially, given an instruction, I, and the initial program generated P, we input I to the LLM, prompting it to generate an improved program, P', without the inclusion of in-context examples. Subsequently, both P and P' are input into the LLM, enabling the generation of a refined program that aligns more closely with the LLM's reasoning, all while excluding the influence of in-context examples. Replicating this process for n instructions yields a set of n new in-context examples, enhancing the module's ability to perform tasks effectively. Note that as shown in Fig. 2, the auto self-refinement module can be applied to a single user query to iteratively improve the generated program as well. However, this is inefficient and costly and thus we show that even pre-refining the in-context program-instruction pairs can improve the performance of the VURF (Table 1). A concrete illustration of this process on a single program and instruction pair is provided in Fig. 3.

4 Tasks

Our video understanding and reasoning framework (VURF) aims to offer a versatile approach adaptable to various visual tasks. By integrating an interpreter component into an existing state-of-the-art (SOTA) vision model, VURF addresses four diverse challenges: Video Question Answering (VQA), Video Anticipation, Pose Estimation, and Multi-Video VQA. VURF functions by employing a large language model (LLM) as a reasoning module to generate a visual program. This program outlines a sequence of steps, each executed independently. The output of one step is fed as input to the next, creating a cohesive workflow.

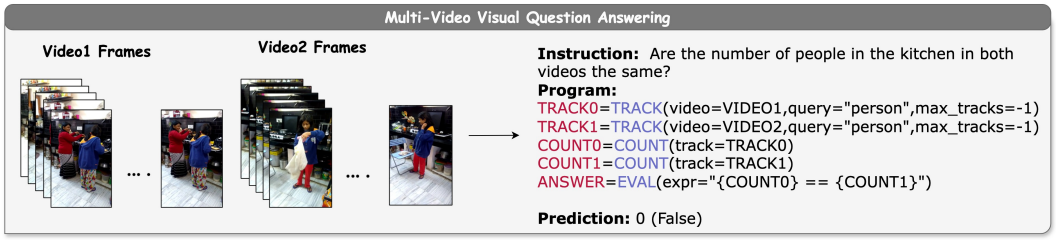


Figure 5: A qualitative example showing the Program steps in the **Multi-Video VQA task**. The programs provide a logical decomposition of the original complex tasks.

4.1 Video Question Answering

VQA is an important task in video comprehension that endeavors to connect natural language processing with video comprehension. The objective is to empower models to interpret the content of a video and respond to user queries related to the video content. Effective resolution of such queries often involves breaking down the overarching problem into manageable sub-problems.

For instance, consider a video of a man entering a room and then performing some action. If a human is asked, "What does the man do after entering the room?" they would visually inspect the video, locate the man, observe his actions by examining the video, and then deduce what the nature of his activities. This example illustrates the critical role of logical reasoning in decomposing complex questions into more manageable sub-queries, suitable for evaluation using off-the-shelf models. The inherent complexity of such tasks makes VQA an ideal candidate for a visual programming approach, given its primary function of decomposing intricate tasks into more manageable components. In the context of the aforementioned example, VURF, as demonstrated in Fig. 7 (Right), initially employs a GROUNDING model to identify an interval where the man enters the room. Subsequently, it would call the TRIMAFTER module to retrieve the relevant part of the video, and finally using the VQA module, it will engage in visual examination of the video to answer the question "Pick up towel". This approach makes the task more logical and interpretable, with possible explanations in case an output is wrong.

VURF also extends its functionality to Multi-Video Question Answering tasks which involves synthesizing information from two distinct videos to provide accurate responses to user queries. Our approach simplifies this intricate process by breaking it down into a sequence of steps executed by a language model. An example is shown in Fig. 5.

4.2 Pose Estimation

Pose Estimation identifies the position and orientation of human bodies in images or videos using a skeletal model and has been applied in areas like HCI, virtual reality, and clinical assessments Zheng et al. [2023], Erol et al. [2007], Escobar et al. [2019]. The process typically involves two stages: detecting joint orientations, as seen in tools like MMPose Sengupta et al. [2020] and OpenPose Cao et al. [2017], followed by using a model to estimate the pose from the skeletal representation Andriluka et al. [2014]. While these approaches are effective, they often struggle to generalize to unseen datasets.

