Tasks Reflected in the Eyes: Egocentric Gaze-Aware Visual Task Type Recognition in Virtual Reality

Zhimin Wang, and Feng Lu, *Senior Member, IEEE*

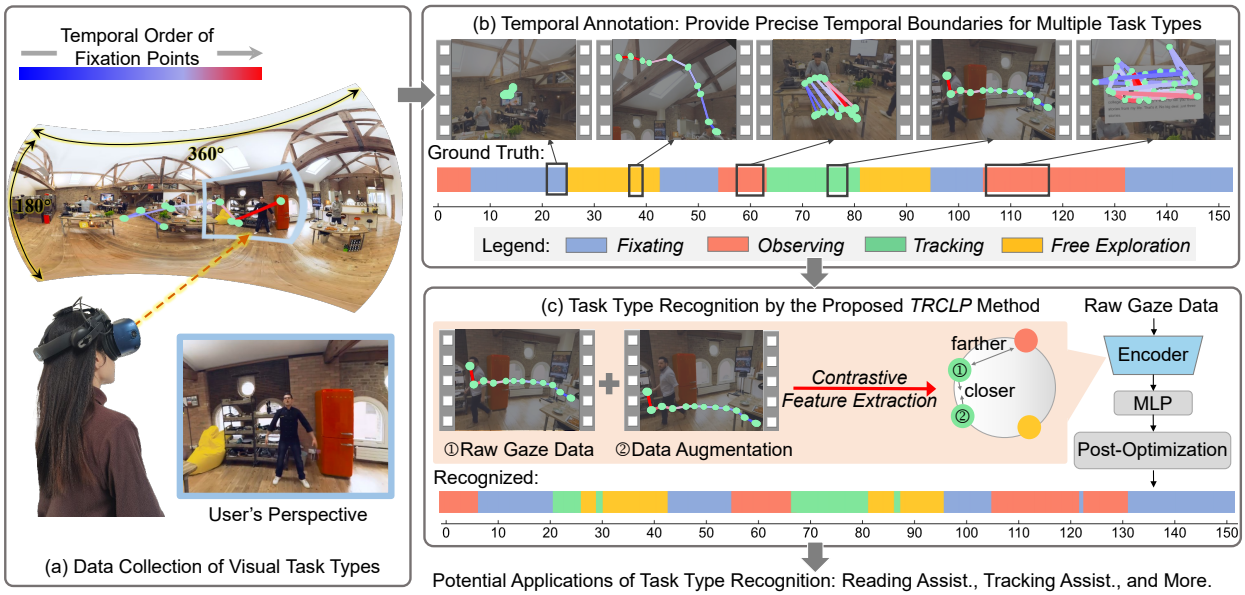


Fig. 1: We aim to recognize user's visual task types in virtual reality. To achieve this, we provide four scene-agnostic task types and a novel dataset including temporal annotations. Using this dataset, we also propose the Task type Recognition method via Contrastive Learning and Post-optimization (TRCLP) to recognize user task types.

Abstract—With eye tracking finding widespread utility in augmented reality and virtual reality headsets, eye gaze has the potential to recognize users' visual tasks and adaptively adjust virtual content displays, thereby enhancing the intelligence of these headsets. However, current studies on visual task recognition often focus on scene-specific tasks, like copying tasks for office environments, which lack applicability to new scenarios, *e.g.*, museums. In this paper, we propose four scene-agnostic task types for facilitating task type recognition across a broader range of scenarios. We present a new dataset that includes eye and head movement data recorded from 20 participants while they engaged in four task types across 15 360-degree VR videos. Using this dataset, we propose an egocentric gaze-aware task type recognition method, TRCLP, which achieves promising results. Additionally, we illustrate the practical applications of task type recognition with three examples. Our work offers valuable insights for content developers in designing task-aware intelligent applications. Our dataset and source code will be released upon acceptance.

Index Terms—Virtual reality, eye tracking, visual task type recognition, depth learning, intelligent application

1 Introduction

Virtual Reality (VR), Augmented Reality (AR), and Mixed Reality (MR), collectively referred to as Extended Reality (XR), have found numerous applications in various fields such as healthcare, education, and entertainment in recent years [4, 24, 40, 53]. With the widespread adoption of MR and AR headsets, such as Apple's Vision Pro and Microsoft HoloLens, an increasing number of researchers from both industry and academia are engaging in XR research.

In terms of interaction design, many new studies aim to make XR systems more intelligent [21, 38, 52]. A prospective way is to predict user's visual intents and dynamically adapting the virtual content within XR environments [43]. For example, when a user is recognized as viewing a painting, the XR system could play coordinating music to create an immersive atmosphere. Alternatively, when a user is reading an electronic book, the XR system could automatically turn the pages [33]. Predicting user's intents is also known as visual task recognition, which has numerous applications in the XR field, *e.g.*, low-friction user interfaces [11, 29, 37], adaptive virtual content design [12, 15], and attention-driven intelligent systems [25, 52]. Consequently, XR systems have the potential to reduce the interaction burden on users by recognizing their tasks and interaction goals, thereby facilitating the completion of corresponding tasks with less friction [28, 56].

An effective approach for visual task recognition is to leverage eye gaze data. Yarbus's seminal work analyzed eye positions across seven different visual tasks and found significant differences in eye movement patterns [55]. Eye movements have also been proven valuable for revealing user behavior and cognition. Numerous studies have interpreted the relationship between eye movements and visual attention [7, 13, 23],

- *Zhimin Wang and Feng Lu are with the State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University, Beijing 100191, China. e-mail: {zm.wang \ lufeng}@buaa.edu.cn.*
- *Feng Lu is the corresponding author.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

cognitive states [16, 18], and memory [3]. Inspired by these findings, researchers have attempted the "inverse Yarbus process", *i.e.*, identifying user tasks from eye movement patterns [8, 38, 41]. For instance, Bulling *et al.* recognized tasks such as copying, reading, writing and web browsing, *etc*, in an office scenario based on eye movement features like fixations and saccades [9]. Several studies have explored task recognition in specific scenarios, such as museum tours [6, 33], robot repair [5], and desktop software usage [50]. These works have achieved promising task recognition results within their respective domains.

However, these studies on task recognition face two challenges when applied in practical scenarios. 1) These studies typically define scene-specific tasks like copying tasks for office environments [9, 43] or memory task for corridors [38]. When the scene changes, these tasks need to be redesigned for the new context, restricting their utility across diverse scenarios. 2) For ease of data collection, participants are often asked to perform a singular task during each session. In contrast, real-world situations involve frequent switching between multiple tasks. For instance, as depicted in Fig. 1 (b), a user might initially focus on a television screen, then explore their surroundings, and subsequently count items on a table, *etc*. Existing datasets, therefore, fall short in accommodating the free switching between multiple tasks. These challenges highlight the limitations of traditional task recognition research and emphasize the need for more versatile and pragmatic solutions.

To address the first challenge, we propose four scene-agnostic task types derived from an analysis of eye movement patterns and object states, facilitating task type recognition across a broader range of scenarios. Regarding the second challenge, we design a temporal annotation method for data collection, which provides precise temporal boundaries for multiple task types in every video clip. This collected dataset enables training the recognition model to support free task type switching. Furthermore, we conduct a collection process to capture eye and head movement data from 20 participants while performing these task types. Based on this dataset, we propose an egocentric gaze-aware Task type Recognition method through Contrastive Learning and Post-optimization (TRCLP, pronounced as "try-clip"), which learns the mapping between eye movement data and task types, enabling task type classification. We also evaluate its performance through extensive experiments. The results demonstrate that our TRCLP outperforms state-of-the-art methods in terms of recognition accuracy and the required length of time window. Finally, we demonstrate the applications of task type recognition with three examples. Our work has the potential to enhance the intelligence of XR systems in diverse contexts.

Overall, our paper makes the following contributions:

- We propose four scene-agnostic visual task types for VR systems, enabling task type recognition in a broader range of scenarios.
- We provide a new dataset for task type recognition that provides precise temporal boundaries for multiple task types in every video clip, using which we can train the recognition method to support free task switching.
- We present TRCLP, a novel learning-based approach for recognizing task types, which outperforms the state-of-the-art methods. Additionally, we also demonstrate the utility of task type recognition through three examples.

2 Related Works

In this section, we review basic eye movement Types and visual task definitions, as well as a discussion of eye-tracking-based methods for task recognition.

2.1 Eye Movement Types

The main types of eye movements include fixations, saccades, and smooth pursuit, blinks [35]. Fixation involves holding a stationary object in the foveal region for visual information acquisition [19]. The duration of a fixation varies between 50∼600 ms and typically includes small eye movements such as tremors and drifts to aid in aligning the eye with the target [39]. Saccades are rapid eye movements between fixation points to bring the visual scene onto the fovea [19]. The duration of each saccade depends on the specific task, with an average duration of 20∼40 ms, and the amplitude of the saccade also depends on the task. Smooth pursuit is a tracking eye movement used to keep a moving object on the fovea. It can only be executed when a moving object is present and the eye movement speed is generally less than 30°/s [34]. Blinking refers to the opening and closing of the eyelids to keep the eyes comfortable, with a frequency of 4∼6 seconds [1].

