[ASE 25] VRExplorer: A Model-based Approach for Semi-Automated Testing of Virtual Reality Scenes

Zhengyang Zhu, Hong-Ning Dai*, Hanyang Guo, Zeqin Liao, Zibin Zheng

Sun Yat-sen University & Peng Cheng Laboratory & Hong Kong Baptist University

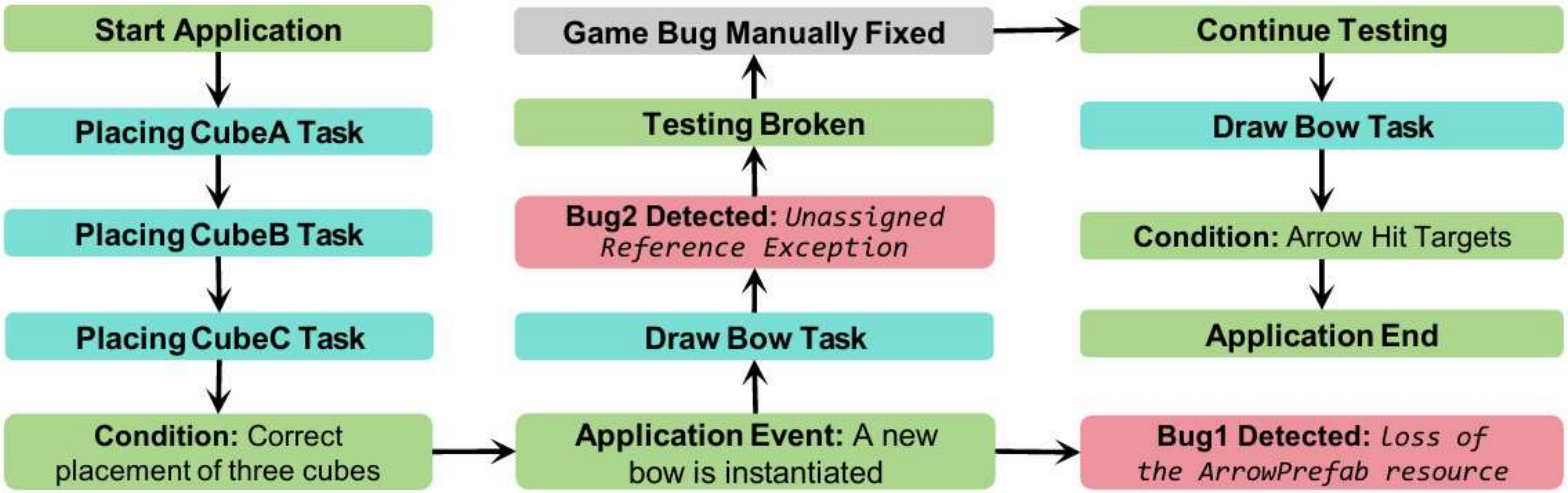
Introduction

Research Background

- Manual VR testing is **labor-intensive and inefficient**
- **Limitation of SOTA**: **VRTest** (ICSE 22 short) & **VRGuide** (only can \`click\`, struggle to perform other VR-specific actions)
- **Limitation of CV/LLM Methods**: only analyze screenshots, **lack code execution capability**)
- **Core Challenge:** It remains difficult to test VR apps comprehensively and efficiently