In our Visual Programming approach, pose estimation is used to pre-process videos by tracking and cropping specific individuals, followed by pose classification. We utilize MMPose for keypoint detection and classify poses for tasks like fall detection, which can be extended to applications such as hazard or crime detection. For example, Fig. 7 shows pose tagging in a video, and Fig. 6 illustrates a fall detection scenario where relevant frames are trimmed, pose is detected, and the fall is classified.

4.3 Video Editing

Video editing is a crucial process in the post-production phase of film-making, television production, and other visual media industries Dancyger [2018]. It involves manipulating and rearranging video clips to create a coherent and engaging narrative or visual presentation. Video editing encompasses various tasks such as trimming, cutting, merging, and arranging video segments, as well as adding

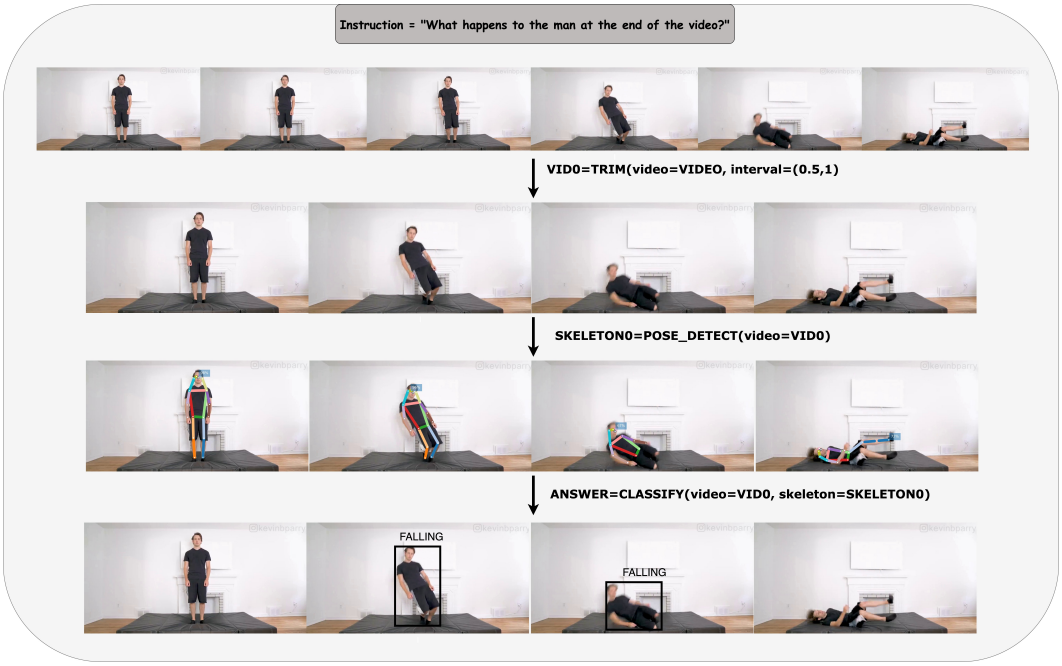


Figure 6: **Qualitative example** showing the program steps of the Pose Estimation task of VURF.

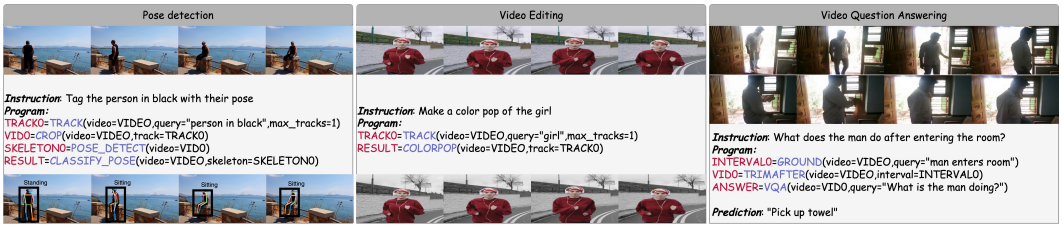


Figure 7: **Qualitative examples** demonstrating different use cases of VURF. **Left**: An example of Pose Detection that uses the visual program generated by VURF. **Middle**: A qualitative example of Video Editing that directly leverages VURF's list of modules. **Right**: An example of VQA task decomposition by manageable sub-tasks by our framework. *Best viewed in zoom.*

visual effects, transitions, and audio enhancements. In today's digital age, video editing is typically performed using specialized software applications that offer a wide range of tools and features to facilitate the editing process.