2.2 Visual Task Definitions

Yarbus *et al.*'s seminal work discovered that the eye movement trajectories differ when performing seven different visual tasks while observing a painting [55]. For instance, the eye movement trajectory differs significantly between observing an image with and without instructions, indicating that a user's task can be inferred from their eye movements. Since then, many researchers have linked tasks with eye movements by different task settings [10, 27, 31, 47, 50]. For example, Bulling *et al.* designed six tasks for an office setting, including copy, read, write, video, browse, and null [9]. Bektas *et al.* designed reading, inspecting, and searching tasks for a robot repair scene in an AR setting [5]. Hild *et al.* required users to explore, observe, search, and track while watching street walking videos [20]. Lan *et al.* set reading, conversation, and watching tasks for a museum scene [33]. Hu *et al.* designed free viewing, search, saliency, and track tasks for a more general scene [22].

In summary, these works mainly explored visual tasks in specific scenes. In this research, we design four scene-agnostic task types, enabling the recognition across a broader range of scenarios. A detailed comparison of between prior works and our research is shown in Tab. 1. Furthermore, in the previous works, only one task was collected per session, and continuous task type switching was not supported. In this research, we design a method that provides precise temporal boundaries between task types in every video clip, which can be used to train the recognition method to support free task type switching.

2.3 Task Recognition Methods

Researchers have proposed many eye-tracking-based task recognition methods [5, 11, 41, 48]. Commonly used methods can be divided into two categories. One is to define a rich set of eye-tracking metrics, conduct maximum correlation analysis, select the most relevant metrics, and use machine learning methods for learning and recognition. For example, Bulling *et al.* defined 90 eye-tracking metrics based on behaviors such as saccades, fixations, and blinks, and used support vector machines (SVM) for task recognition [9]. Srivastava *et al.* designed 50 eye-tracking metrics, including low-level and mid-level features, and used SVM and other machine learning methods for recognition [50]. The disadvantage of this method is that it requires a lot of effort to design hand-crafted features, and the relevant eye-tracking metrics also differ when the visual tasks are different.

In recent years, deep learning methods have shown strong generalization ability, which simplifies the requirements for eye-tracking metrics. For example, Ishlmaru *et al.* used three metrics, including blink speed, eye movement coordinates, and head acceleration, and used convolutional neural network (CNN) and Long short-term memory (LSTM) methods to predict user tasks [27]. Hu *et al.* used three metrics, including gaze position on the screen, head orientation, and gaze direction in the world, and used CNN and bidirectional gated recurrent unit (BiGRU) for prediction [22]. In contrast with prior works, in this research, we use contrastive learning to improve the generalization of the method and use post-optimization to make the results smoother.

3 Data Collection

First, we analyze eye movement patterns and summarize four visual task types, as introduced in Section 3.1. Then, we introduce the visual stimuli, including our 360° VR video datasets and text images, in Section 3.2. Next, we described the temporal annotation process for the data in Section 3.3, followed by a detailed explanation of the system implementation in Section 3.4. We also provide information on the participants and data collection process in Section 3.5. Finally, we presented an analysis of the dataset at the end.

Table 1: A comparison between selected prior works and our research on task recognition. Our task types support recognition across a broader range of scenarios. Moreover, our dataset incorporates temporal annotations, facilitating the recognition of switching between multiple task types.

Research	Task Definitions	Stimuli Number	Users	Temporal Annotation	Free Task Switching
Bulling et al. [9]	Office: Copy ○ Read ○ Write ○Video ○ Browse ○ Null	Real World	8	No	No
Borji et al. [8]	Paintings: Yarbus's original 7 tasks for paintings	15 images	21	No	No
Kiefel et al. [31]	Maps: Explore ○ Search ○ Plan ○ Follow ○ Comparison	1 image	17	No	No
Srivastava et al. [50]	Desktop software: Desktop software tasks	Real World	24	No	No
Hild et al. [20]	Street walking: Explore ○ Observe ○ Search ○ Track	1 video	30	No	No
Bektas et al. [5]	Robot repair: Read ○ Inspect ○ Search	Real World	10	No	No
Lan et al. [33]	Museum: Read ○ Communicate ○ Browse ○ Watch	203 images, 13 videos	8	No	No
Malpica et al. [38]	Indoor corridor: Free exploration ○ Memory ○ Visual Search	3 VR scenes	37	No	No
Hu et al. [22]	**Task Types**: Free viewing ○ Search ○ Saliency ○ Track	15 VR videos	30	No	No
Ours	**Task Types**: Fixate ○ Observe ○ Track ○ Free exploration	10 images, 15 VR videos	20	**Yes**[1]	**Yes**[1]

[1] Our temporal annotation provides precise temporal boundaries for multiple task types in every video clip, using which we can train the recognition method to support seamless task type switching.

(a) Track a bicycle rider (b) Track a walker (c) Track a shark

Fig. 2: Visual tasks and associated eye movement patterns across three scenarios.

3.1 Design of Visual Task Types

Our goal is to design scene-agnostic task types for VR systems. Prior research first analyzed the specific scenes, and then defined relevant visual tasks [5, 9, 50]. Instead, we focus on analyzing eye movement characteristics to categorize task types. We begin by clarify the meanings of eye movement type, visual task and visual task type.

- **Eye Movement Type.** The types of eye movements refer to the movement patterns of gaze points, include fixations, saccades and smooth pursuit, *etc*. The characteristics and distinctions of these eye movement types were discussed in Section 2.1.
- **Visual Task.** This concept pertains to specific tasks that users are performing, typically connected to particular objects or scenes. For example, a user visually tracks a bicycle rider or tracks a shark, as shown in Fig. 2.
- **Visual Task Type.** Visual tasks are classified into distinct types based on object states and eye movement types. For example, tasks such as tracking a bicycle rider, a walker, or a shark fall under the same task type, which involves tracking moving objects.

Eye movements facilitate the clear observation of objects in different motion states by aligning the image of an object of interest with the central fovea of the retina. Therefore, this process of observing various states of objects corresponds to perform different types of visual tasks. Common eye movement types include fixation, smooth pursuit and saccade [35]. *Fixation* typically aims at fixating on a stationary object for a certain period. Hence, **Fixating on a Stationary object** (*FS*) is identified as one of our task types, which is rarely explored by prior studies. *Smooth pursuit* involves following a moving target. Many studies set the specific targets to track [20, 31]. We categorize these tracking tasks under the task type **Tracking a Moving object** (*TM*).

Saccade is generally made to observe multiple objects. During this process, users usually engage in a series of saccades interrupted by short fixations to examine the objects. We categorize the saccades involved in observing multiple objects into sequential and irregular types. Sequential saccades, often aim at observing objects arranged in order, such as reading text or counting objects in a sequence [17, 44, 46, 50], leading us to identify **Observing Sequential objects** (*OS*) as another task type. In contrast, irregular saccades are generally utilized to observe randomly arranged objects. Common visual task types in this category include object searching and free exploration [5, 22, 38]. However, object searching is often a transient precursor to other task types with a shorter duration, making it challenging to record. Therefore, we do not consider it as a separate visual task type in this study but include it within other task types, *e.g.*, searching for a target to fixate on. In this work, **Free Exploration** (*FE*) serves as the representative task type for irregular saccades.

Table 2: Four visual task types proposed in this study.

Visual Task Type	State of Object	Eye Movement Type
Fixating on a Stationary object (FS)	Stationary object	Long fixations
Tracking a Moving object (TM)	Moving object	Smooth pursuit
Observing Sequential objects (OS)	Sequentially stationary objects	Sequential saccades with short fixations
Free Exploration (FE)	Unordered stationary or moving objects	Irregular saccades with short fixations

The relationship among these four task types, object states, and eye movement types is summarized in Table 2. Additionally, examples of eye movements for these four types are illustrated in Fig. 3.

3.2 Visual Stimuli

For these task types, we employ VR 360° videos as visual stimuli for data collection. We establish specific criteria for video selection: 1) To ensure that the movement of targets does not introduce any ambiguity in the task instruction, the panoramic camera remains stationary. 2) Each video includes stationary objects, moving objects, and multiple objects of the same type, serving as targets for specific visual tasks. 3) Our videos offer a diverse content and styles, including indoor and outdoor scenes, sports, shows, and exhibitions, *etc*.