Introduction

An Example of Challenge



Start Application

Placing CubeA Task

Placing CubeB Task

Placing CubeC Task

Condition: Correct placement of three cubes

Application Event: A new bow is instantiated

Draw Bow Task

Bug2 Detected: *Unassigned Reference Exception*

Testing Broken

Game Bug Manually Fixed

Bug1 Detected: *Loss of the ArrowPrefab resource*

Continue Testing

Draw Bow Task

Condition: Arrow Hit Targets

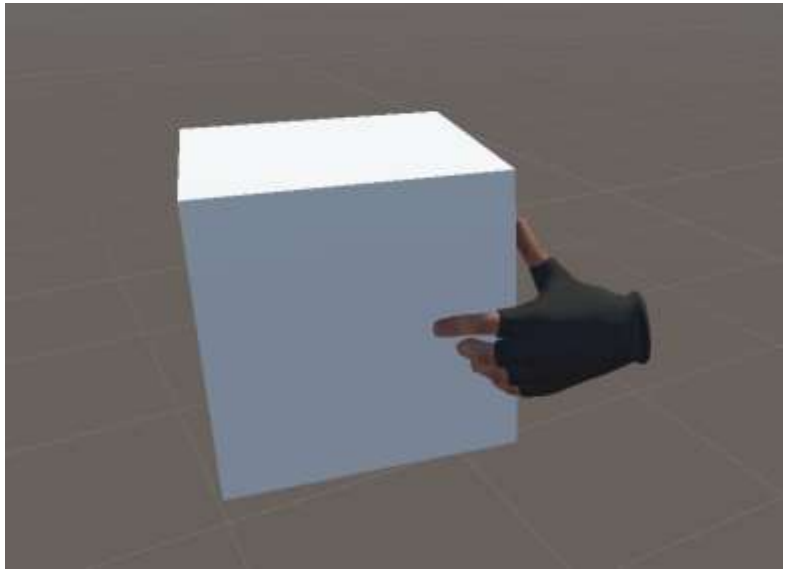
Application End

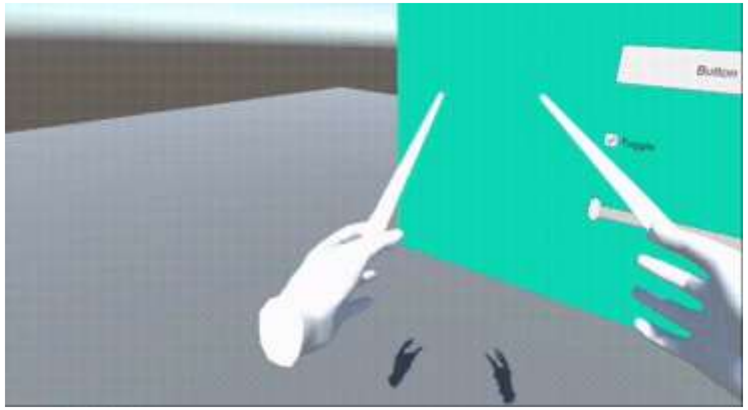
Fig. 9: Testing Process of EscapeGameVR

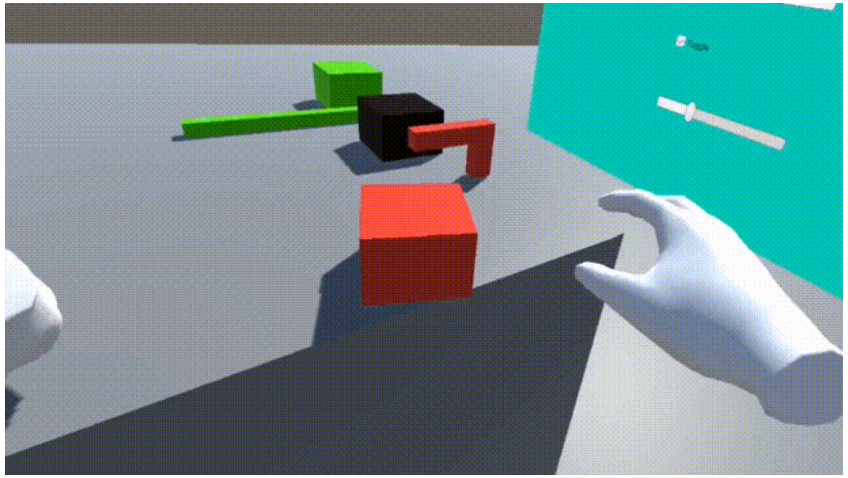
Introduction

Challenges

- **Diversity of interactions** (Grab, Move, Throw, Raycast, Press, Touch, Pull, Trigger, Transform, Shoot/Fire)