Our approach directly applies to Video Editing by breaking any instruction down into a series of sequential steps. For this task we employ the use of the MERGE, CROP, TRIM, BGBLUR, and COLORPOP functions. Some concrete qualitative examples can be found in Fig. 1 and 7 (Middle).

5 Experiments and Results

Our experiments encompass both quantitative and qualitative evaluation of the proposed video reasoning framework. In the quantitative experiments, we assess the performance impact of the proposed video programming approach built on a pre-trained model designed specifically for the Video Question Answering task. Additionally, we examine how the self-refinement approach affects the performance of the video programming approach, exploring the effects of altering the number of iterations on the model outputs. Finally, we present qualitative examples involving diverse tasks such as Pose Detection and Video Editing to highlight the efficacy of video programming in the context of video understanding tasks.

Table 1: ***Performance of VURF on Video Question Answering (zero-shot) compared to other existing models.*** Each result is evaluated on the validation set of the corresponding dataset.

Datasets	NextQA	STAR	Social-IQ-2.0	TrafficQA
InternVideo Wang et al. [2022]	50.2%	41.8%	30.1%	31.2%
ViperGPT Surís et al. [2023]	60.0%	40.3%	37.8%	35.7%
SeViLA Yu et al. [2024]	63.8%	44.3%	47.3%	39.1%
VURF	64.0%	47.2%	51.6%	43.5%

Table 2: ***Comparison with Visual Instruction Tuning.*** We finetune Video-ChatGPT on program-question pairs (which we use as in-context examples in our system) and then test their program-generating performance.

Dataset	VURF (ours)	Visual Instruction Tuning
STAR	44.5%	22.3%

5.1 VQA Evaluation

Diverse approaches to video question answering (VQA) have been investigated, yielding promising outcomes. However, the SeViLA framework Yu et al. [2024], has emerged as a distinguished approach, integrating temporal keyframe localization and question answering by employing a unified image-language model (BLIP2)Li et al. [2023]. We incorporate SeViLA as the VQA module, showcasing improved performance in the zero-shot video question-answering task with our VURF. The assessment is conducted on four benchmark datasets: STAR, NextQA, Social-IQ QA, and TrafficQA Xu et al. [2021].

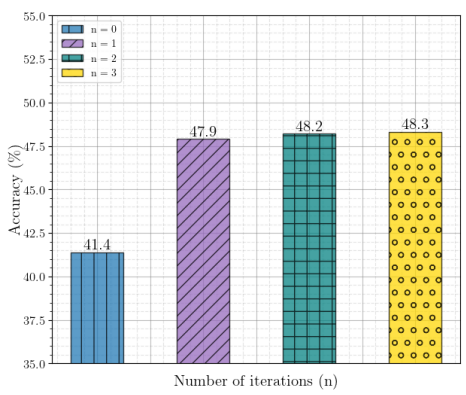
We conduct a comprehensive evaluation, commencing with the application of the base SeViLA zero-shot model on each dataset. Subsequently, we assess the effectiveness of the video programming approach on the same validation set. For the latter, a curated set of in-context examples is manually assembled for ICL, tailored to the specific characteristics of each dataset.

Several approaches, including InternVideo Wang et al. [2022] and ViperGPT Surís et al. [2023], have been proposed to address the challenge of Visual Question Answering alongside SeViLA. However, as illustrated in Table 1, VURF outperforms all three in zero-shot video question answering. VURF offers a significant advantage over SeViLA and InternVideo by enabling reasoning through instructions rather than functioning as a black box, facilitating self-improvement. This capability proves advantageous over ViperGPT, which, despite employing the reasoning power of LLMs, is outperformed by VURF. The primary reason for this discrepancy is ViperGPT's susceptibility to contextual hallucinations. We demonstrate that integrating the self-refinement of in-context program-instruction pairs and incorporating a simple error correction module substantially enhances the performance of a visual programming approach.