Based on these requirements, we select three videos from the EHTask and twelve videos from YouTube, as shown in Fig. 4. Notably, while our visual stimuli are 360° VR videos, these videos are captured in real-world scenarios, such as basketball games, street scenes, and stations. Consequently, we believe these videos provide visual stimuli consistent with real-world settings, enabling our visual task type recognition method to be effectively applied in practical scenarios. Each 360° video is saved using the equirectangular map format. The videos have a resolution of 3840×(1920∼2160) pixels and a frame rate of 30Hz. We crop each video to 150 seconds, as done in EHTask. Furthermore,

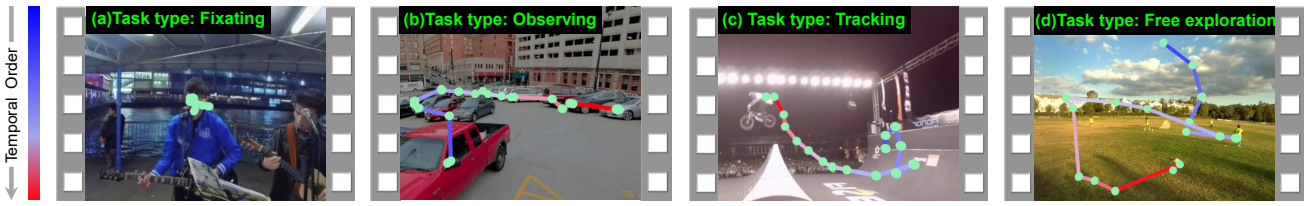


Fig. 3: The eye movement patterns related to four task types. (a) Fixating on a stationary object, (b) Observing sequential objects, (c) Tracking a moving object, (d) Free exploration.

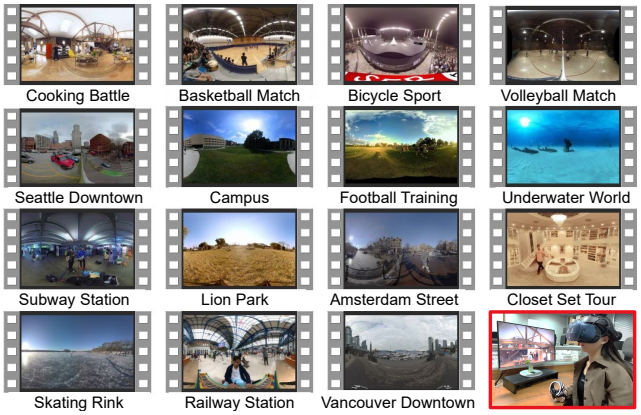


Fig. 4: The experimental setup (bottom-right) and the 15 real-scene 360° videos used in our experiments.

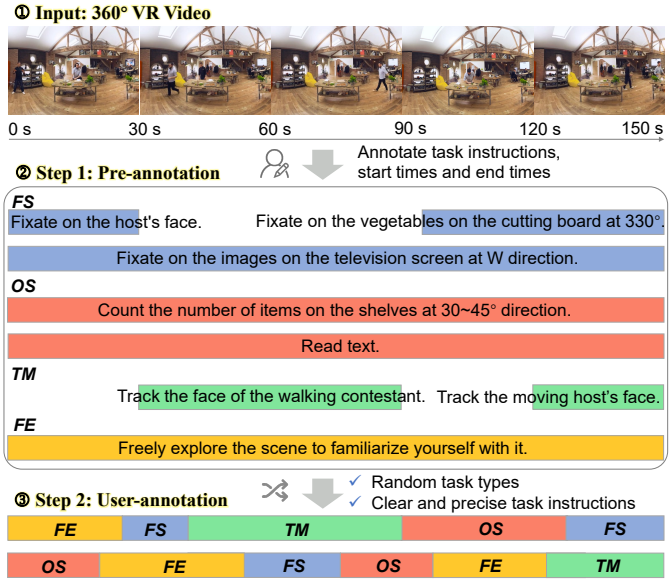


Fig. 5: Temporal annotation of task type datasets. We pre-annotate each video by labeling four task types with specific task instructions and time intervals. During user-annotation, users are presented with random tasks and are responsible for determining the start and end times of each task.

these videos are projected onto the inner surface of a spherical skybox, allowing users to view them from the inside of the sphere in VR.

We employ visual tasks such as sequential counting and reading text to represent the *OS*. Therefore, we include ten text images for reading, which are extracted from two books and a speech text: "Rich Dad Poor Dad", "The Craft of Research" and "Steve Jobs' Speech". Each text varies in line widths, column heights and text backgrounds.

3.3 Data Temporal Annotation

Based on our abundant visual stimuli above, we propose an annotation method for collecting data that correlates gaze patterns with task types. Traditional approaches to annotate datasets typically involve post-hoc annotation, where experts manually label the data after its collection [26, 49]. However, this method faces challenge when individuals frequently switch between multiple tasks. Since visual tasks are highly subjective and user-dependent, it is difficult to determine the specific task a user is engaged in and the exact moments of task switching. To address this limitation, we introduce a temporal annotation method comprising two stages: pre-annotation and user-annotation.

Pre-annotation. During this stage, annotation experts label multiple annotations for each video. Each annotation consists of a specific task instruction, and a time interval during which the tasks appeared continuously, represented as $[L_i, R_i]$ where $0 \leq L_i \leq R_i \leq 150s$, $i = 1...N$. Here, N denotes the number of annotations of one video. Fig. 5 illustrates an example of pre-annotation for one video.

In this process, the annotation expert follows these steps: 1) Watch each 360° VR video 2∼3 times to be familiar with the content. 2) Identify objects in the video that correspond to each visual task type (e.g., *FS*). For example, the expert may consider the stationary "host's face" for the *FS*. 3) Play the video and identify time intervals when the "host's face" remains static. This yields a specific task instruction (*e.g.*, "fixate on the host's face at 45° direction") and a corresponding time interval (*e.g.*, [0, 27] s). 4) Except for the *FE*, repeat steps 2∼3 multiple times for each task type to increase the number of annotations. 5) Move on to the next task type and repeat the process.

It is possible for the time intervals of all annotations within a given task to overlap, as shown by *FS* in Fig. 5. Additionally, the concatenation of all annotations for each task type may not cover the entire video segment, as indicated by *TM* in Fig. 5. This occurs because there are moments in the video without any moving objects. On average, each video contains $\hat{N} = 13.9$ annotations, which include 4.5 (*FS*), 4.6 (*OS*), 3.8 (*TM*), and 1 (*FE*). Notably, *FS* and *TM* annotations in some videos refer to a group of objects, allowing the user to select one object to fixate on or track. Therefore, the number of stimuli in each video exceeds the number of annotations.

User-annotation. The user-annotation stage (*i.e.*, data collection) proceeds as follows. 1) The system randomly assigns task instructions to users, who can pause the system using the joystick to read the instructions (the first instructions pause automatically). Users then locate targets according to the instructions. Data is not recorded during system pauses. 2) Before resuming, users must align their gaze and head directions with those of the first frame from the pause state, displayed on the VR HMD. The system then checks if the current are within 5° of the first frame, prompting adjustments if needed. This ensures the spatial continuity of recorded head and gaze data, enabling seamless switching between task types. Once aligned, users start performing tasks using the joystick, and the instruction disappears. 3) Users determine when to terminate a task, or the system automatically ends the task if it exceeds the time interval limit for that task. Users then proceed to the next randomly assigned task. 4) For each video, users repeat steps 1∼3 to complete the entire data collection process.

This approach ensures that all collected task annotations have well-defined temporal boundaries. It is worth noting that these task instructions, such as "fixating on the host's face" or "fixating on the vegetable" in Fig. 5, are provided to assist users in understanding and performing visual tasks. However, in our data annotation, all fixation instructions across different scenes are considered as the same task type, *i.e.*, *FS*. Therefore, our task types are scene-agnostic.

Table 3: Statistics of four task types in the dataset.

Task Type Name	FS	OS	TM	FE
Task Type Number	498	499	373	392
Task Type Duration (Total, min)	163.5	173.0	132.4	130.1
Duration Proportion (%)	27.3%	28.9%	22.1%	21.7%

Table 4: Statistics of task type switching in the dataset.

Switch	Count	Switch	Count	Switch	Count
FS→OS	173	FS→TM	140	FS→FE	121
OS→FS	196	OS→TM	131	OS→FE	108
TM→FS	117	TM→OS	109	TM→FE	90
FE→FS	130	FE→OS	142	FE→TM	65

3.4 System Implementation

By combining the aforementioned task types, visual stimuli, and temporal data annotations, we describe the implementation details of the data collection system as follows.

Random Process for Selecting Task Types. The random process for selecting specific tasks is illustrated in Supplementary Material. We found one-third of the time in the video datasets has no moving objects. Our objective is to ensure that four task types have the equal probability of appearing in the final dataset. To achieve this, we have assigned probabilities to each task based on the proportion of time taking up in pre-annotation, *i.e.*, *FS* (22%), *OS* (22%), *TM* (34%) and *FE* (22%). For *OS*, in real-world scenarios, we argue that there are more types of sequential counting with different representations. Therefore, we assign a 75% probability to counting and a 25% probability to reading. To prevent user sickness from large numbers of head rotations, we set that in one video session, the *FE* does not occur continuously, and the total number of *FE* does not exceed two.