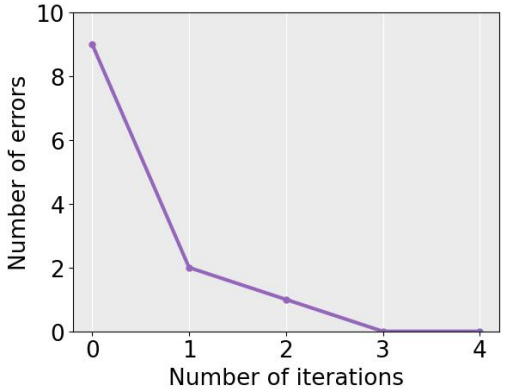
5.2 VQA Ablations

ViperGPT Surís et al. [2023] diverges from VURF by not incorporating self-refinement mechanisms. This distinction is noteworthy as it highlights differing approaches to handling instruction comprehension within the context of VQA. Sole reliance on pre-trained language models without iterative refinement mechanisms makes the approach susceptible to contextual hallucinations. Our ablations (see Table 3) clearly demonstrate that integrating the self-refinement of in-context program-instruction pairs and incorporating a simple error correction module substantially enhances the performance of a visual programming approach.

Moreover, we also compare our method with visual instruction tuning Liu et al. [2023]. To this end, we finetune a LlaVA-based model (trained using Visual instruction tuning) called Video-ChatGPT Maaz et al. [2024] on question-program pairs and test the performance of generated programs on the STAR VQA dataset. VURF outperforms the visual instruction tuning method (Table 2), even when the latter has access to more data during the training phase compared to the number of in-context examples used by VURF. VURF's superior performance stems from its dynamic adaptation to various scenarios, leveraging contextual cues to manage tasks on the fly.



(a) **Accuracy plotted against the number of iterations of self-refinement**.



(b) **Number of errors plotted against the number of iterations of self-refinement.**

Figure 8: The impact of the self-refinement stage on quality of video programs.

Table 3: ***Ablations demonstrating the effectiveness of the refinement pipeline.*** Results that do not use the error correction module, are such that if a syntactically incorrect program is generated it is considered as a wrong prediction.

Datasets	NextQA	STAR	Social-IQ-2.0	TrafficQA
Accuracy ***w/o error correction & self refinement***	47.8%	42.1%	45.5%	38.3%
Accuracy ***w/o error correction***	57.1%	44.9%	48.3%	41.1%
Accuracy ***w/o self refinement***	56.5%	43.1%	47.2%	40.2%
Accuracy ***with self refinement & error correction***	64.0%	47.2%	51.6%	43.5%

5.3 Self-Refinement

1. **In-Context Example Refinement:** In examining the self-refinement process, we employ the NeXt-QA dataset by randomly sampling 50 videos for the test set. Additionally, we curate a set of 20 in-context examples for evaluation. The accuracy of the test set is initially calculated. Subsequently, these in-context examples undergo self-refinement. This approach initially presents instructions to an LLM for program generation without contextual influence. The generated program, along with the original program, is then provided to the LLM with a prompt to maintain the structure of the initial program while enhancing it using the non-contextual program. The refined in-context examples undergo multiple iterations of this process, and we report the accuracy on the test set concerning the number of iterations of self-refinement applied to the in-context examples in Fig. 8a.

2. **Error Correction Evaluation:** To assess the efficacy of automatic error correction, we randomly select 400 videos from the STAR dataset. The evaluation involves calculating the number of errors stemming from program invalidity. Given the ability to feed a program into the auto-correction module multiple times, we explore the impact of increasing these iterations on error occurrences within the 400 video-instruction pairs and report our evaluations in Fig. 8b.

6 Conclusion

Our work expands the boundaries of Visual Programming by integrating it into the realm of video understanding, showcasing its efficacy in diverse applications such as Video Question Answering, Pose Estimation, and Multi-Video VQA. Additionally, we introduce a novel approach for enhancing the program generation process within the LLM through self-refinement, consequently elevating the efficacy of few-shot prompting to the LLM. Our approach not only broadens the scope of Visual Programming but also underscores the potential for continuous self-refinement to optimize the capabilities of LLMs for video reasoning tasks.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Jon Ander Campos and Jun Shern. Training language models with language feedback. In *ACL Workshop on Learning with Natural Language Supervision. 2022.*, 2022.

Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017.

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

Jingyuan Chen, Xinpeng Chen, Lin Ma, Zequn Jie, and Tat-Seng Chua. Temporally grounding natural sentence in video. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 162–171, 2018.

Peihao Chen, Chuang Gan, Guangyao Shen, Wenbing Huang, Runhao Zeng, and Mingkui Tan. Relation attention for temporal action localization. *IEEE Transactions on Multimedia*, 22(10): 2723–2733, 2019.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*, 2023.