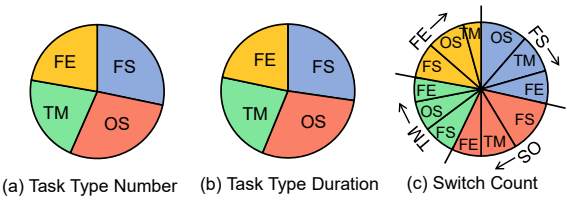
Data Recording. The eye feature data we record follows the same format as EHTask [22]. This includes eye-in-head data (EiH (e_x, e_y), where $e_x, e_y \in [0,1]$), head orientation data (Head (h_x, h_y), where $h_x \in [-180°, 180°]$ and $h_y \in [-90°, 90°]$), and gaze-in-world data (GiW (g_x, g_y), where $g_x \in [-180°, 180°]$ and $g_y \in [-90°, 90°]$). Besides eye feature data, we also record the task instruction, start time and end time of each task. Data is not recorded during the pause stages of the system. Before resuming the system, users are prompted to actively rotate their gaze and head directions with their directions before the system paused, ensuring the continuity of recorded data. Each task has a duration of 5∼20 seconds, and during the training phase, we inform the user that they do not need to keep track of time themselves. Instead, they only need to ensure the duration falls approximately within this time range.

Apparatus. We conduct data collection using a computer equipped with an Intel Core i5-8500 CPU running at 3.00Ghz and an NVIDIA GeForce RTX 2080 SUPER GPU. The HTC Vive Cosmos headset, together with the 7invensun Droolon F1 eye tracker, which provides an accuracy of 0.5°, is employed for presenting 360° VR videos. We utilize Unity3D to render the VR videos, and recorded EiH, Head, and GiW data. The experimental setup is shown in the bottom right of Fig. 4. To assist users in understanding the task instructions that describe the positions of stimuli, we employ a navigation compass that indicates direction, as shown in Supplementary Material. The user is positioned at the center of the compass, with the 360° directions marked at 15-degree intervals, allowing for easy identification of the target position. The compass is placed at the bottom of the field of view.

3.5 Participants and Procedure

We recruited a total of 20 participants from the campus (12 males and 8 females). The average age of the participants was 23 years (std = 2.4), and all had normal or corrected-to-normal vision. The VR headset allowed individuals to wear glasses while using it. Additionally, all participants were proficient in reading English text.

Each participant began the study by completing a pre-study questionnaire. Following this, they watched an introductory video that explained the visual task types they would be completing. The video also included demonstrations of a user watching a 360° VR video and performing the visual tasks under task instructions. Next, participants underwent a gaze calibration procedure. They then proceeded to the training phase, where they randomly executed 1∼2 videos to familiarize themselves with the overall operation, before moving on to the experimental phase. If the gaze estimation was found to be inaccurate before playing each video, the users could perform gaze calibration again. After watching 3 videos, the users were instructed to take a one-minute break.



(a) Task Type Number (b) Task Type Duration (c) Switch Count

Fig. 6: Statistics of Task Type Datasets. (a) Comparison of the frequency of each type in the datasets. (b) Comparison of the duration of each type. (c) Comparison of the number of switches between different task types.

In our experiment, each participant was instructed to watch 12 videos, randomly selected from the 15 videos. Each video was played once and presented in a random order. It resulted in a total of 240 recordings (20 participants × 12 videos). Each recording contained EiH data (25Hz), Head data (25Hz), and GiW data (25Hz). Each of the 15 videos was viewed by 16 participants.

3.6 Data Analysis

The analysis of the collected data is presented in Tab. 3 and 4, from which we make the following observations. 1) The total duration of the dataset is 10 hours, with each task type ranging from 130 minutes (21.7%) to 173 minutes (28.9%), and the distribution of task type durations is relatively balanced, as shown in Fig. 6 (b). 2) As discussed in Section 3.5, we limit the frequency of *FE* occurrences due to the sickness caused by excessive head rotations, resulting in the shortest total duration. 3) As described in Section 3.4, one-third of the time in the video datasets has no moving objects. To achieve a relatively balanced duration proportion, we increase the random probability of *TM*, as shown in Fig. 6 (a). 4) The total number of task switching is 1522, with an average of 6.34 switches per video. 5) The average number of switches between different task types is 126.83 (std = 33.3). Except for *TM*→*FE* (90) and *FE*→*TM* (65), the counts for switching types are over 100, and the switching between different task types was relatively balanced, as shown in Fig. 6 (c).

4 Recognition Model

Based on the extensively collected datasets mentioned above, we propose the TRCLP to recognize user's task types. An overview of the TRCLP is described in Fig. 7. Firstly, time-series data augmentation is applied to the input data, as described in Section 4.1. Subsequently, the different views of the data are inputted into the temporal contrasting module to minimize the impact of data augmentation. This is achieved by optimizing the contrastive loss, as explained in Section 4.2. The trained encoder is then incorporated into the task type recognition framework, as outlined in Section 4.3. Finally, a filtering method that has been specifically designed is utilized to process the output of network, thereby producing smoother results and improved recognition accuracy, as discussed in Section 4.4.

4.1 Time-Series Data Augmentation

Contrastive methods aim to maximize the similarity between different views of the same instance while minimizing their similarity to other instances. Thus, appropriate data augmentation techniques are crucial for contrastive learning. Common augmentations for time-series data include jitter, magnitude-warping, time-warping, scaling, and flipping [51]. After analyzing eye movement trajectories and conducting tests, we found that scaling and flipping provide the most significant enhancements. Data from an eye tracker is typically filtered

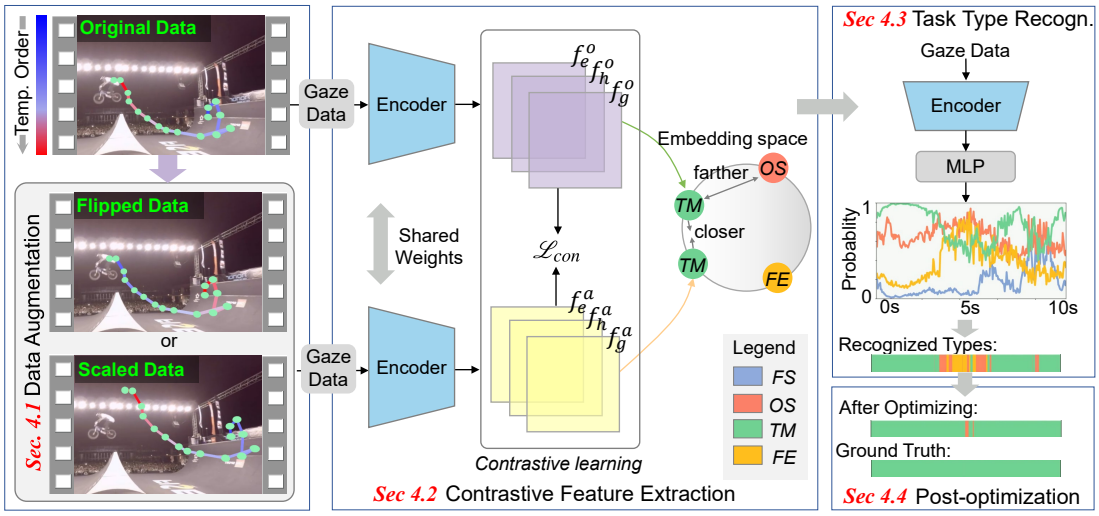


Fig. 7: Overview of the proposed TRCLP. Our method augments input gaze data through scaling or flipping. The augmented data is then processed by the contrastive feature extraction module to enhance the encoder's generalization. This encoder, combined with an MLP module, recognizes task types. The recognized task types are then refined by a post-optimization module for smoother results and improved accuracy. Note that the scaled eye gaze data does not match the video background as only the gaze data is augmented.

and smooth, showing no signs of jitter. Introducing magnitude-warping or time-warping can cause inconsistent changes within each temporal segment, leading to qualitative variations in eye movement patterns and decreased accuracy.

In contrast, scale augmentation ensures consistent magnitude changes across temporal segments. For instance, when tracking moving objects, scaling affects only the height or width of the eye movement trajectory, enlarging or shrinking the motion area without altering the patterns, as shown in Fig. 7. Flip augmentation reverses the temporal order of the eye movement data without changing the motion trajectory, only its direction. For each input sample without task switching, we set a 50% probability for scaling augmentation and a 50% probability for flipping augmentation. During task switching, flip augmentation is not applied to avoid changing the label in the last frame.

4.2 Contrastive Feature Extraction

Supervised contrasting learning leverages contrastive loss to prompt the encoder in producing more similar feature representations for entries belonging to the same visual task type, leading to enhanced clustering in the feature space. Assuming that each batch consists of B samples, denoted as $\{X_i, Y_i\}_{i=1\cdots B}$, where X_i comprises EiH, Head, and GiW data represented as (x_i^e, x_i^h, x_i^g), and Y_i is the task type label of the last frame of each time window. After applying data augmentation, each batch in the network training contains $2B$ samples, denoted as $\{\tilde{X}_j, \tilde{Y}_j\}_{j=1\cdots 2B}$. Here, $\tilde{X}_{2i}$ and $\tilde{X}_{2i-1}$ correspond to the augmented and non-augmented views of X_i, respectively. Additionally, $\tilde{Y}_{2i} = \tilde{Y}_{2i-1} = Y_i$. The Encoder network maps the data to a set of feature vectors, $F_j = \text{Enc}\left(\tilde{X}_j\right)$, where $F_j = \left(f_j^e, f_j^h, f_j^g\right)$. Let $I \equiv \{1\cdots 2B\}$ represent all the index values in the batch. We define $K(j) \equiv \left\{k \mid k \in I \backslash \{j\}, \tilde{Y}_k = \tilde{Y}_j\right\}$ as the set of indices of views that have the same task type label as sample j in $2B$ samples except j. To compute the supervised contrasting loss $\mathscr{L}_{con}$, we use f_j^e as an example, which is formulated as follows:

$$\mathscr{L}_{con} = \sum_{j\in I} \frac{-1}{|K(j)|} \sum_{k\in K(j)} \log \frac{\exp\left(f_j^e \cdot f_k^e / \tau\right)}{\sum_{a\in I\backslash\{j\}} \exp\left(f_j^e \cdot f_a^e / \tau\right)}, \qquad (1)$$

where τ represents a temperature parameter [30]. We empirically set τ to 0.07. Similar formulas apply to f_j^h and f_j^g. This loss function brings together features that belong to the same task type in the embedding space, while separating features from different task types.

4.3 Task Type Recognition

Our task type recognition framework is based on the architecture proposed by EHTask [22]. This approach has demonstrated high effectiveness by leveraging a CNN for feature extraction and subsequently processing the temporal features using a BiGRU. Therefore, we also adopt this framework. The framework initiates by employing an encoder to extract temporal features. This encoder comprises three branches, each containing a 1D CNN and a BiGRU. These branches independently process EiH Data, Head Data, and GiW Data, respectively. We propose to pre-train this encoder in the contrastive feature extraction module of Section 4.2. Following the encoder, an Multilayer Perceptron (MLP) block consisting of two fully connected (FC) layers is utilized.

The sampling frequency of our eye-tracking data is set at 25Hz. During the training process, we utilize a sliding window approach to segment the data. We explore various window sizes ranging from 2 seconds to 10 seconds, with a frame interval of 1 frame between adjacent windows. Regarding other parameters like the learning rate and batch size, we maintain the same settings as the EHTask. The network training is conducted on a NVIDIA GeForce RTX 1080 GPU with 11 GB of memory.

4.4 Post-optimization

Based on our observations, we found that there are several frames of incorrect recognition results in the output of our network, as illustrated in the first line of results in Fig. 8. To address this issue, we propose a post-optimization strategy to smooth the results. Let R_i denote the recognized output of the i-th frame. Suppose that the recognized task types $R_1, ..., R_{i-1}$ are task A, and R_i is task type C, where $A \neq C$. In this case, we need to determine whether R_i is a correct task type switching or a recognition error. To address this, we consider the results from the past 2 seconds and the next 1 second to obtain a reasonable result. Specifically, we use the following criteria:

- *Condition 1*: A certain task type appears for more than 80% of the time within the past 2 seconds.
- *Condition 2*: A certain task type appears for more than 80% of the time within the next 1 second.

When *Condition 1* is satisfied, we consider the task type that appeared within the past 2 seconds as A. Similarly, when *Condition 2* is satisfied, we consider the task type that appears within the next 1 second as B. The task type of the current frame is C. If *Condition 2* is met and $B \neq C$, we modify C to B in two cases: 1) *Condition 1* is met, but $A \neq C$; 2) *Condition 1* is not met. If *Condition 2* is not met, but

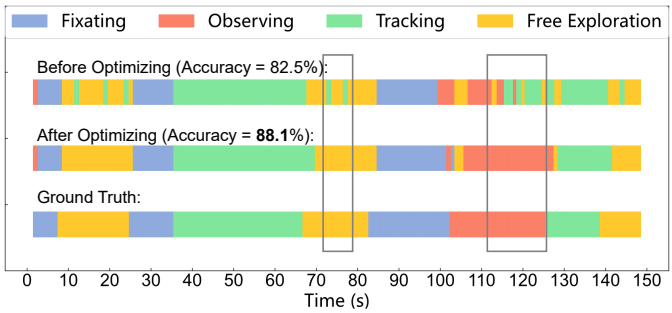


Fig. 8: The recognization results of one sample using the proposed post-optimization method. The first line of results represents the direct output of our network.

Condition 1 is met and $A \neq C$, we modify C to A. If the above strategy is not satisfied, we do not modify C.

Our post-optimization method has been demonstrated to be highly effective in Section 5.2, as it strives to produce smoother results and eliminate error frames with abrupt changes. Fig. 8 provides a specific example of the effectiveness of our method.

5 Experimental Results

In this section, we conduct extensive experiments to compare our method with state-of-the-art methods in task type recognition on our dataset. We perform a cross-user evaluation and a cross-scene evaluation across multiple time windows. We analyze the specific recognition performance for different task types and further examine the recognition results for specific samples. Finally, we conduct an ablation study to validate the effectiveness of different modules.

5.1 Evaluation Metric and Comparisons

As common used in prior works, we used classification accuracy as the metric to evaluate the performance of task type recognition methods. For the evaluation of different methods generalization capability across different users, we conduct five-fold cross-user evaluation. Specifically, the dataset is equally divided into five folds, with four folds used for training and the remaining fold used for testing. Each method is trained and tested five times, with each fold being tested once. The mean recognition accuracy from five tests are employed for following analysis. The similar process is adopted for cross-scene evaluation.

We compared the performance of our model with state-of-the-art methods from task type recognition and time series data prediction.

- EHTask: The EHTask is regarded as the best-performing method among known visual task type recognition approaches, and it serves as the baseline method for our study. The EHTask consists of three CNN+BiGRU models, with the extracted features fed into a two-layer fully connected network for regression.
- Random Forests (RF): Random Forests have also been extensively utilized in task prediction [20, 36]. Hu et al. conducted experiments that proved Random Forests to outperform other machine learning methods, such as Support Vector Machine (SVM) [9] and Boosting Classifier [10], in task recognition. Therefore, we employ Random Forests to represent the performance of conventional machine learning methods.
- MLP with Varying Network Depths: MLP is frequently employed in time series data prediction [2, 14, 54]. In our study, we explore the performance of MLP in task recognition by varying the number of hidden layers (Number = 2, 4, 6). We denote them as MLP-2, MLP-4, and MLP-6, respectively.
- CNN-GRU-Attention (CGA): Attention mechanisms focus on crucial parts of the data using temporal features processed by RNN in time-series prediction [45]. We utilize the implementation provided by Oguiza et al. [42]. The CNN module used is the same as that in EHTask, while the attention mechanism employs the recommended parameters, with a dropout rate set to 0.1.

Table 5: The performance of various methods is evaluated across different time windows. The best method is highlighted in bold font, while the second-best method is emphasized with an underline.

	Time Window (s)	2s	6s	10s
Cross-User	EHTask [22]	72.1%	<u>73.3%</u>	71.0%
	RF [20]	68.3%	70.1%	69.4%
	MLP-2 [14]	64.2%	63.5%	67.8%
	MLP-4 [14]	65.8%	70.5%	70.1%
	MLP-6 [14]	64.9%	70.2%	70.0%
	CGA [42]	72.4%	69.1%	NaN
	TRCLP (Ours)	**76.1%**	76.0%	71.1%
Cross-Scene	EHTask [22]	67.3%	<u>70.4%</u>	68.0%
	RF [20]	59.8%	62.3%	61.2%
	MLP-2 [14]	60.6%	63.5%	62.6%
	MLP-4 [14]	61.3%	64.3%	63.7%
	MLP-6 [14]	60.9%	64.3%	63.6%
	CGA [42]	68.3%	65.9%	64.6%
	TRCLP (Ours)	**71.2%**	71.0%	66.6%

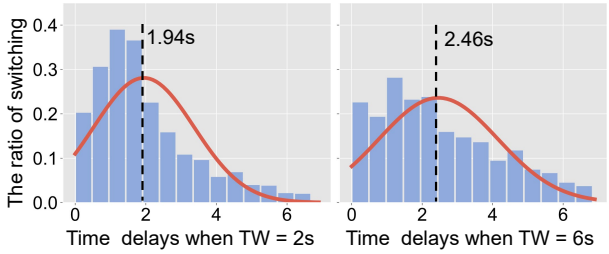


Fig. 9: The comparison of the time delays in recognizing new task types, considering time windows (TWs) of both 2 seconds and 6 seconds. The red line represents the normal distribution of the data.