Ken Dancyger. *The technique of film and video editing: history, theory, and practice*. Routledge, 2018.

Adrien Deliege, Anthony Cioppa, Silvio Giancola, Meisam J Seikavandi, Jacob V Dueholm, Kamal Nasrollahi, Bernard Ghanem, Thomas B Moeslund, and Marc Van Droogenbroeck. Soccernet-v2: A dataset and benchmarks for holistic understanding of broadcast soccer videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4508–4519, 2021.

Ali Erol, George Bebis, Mircea Nicolescu, Richard D Boyle, and Xander Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 108(1-2):52–73, 2007.

María Escobar, Cristina González, Felipe Torres, Laura Daza, Gustavo Triana, and Pablo Arbeláez. Hand pose estimation for pediatric bone age assessment. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part VI 22*, pages 531–539. Springer, 2019.

Owain Evans, Owen Cotton-Barratt, Lukas Finnveden, Adam Bales, Avital Balwit, Peter Wills, Luca Righetti, and William Saunders. Truthful ai: Developing and governing ai that does not lie. *arXiv preprint arXiv:2110.06674*, 2021.

Zhaopeng Feng, Yan Zhang, Hao Li, Wenqiang Liu, Jun Lang, Yang Feng, Jian Wu, and Zuozhu Liu. Improving llm-based machine translation with systematic self-correction. *arXiv preprint arXiv:2402.16379*, 2024.

Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Predicting the future: A jointly learnt model for action anticipation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5562–5571, 2019.

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE international conference on computer vision*, pages 5267–5275, 2017.

Rohit Girdhar and Kristen Grauman. Anticipative video transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13505–13515, 2021.

Olga Golovneva, Moya Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. Roscoe: A suite of metrics for scoring step-by-step reasoning. *arXiv preprint arXiv:2212.07919*, 2022.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*, 2023.

Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6047–6056, 2018.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14953–14962, 2023.

Qingqiu Huang, Yu Xiong, Anyi Rao, Jiaze Wang, and Dahua Lin. Movienet: A holistic dataset for movie understanding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 709–727. Springer, 2020.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. *arXiv preprint arXiv:2303.17491*, 2023.

Irena Koprinska and Sergio Carrato. Temporal video segmentation: A survey. *Signal processing: Image communication*, 16(5):477–500, 2001.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. *Advances in Neural Information Processing Systems*, 35:21314–21328, 2022.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L Berg. Tvqa: Localized, compositional video question answering. *arXiv preprint arXiv:1809.01696*, 2018.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023.

Wei Li, Can Gao, Guocheng Niu, Xinyan Xiao, Hao Liu, Jiachen Liu, Hua Wu, and Haifeng Wang. Unimo: Towards unified-modal understanding and generation via cross-modal contrastive learning. *arXiv preprint arXiv:2012.15409*, 2020.

Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for classification and detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4804–4814, 2022.

Kevin Qinghong Lin, Pengchuan Zhang, Joya Chen, Shraman Pramanick, Difei Gao, Alex Jinpeng Wang, Rui Yan, and Mike Zheng Shou. Univtg: Towards unified video-language temporal grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2794–2804, 2023a.

Xudong Lin, Simran Tiwari, Shiyuan Huang, Manling Li, Mike Zheng Shou, Heng Ji, and Shih-Fu Chang. Towards fast adaptation of pretrained contrastive models for multi-channel video-language retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14846–14855, 2023b.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. URL `https://arxiv.org/abs/2304.08485`.

Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3202–3211, 2022.

Jianqiao Lu, Wanjun Zhong, Wenyong Huang, Yufei Wang, Fei Mi, Baojun Wang, Weichao Wang, Lifeng Shang, and Qun Liu. Self: Language-driven self-evolution for large language model. *arXiv preprint arXiv:2310.00533*, 2023.

Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024)*, 2024.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024.

Arun V Reddy, Ketul Shah, William Paul, Rohita Mocharla, Judy Hoffman, Kapil D Katyal, Dinesh Manocha, Celso M de Melo, and Rama Chellappa. Synthetic-to-real domain adaptation for action recognition: A dataset and baseline performances. *arXiv preprint arXiv:2303.10280*, 2023.

Jingqing Ruan, Yihong Chen, Bin Zhang, Zhiwei Xu, Tianpeng Bao, Guoqing Du, Shiwei Shi, Hangyu Mao, Xingyu Zeng, and Rui Zhao. Tptu: Task planning and tool usage of large language model-based ai agents. *arXiv preprint arXiv:2308.03427*, 2023.