5.2 Results

Performance comparison of different methods on our dataset. We evaluated the accuracy of our models against state-of-the-art techniques using time windows of 2, 6, and 10 seconds. The 10-second window size was employed in EHTask. This data augmentation mentioned in Section 4.1 was also used for all other methods. The results were shown in Tab. 5, from which we made the following observations. 1) In a cross-user evaluation, our method outperformed the second-best method by a margin of 2.8%, while in cross-scene evaluation, our method slightly outperformed the second-best method. 2) Analysis of different time windows within the same method revealed that while other methods almost reached peak accuracy with a 6-second window, our TRCLP method demonstrated optimal accuracy with a shorter 2-second window. 3) When all methods use a 2-second time window, our method outperforms the second-best method by 3.7% and 2.9% in cross-user evaluation and cross-scene evaluation, respectively.

It is noteworthy that, although the performance improvement of our method is not substantial when considering all time windows, a significant additional advantage is its efficiency in requiring only a 2-second window to achieve optimal recognition results, compared to the 6-second window required by EHTask. This efficiency allows our method to predict task types at least 4 seconds faster than EHTask, enhancing its practical application in real-time scenarios.

Comparison of time delays in recognizing new task types across different windows for the prposed TRCLP. We aimed to investigate whether different time windows have an impact on the time delays during task switching. According to the results of Tab. 5, we considered the comparison of TWs of both 2 seconds and 6 seconds in cross-user evaluation. The results were shown in Fig. 9. Upon analyzing the average time delays, we found that the TW = 2s data inputs yield a more prompt and accurate recognition of newly emerged task types, with a time advantage of 0.52s over the TW = 6s input. The reason

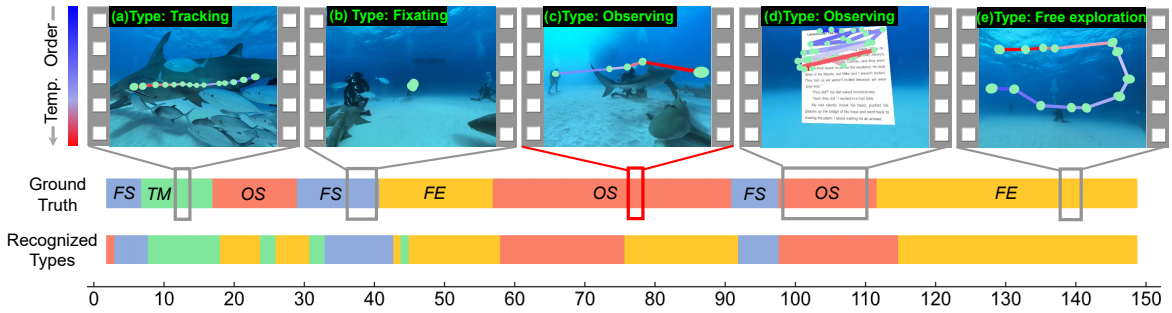


Fig. 10: To illustrate the recognized results, we present a specific example. This example shows the eye movement trajectories of a user while performing four task types in the VR video of the underwater world.

Table 6: Ablation study of each component in our method. The results demonstrate that our method outperforms all ablated models.

Baseline	DA	CL	POST	Cross-User	Cross-Scene
✓				71.8%	66.9%
✓	✓			72.1%	67.6%
✓	✓	✓		72.4%	67.9%
✓	✓	✓	✓	**76.1%**	**71.2%**

we analyze was that the data length of TW = 2s was much shorter compared to TW = 6s. Consequently, when switching to a new task, the data variation grabbed attention more quickly for TW = 2s. Based on the comprehensive analysis above, we decided to adopt TW = 2s as the window size for practical use of our TRCLP.

Ablation study of each component in our method. We conducted ablation studies on our dataset to validate the effectiveness of each component in our method, including the Data Augmentation module (DA), Contrastive Learning module (CL), and Post-optimization module (POST). To compare with the baseline method, we sequentially added each module and retrained the network. We evaluated the performance using cross-user and cross-scene evaluation with 5-fold cross-validation. Table 6 presented the recognition performance of each component in our method, which showed improvement in both evaluations. Notably, the addition of the Post-optimization module resulted in the most significant improvement. Our method achieved a 4.3% improvement over the baseline method in both evaluations.

5.3 Discussion

Our research enhances practical visual task recognition through the introduction of scene-agnostic task types and a novel dataset, characterized by accurate temporal boundaries for multiple task types in every video clip. With this dataset, we can train the task type recognition method that supports free task switching. The proposed TRCLP method achieves promising results using a shorter 2-second window, thereby improving its utility in real-time applications. In the following sections, we analyze the performance of our method across individual task types and explain the recognition results by integrating the specific scene and gaze trajectory analysis.

Analysis of recognition for different visual task types. We present the confusion matrices of recognition results of different task types, as shown in Fig. 11. The diagonal elements indicate the accuracy of each task type, while the off-diagonal elements represent the probability of misclassifying task types. Based on this matrix, the following observations can be made: 1) In cross-user evaluation, *OS* and *TM* achieve an average accuracy of 70%, but in cross-scene evaluation, the average drops to 60%. There is significant confusion (13.4% and 16.6%) between *OS* and *TM* in cross-scene evaluation. It is believed that when the speed of the object being tracked by the user is too fast, smooth pursuit is replaced by catch-up saccades to track the target. Land *et al.* reported that this change occurs when the object's speed exceeds 30°/s [34]. At this point, the saccades resemble the scanning movements of the *OS*. 2) In cross-scene evaluation, *OS* and *TM* have probabilities of 14.0% and 12.6% of being misclassified as *FE*, respectively. This is because in certain scenarios, the eye and head movements of users during *OS* and *TM* resemble those during *FE*. For instance, in

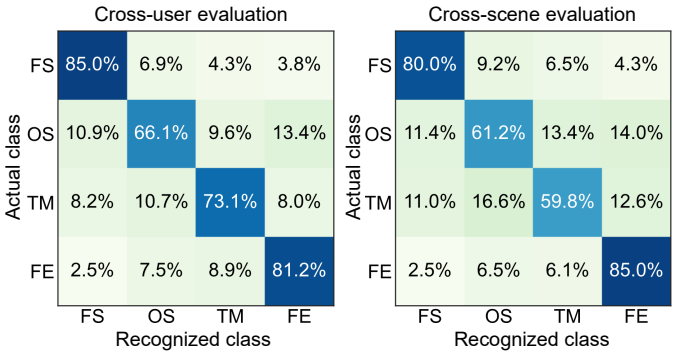


Fig. 11: The confusion matrices of our method are presented for the cross-user (left) and the cross-scene (right) recognition results, with a time window of 2 seconds. These matrices have been normalized based on the ground truth rows.

Fig. 10 (c), users were instructed to count the number of people in the scene. Due to the dispersed positioning of the people in the scene, users exhibited significant head and eye movements. This eye movement behavior is similar to that of *FE*, as shown in Fig. 10 (e). 3) Both *FS* and *FE* achieve accuracy rates above 80% in both cross-user and cross-scene evaluations. This can be attributed to significant differences observed in fixation duration and head movement magnitude between these two task types and the others.

Analysis of specific example. We provide a specific example to elaborate on the recognition results. Fig. 10 displays the eye movement trajectories of a user observing the underwater world scene. The temporal characteristics of eye movements vary significantly across the four visual task types. In the task type of fixating, the fixation points exhibit slight movement within a very small area, as depicted in Fig. 10 (b). In the task type of observing, there is typically an sequential transition between objects, as shown in Fig. 10 (d). The tracking, on the other hand, involves a relatively smooth gaze trajectory without sudden turns, as demonstrated in Fig. 10 (a). Finally, in the free exploration, the gaze points move irregularly, accompanied by large-angle movements, as illustrated in Fig. 10 (e).

6 Application Demos for Task Type Recognition

In this section, we explore the applications of task type recognition in XR systems, which can adapt virtual content displays to user needs, facilitating smoother task completion. Our application design involves three steps. 1) When a user performs a specific task within an XR system, such as tracking a walker as shown in Fig. 2 (b), the system uses our task type recognition method to identify the task type as *TM*. 2) An object detection algorithm detects the target of the user's gaze, obtaining its label (*e.g.*, "person") and contour. 3) Based on the recognized task type and target label, the system infers the user's specific visual task as "track a person" and adapts virtual content displays, such as showing the contour and trajectory of the tracked target.

The main contribution of this paper is task type recognition and we do not use any object detection algorithms. Therefore, in the following application demos, target locations are pre-annotated offline.

(a) Fixation Assistance (Product recommendation)

(b) Reading Assistance (Highlighting the line to read)

(c) Tracking Assistance (Showing the boundingbox and trajectory)

Fig. 12: We show the applications of task type recognition through three examples. (a) Fixation Assistance (Product recommendation). (b) Reading Assistance (Highlighting the line to read). (c) Tracking Assistance (Showing the boundingbox and trajectory).

A promising approach is to use the Segment Anything Model (SAM) proposed by Meta [32], which can generalize to unfamiliar objects without additional training. SAM can use the user's gaze points as prompts to recognize the label and contour of the target. In summary, by integrating our task type recognition method with SAM, it is feasible to design task-aware intelligent applications, referred to as intelligent assistance, to help users complete tasks more efficiently.

Below, we design three specific applications to provide valuable insights for content developers. For further details, please refer to the supplemental video. **1) Fixation Assistance (Product Recommendation).** When our system recognizes that the user is fixating on a television, it triggers an internet search for discounted televisions to recommend to the user, as pre-configured for this demo (Fig. 12 (a)). **2) Reading Assistance (Highlighting the Reading Line).** Reading is a specific visual task under the task type *OS*. When our system infers the user's task as reading, it triggers the reading assistance (Fig. 12 (b)). The designed assistance highlights the text line being read while dimming the others, aiding user focus. **3) Tracking Assistance (Showing the boundingbox and trajectory)**. When our system recognizes that the user is tracking a person, it triggers the tracking assistance, which involves displaying the bounding box and historical movement trajectory, as depicted in Fig. 12 (c).

Besides the specific applications mentioned above, we also explore both explicit and implicit uses of our visual task types to fully uncover the potential applications of this technology. **Explicit Uses.** 1) Virtual Museum Tours: Recognizing the task type as *FS* can trigger the display of additional information about exhibits, such as the artist's name and artwork history. For *OS*, the system can guide users through a tour of related items, like different exhibits. 2) AR Navigation: When a user's task type is *FS*, such as a landmark or building, the AR system can display relevant information, including distance. For *TM*, like a vehicle or person, the system can provide real-time updates about movement or identity, such as speed, overlaid around the moving object. 3) Adaptive User Interfaces: By recognizing the user's current task type, the system can adapt the user interface to prioritize relevant information. For *FE*, the interface can minimize clutter and provide a wider field of view. For *OS*, it can highlight the next object in the sequence.

Implicit Uses. 1) Attention Analysis: In a classroom, recognizing *FS*, *e.g.*, fixating on the teacher or teaching materials, can assess the user's level of focus. 2) Training Simulations: In military simulations, recognizing *TM*, such as tracking a fighter jet, can evaluate the user's response time to specific procedures. 3) Cognitive Monitoring: Recognizing the timing of task switching can indicate cognitive load levels, allowing adaptive systems to adjust difficulty or provide supplemental information when the user is overwhelmed or under-engaged.

7 System Limitations and Future Work

Although our task type recognition method can support free switching between task types, we found several limitations that we plan to address in future work. Firstly, as discussed in Section 5.3, there is an overlap in eye movement patterns, particularly between *OS* and *TM*. This occurs when an object's motion exceeds a certain speed (*e.g.*, 30°/s in visual angle), causing smooth pursuit to transition into catch-up saccades. A potential solution is to introduce gaze target detection when eye movements overlap. This is because in *OS*, users observe a series of objects and switch their gaze between them, whereas in *TM*, they continuously focus on a single object.

Secondly, to ensure that the movement of targets does not introduce any ambiguity in the task instruction, the panoramic camera is required to be stationary. Participants stood at the center of the camera position and watched the 360-degree panoramic video without moving. We plan to consider more complex scenarios, such as user movement, which is more in line with actual usage habits. In the future, we plan to design new task instructions for user movement scenarios.

Thirdly, this paper selects *FE* to represent the visual task type associated with irregular saccades. In fact, irregular saccades are the most complex type of eye movement in real-world contexts, potentially corresponding to various task types. As mentioned in Section 3.1, these task types include "object searching", although it is not explored in this paper. Another example is observing a painting, which also involves numerous irregular saccades [33, 55]. Future research plans include a more in-depth analysis of task types involving irregular saccades to enhance the practical application of visual task type recognition.

8 Conclusion

In this study, we focused on egocentric gaze-aware visual task type recognition in immersive VR environments. We proposed four versatile visual task types to enable task type recognition across a broader range of scenarios. A dataset was created by annotating these task types on 15 360-degree VR videos, facilitating free switching between multiple task types. A user study captured eye and head movements of 20 participants performing these tasks. We present a novel learning-based approach for recognizing task types that outperforms state-of-the-art methods in terms of recognition accuracy and the required length of time window. Three examples demonstrated the applications of task type recognition. This study provides a technological foundation and valuable insights for the intelligent development of XR systems.

Acknowledgments

The authors wish to thank A, B, and C. This work was supported in part by a grant from XYZ.

References

[1] S. Agnoli, S. Mastria, M. Zanon, and G. Corazza. Dopamine supports idea originality: the role of spontaneous eye blink rate on divergent thinking. *Psychological Research*, 87, 02 2022. doi: 10.1007/s00426-022-01658-y 2

[2] I. Aizenberg, L. Sheremetov, L. Villa-Vargas, and J. Martinez-Muñoz. Multilayer neural network with multi-valued neurons in time series forecasting of oil production. *Neurocomputing*, 175:980–989, 2016. 7

[3] D. H. Ballard, M. M. Hayhoe, and J. B. Pelz. Memory representations in natural tasks. *Journal of cognitive neuroscience*, 7(1):66–80, 1995. 2

[4] S. Barteit, L. Lanfermann, T. Bärnighausen, F. Neuhann, and C. Beiersmann. Augmented, mixed, and virtual reality-based head-mounted devices for medical education: Systematic review. *JMIR Serious Games*, 9(3):e29080, Jul 2021. doi: 10.2196/29080 1

[5] K. Bektaş, J. Strecker, S. Mayer, D. K. Garcia, J. Hermann, K. E. Jenß, Y. S. Antille, and M. Solèr. Gear: Gaze-enabled augmented reality for human activity recognition. In *Proceedings of the 2023 Symposium on Eye Tracking Research and Applications*, ETRA '23, article no. 9, 9 pages. Association for Computing Machinery, New York, NY, USA, 2023. doi: 10.1145/3588015.3588402 2, 3

[6] J. M. Bird, P. A. Smart, D. J. Harris, L. A. Phillips, G. Giannachi, and S. J. Vine. A magic leap in tourism: Intended and realized experience of head-mounted augmented reality in a museum context. *Journal of Travel Research*, 62(7):1427–1447, 2023. 2

[7] A. Borji and L. Itti. State-of-the-art in visual attention modeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):185–207, 2012. 1

[8] A. Borji and L. Itti. Defending yarbus: Eye movements reveal observers' task. *Journal of Vision*, 14(3):29–29, 03 2014. doi: 10.1167/14.3.29 2, 3

[9] A. Bulling, J. A. Ward, H. Gellersen, and G. Tröster. Eye movement analysis for activity recognition using electrooculography. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):741–753, 2011. doi: 10.1109/TPAMI.2010.86 2, 3, 7

[10] A. Coutrot, J. H. Hsiao, and A. B. Chan. Scanpath modeling and classification with hidden markov models. *Behavior research methods*, 50(1):362–379, 2018. 2, 7

[11] B. David-John, C. Peacock, T. Zhang, T. S. Murdison, H. Benko, and T. R. Jonker. Towards gaze-based prediction of the intent to interact in virtual reality. In *ACM Symposium on Eye Tracking Research and Applications*, ETRA '21 Short Papers, article no. 2, 7 pages. Association for Computing Machinery, New York, NY, USA, 2021. doi: 10.1145/3448018.3458008 1, 2

[12] L. T. De Paolis and V. De Luca. The impact of the input interface in a virtual environment: the vive controller and the myo armband. *Virtual Reality*, 24(3):483–502, 2020. 1

[13] S. Frintrop, E. Rome, and H. I. Christensen. Computational visual attention systems and their cognitive foundations: A survey. *ACM Transactions on Applied Perception (TAP)*, 7(1):1–39, 2010. 1

[14] M. W. Gardner and S. Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636, 1998. 7

[15] J. Hadnett-Hunter, G. Nicolaou, E. O'Neill, and M. Proulx. The effect of task on visual attention in interactive virtual environments. *ACM Transactions on Applied Perception (TAP)*, 16(3):1–17, 2019. 1

[16] M. Hayhoe and D. Ballard. Eye movements in natural behavior. *Trends in cognitive sciences*, 9(4):188–194, 2005. 2

[17] J. M. Henderson and A. Hollingworth. Chapter 12 - eye movements during scene viewing: An overview. In G. Underwood, ed., *Eye Guidance in Reading and Scene Perception*, pp. 269–293. Elsevier Science Ltd, Amsterdam, 1998. doi: 10.1016/B978-008043361-5/50013-4 3