Arka Sadhu, Tanmay Gupta, Mark Yatskar, Ram Nevatia, and Aniruddha Kembhavi. Visual semantic role labeling for video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5589–5600, 2021.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*, 2022.

Arindam Sengupta, Feng Jin, Renyuan Zhang, and Siyang Cao. mm-pose: Real-time human skeletal posture estimation using mmwave radars and cnns. *IEEE Sensors Journal*, 20(17):10032–10044, 2020.

Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 510–526. Springer, 2016.

Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Xun Guo, Tian Ye, Yan Lu, Jenq-Neng Hwang, et al. Moviechat: From dense token to sparse memory for long video understanding. *arXiv preprint arXiv:2307.16449*, 2023.

Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. Head-to-tail: How knowledgeable are large language models (llm)? aka will llms replace knowledge graphs? *arXiv preprint arXiv:2308.10168*, 2023.

Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning, 2023.

Omkar Thawakar, Ashmal Vayani, Salman Khan, Hisham Cholakal, Rao M Anwer, Michael Felsberg, Tim Baldwin, Eric P Xing, and Fahad Shahbaz Khan. Mobillama: Towards accurate and lightweight fully transparent gpt. *arXiv preprint arXiv:2402.16840*, 2024.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.

Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.

Liang Wang, Nan Yang, and Furu Wei. Learning to retrieve in-context examples for large language models. *arXiv preprint arXiv:2307.07164*, 2023.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

Yi Wang, Kunchang Li, Yizhuo Li, Yinan He, Bingkun Huang, Zhiyu Zhao, Hongjie Zhang, Jilan Xu, Yi Liu, Zun Wang, Sen Xing, Guo Chen, Junting Pan, Jiashuo Yu, Yali Wang, Limin Wang, and Yu Qiao. Internvideo: General video foundation models via generative and discriminative learning, 2022.

Tongshuang Wu, Haiyi Zhu, Maya Albayrak, Alexis Axon, Amanda Bertsch, Wenxing Deng, Ziqi Ding, Bill Guo, Sireesh Gururaja, Tzu-Sheng Kuo, et al. Llms as workers in human-computational algorithms? replicating crowdsourcing pipelines with llms. *arXiv preprint arXiv:2307.10168*, 2023.

Li Xu, He Huang, and Jun Liu. Sutd-trafficqa: A question answering benchmark and an efficient network for video reasoning over traffic events, 2021.

Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Tubedetr: Spatio-temporal video grounding with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16442–16453, 2022.

Hui Yang, Lekha Chaisorn, Yunlong Zhao, Shi-Yong Neo, and Tat-Seng Chua. Videoqa: question answering on news video. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 632–641, 2003.

Jianing Yang, Xuweiyi Chen, Shengyi Qian, Nikhil Madaan, Madhavan Iyengar, David F Fouhey, and Joyce Chai. Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent. *arXiv preprint arXiv:2309.12311*, 2023.

Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13–es, 2006.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. Self-chained image-language model for video localization and question answering. *Advances in Neural Information Processing Systems*, 36, 2024.

Tao Yu, Rui Zhang, He Yang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, et al. Cosql: A conversational text-to-sql challenge towards cross-domain natural language interfaces to databases. *arXiv preprint arXiv:1909.05378*, 2019.

Runhao Zeng, Haoming Xu, Wenbing Huang, Peihao Chen, Mingkui Tan, and Chuang Gan. Dense regression network for video grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10287–10296, 2020.

Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. Temporal sentence grounding in videos: A survey and future directions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

Zhu Zhang, Zhou Zhao, Yang Zhao, Qi Wang, Huasheng Liu, and Lianli Gao. Where does it exist: Spatio-temporal video grounding for multi-form sentences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10668–10677, 2020.

Ce Zheng, Wenhan Wu, Chen Chen, Taojiannan Yang, Sijie Zhu, Ju Shen, Nasser Kehtarnavaz, and Mubarak Shah. Deep learning-based human pose estimation: A survey. *ACM Computing Surveys*, 56(1):1–37, 2023.

Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Paco Guzman, Luke Zettlemoyer, and Marjan Ghazvininejad. Detecting hallucinated content in conditional neural sequence generation. *arXiv preprint arXiv:2011.02593*, 2020.