[18] J. M. Henderson, S. V. Shinkareva, J. Wang, S. G. Luke, and J. Olejarczyk. Predicting cognitive state from eye movements. *PloS one*, 8(5):e64937, 2013. 2

[19] R. Hessels, D. Niehorster, M. Nyström, R. Andersson, and I. Hooge. Is the eye-movement field confused about fixations and saccades? a survey among 124 researchers. *Royal Society Open Science*, 5:180502, 08 2018. doi: 10.1098/rsos.180502 2

[20] J. Hild, M. Voit, C. Kühnle, and J. Beyerer. Predicting observer's task from eye movement patterns during motion image analysis. article no. 58, 5 pages, 2018. doi: 10.1145/3204493.3204575 2, 3, 7

[21] Z. Hu, A. Bulling, S. Li, and G. Wang. Fixationnet: Forecasting eye fixations in task-oriented virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2681–2690, 2021. 1

[22] Z. Hu, A. Bulling, S. Li, and G. Wang. Ehtask: Recognizing user tasks from eye and head movements in immersive virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, 29(4):1992–2004, 2023. doi: 10.1109/TVCG.2021.3138902 2, 3, 5, 6, 7

[23] Z. Hu, S. Li, C. Zhang, K. Yi, G. Wang, and D. Manocha. Dgaze: Cnn-based gaze prediction in dynamic scenes. *IEEE Transactions on Visualization and Computer Graphics*, 26(5):1902–1911, 2020. 1

[24] Z. Hu, C. Zhang, S. Li, G. Wang, and D. Manocha. Sgaze: A data-driven eye-head coordination model for realtime gaze prediction. *IEEE Transactions on Visualization and Computer Graphics*, 25(5):2002–2010, 2019. 1

[25] S. Hutt, K. Krasich, J. R. Brockmole, and S. K. D'Mello. Breaking out of the lab: Mitigating mind wandering with gaze-based attention-aware technology in classrooms. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2021. 1

[26] H. Idrees, A. Zamir, Y.-G. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah. The thumos challenge on action recognition for videos "in the wild". *Computer Vision and Image Understanding*, 155, 04 2016. doi: 10.1016/j.cviu.2016.10.018 4

[27] S. Ishimaru, K. Hoshika, K. Kunze, K. Kise, and A. Dengel. Towards reading trackers in the wild: Detecting reading activities by eog glasses and deep neural networks. In *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*, UbiComp '17, 8 pages, p. 704–711. Association for Computing Machinery, New York, NY, USA, 2017. doi: 10.1145/3123024.3129271 2

[28] T. R. Jonker, R. Desai, K. Carlberg, J. Hillis, S. Keller, and H. Benko. The role of ai in mixed and augmented reality interactions. In *CHI2020 ai4hci Workshop Proceedings. ACM*, 2020. 1

[29] A. Keshava, A. Aumeistere, K. Izdebski, and P. Konig. Decoding task from oculomotor behavior in virtual reality. In *ACM Symposium on Eye Tracking Research and Applications*, pp. 1–5, 2020. 1

[30] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020. 6

[31] P. Kiefer, I. Giannopoulos, and R. Martin. Using eye movements to recognize activities on cartographic maps. 11 2013. doi: 10.1145/2525314.2525467 2, 3

[32] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything, 2023. 9

[33] G. Lan, T. Scargill, and M. Gorlatova. Eyesyn: Psychology-inspired eye movement synthesis for gaze-based activity recognition. In *2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pp. 233–246, 2022. doi: 10.1109/IPSN54338.2022.00026 1, 2, 3, 9

[34] M. Land and B. Tatler. *Looking and Acting: Vision and eye movements in natural behaviour*. Oxford University Press, 07 2009. doi: 10.1093/acprof:oso/9780198570943.001.0001 2, 8

[35] L. Larsson, M. Nyström, and M. Stridh. Detection of saccades and post-saccadic oscillations in the presence of smooth pursuit. *IEEE Transactions on Biomedical Engineering*, 60(9):2484–2493, 2013. doi: 10.1109/TBME.2013.2258918 2, 3

[36] H. Liao, W. Dong, H. Huang, G. Gartner, and H. Liu. Inferring user tasks in pedestrian navigation from eye movement data in real-world environments. *International Journal of Geographical Information Science*, 33(4):739–763, 2019. 7

[37] F. Lu and Y. Xu. Exploring spatial ui transition mechanisms with head-worn augmented reality. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, article no. 550, 16 pages. Association for Computing Machinery, New York, NY, USA, 2022. doi: 10.1145/3491102.3517723 1

[38] S. Malpica, D. Martin, A. Serrano, D. Gutierrez, and B. Masia. Task-dependent visual behavior in immersive environments: A comparative study of free exploration, memory and visual search. *IEEE Transactions on Visualization and Computer Graphics*, 2023. 1, 2, 3

[39] S. Martinez-Conde, S. Macknik, and D. Hubel. The role of fixational eye movements in visual perception. *Nature reviews. Neuroscience*, 5:229–40, 04 2004. doi: 10.1038/nrn1348 2

[40] H. Martínez, D. Skournetou, J. Hyppölä, S. Laukkanen, and A. Heikkilä. Drivers and bottlenecks in the adoption of augmented reality applications. *Journal of Multimedia Theory and Applications*, 2:27–44, 03 2014. doi:

10.11159/jmta.2014.004 1

[41] K. Min and J. J. Corso. Integrating human gaze into attention for egocentric activity recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1069–1078, January 2021. 2

[42] I. Oguiza. tsai - a state-of-the-art deep learning library for time series and sequential data. Github, 2022. https://github.com/timeseriesAI/tsai. 7

[43] K. Pfeuffer, Y. Abdrabou, A. Esteves, R. Rivu, Y. Abdelrahman, S. Meitner, A. Saadi, and F. Alt. Artention: A design space for gaze-adaptive user interfaces in augmented reality. *Computers & Graphics*, 95:1–12, 2021. 1, 2

[44] P. Prasse, D. R. Reich, S. Makowski, S. Ahn, T. Scheffer, and L. A. Jäger. Sp-eyegan: Generating synthetic eye movement data with generative adversarial networks. In *Proceedings of the 2023 Symposium on Eye Tracking Research and Applications*, ETRA '23, article no. 18, 9 pages. Association for Computing Machinery, New York, NY, USA, 2023. doi: 10.1145/3588015.3588410 3

[45] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*, 2017. 7

[46] K. Rayner, X. Li, C. C. Williams, K. R. Cave, and A. D. Well. Eye movements during information processing tasks: Individual differences and cultural effects. *Vision Research*, 47(21):2714–2726, 2007. doi: 10.1016/j.visres.2007.05.007 3

[47] K. Rook, B. Witt, R. Bailey, J. Geigel, P. Hu, and A. Kothari. A study of user intent in immersive smart spaces. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 227–232, 2019. doi: 10.1109/PERCOMW.2019.8730692 2

[48] A. Seeliger, R. Weibel, and S. Feuerriegel. Context-adaptive visual cues for safe navigation in augmented reality using machine learning. *International Journal of Human-Computer Interaction*, pp. 1–21, 09 2022. doi: 10.1080/10447318.2022.2122114 2

[49] K. Soomro, A. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, 12 2012. 4

[50] N. Srivastava, J. Newn, and E. Velloso. Combining low and mid-level gaze features for desktop activity recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(4), article no. 189, 27 pages, dec 2018. doi: 10.1145/3287067 2, 3

[51] T. T. Um, F. M. J. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulić. Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ICMI '17, 5 pages, p. 216–220. Association for Computing Machinery, New York, NY, USA, 2017. doi: 10.1145/3136755.3136817 5

[52] L.-M. Vortmann and F. Putze. Attention-aware brain computer interface to avoid distractions in augmented reality. In *Extended abstracts of the 2020 chi conference on human factors in computing systems*, pp. 1–8, 2020. 1

[53] Z. Wang, Y. Zhao, and F. Lu. Gaze-vergence-controlled see-through vision in augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 28(11):3843–3853, nov 2022. doi: 10.1109/TVCG.2022.3203110 1

[54] I. R. Widiasari, L. E. Nugroho, et al. Deep learning multilayer perceptron (mlp) for flood prediction model using wireless sensor network based hydrology time series data mining. In *2017 International Conference on Innovative and Creative Information Technology (ICITech)*, pp. 1–5. IEEE, 2017. 7

[55] A. L. Yarbus. *Eye movements and vision*. Springer, 1967. 1, 2, 9

[56] B. Zhou and S. Güven. Fine-grained visual recognition in mobile augmented reality for technical support. *IEEE Transactions on Visualization and Computer Graphics*, 26(12):3514–3523, 2020. doi: 10.1109/TVCG.2020.3023635 1