Introduction

Challenges

- **Diversity of interactions**

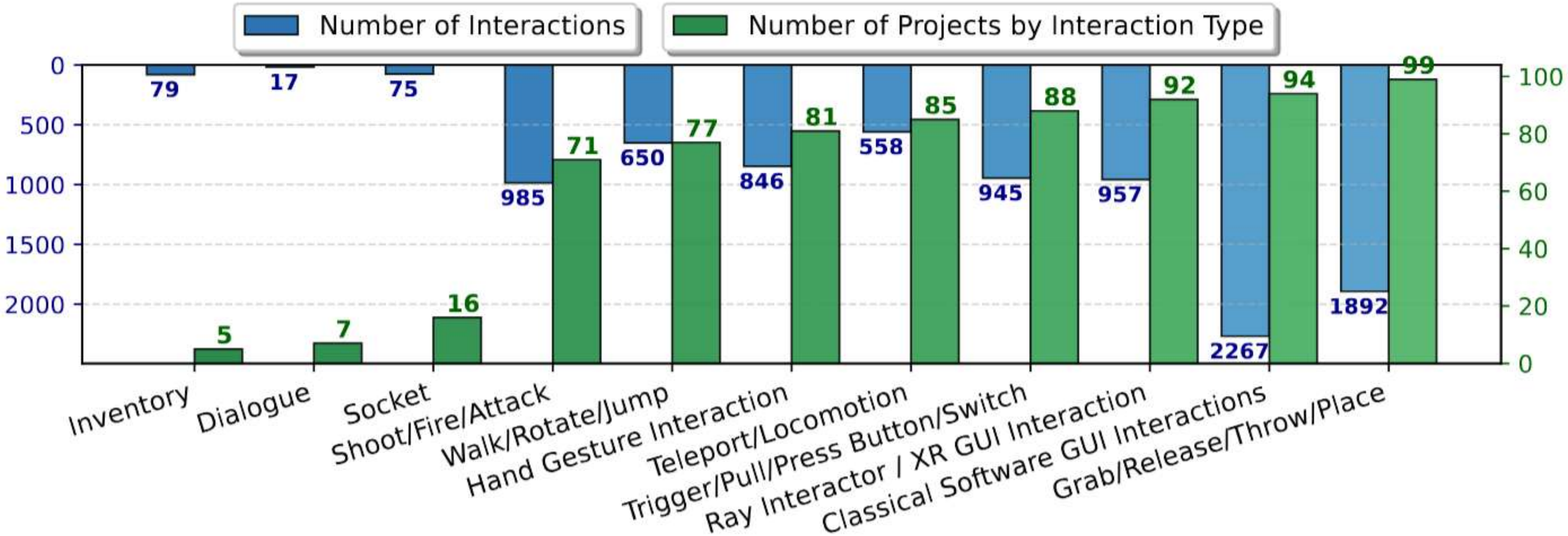
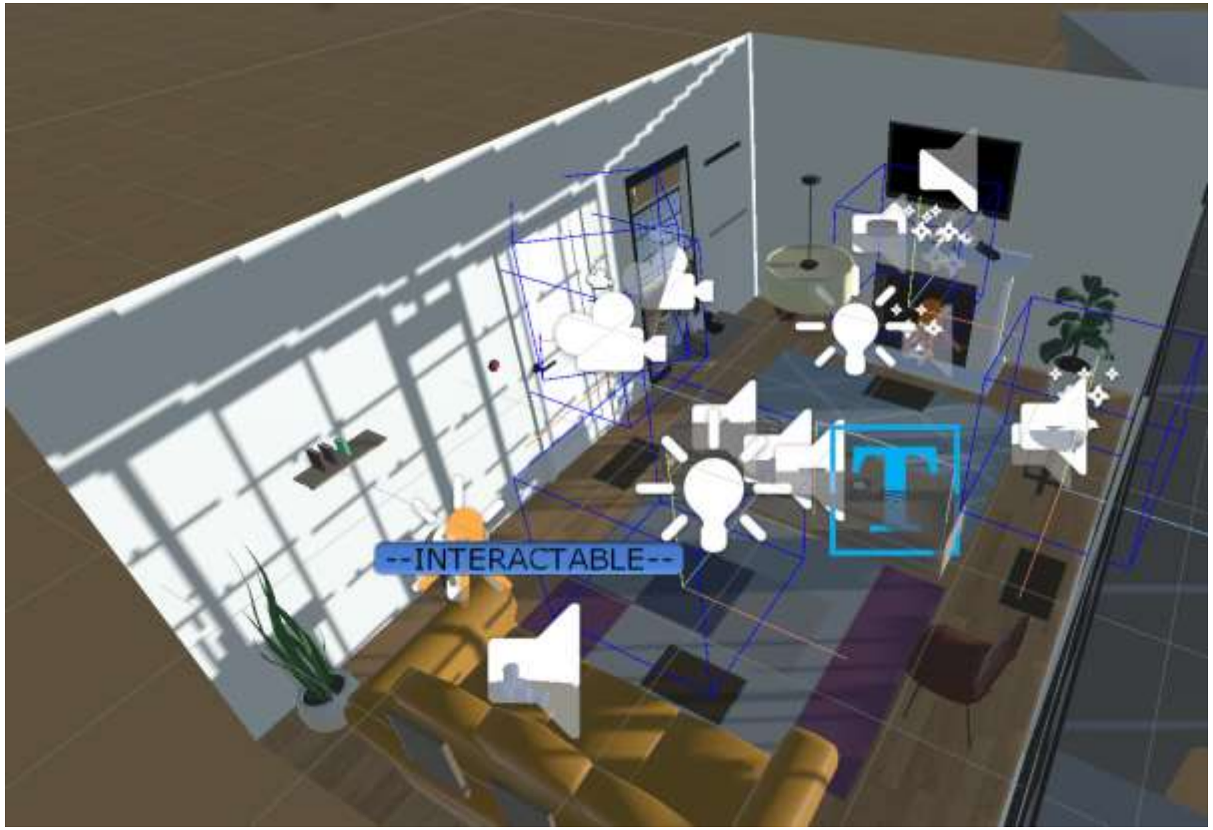


Fig. 6: Distribution of Interactions in Dataset.

Introduction

Challenges

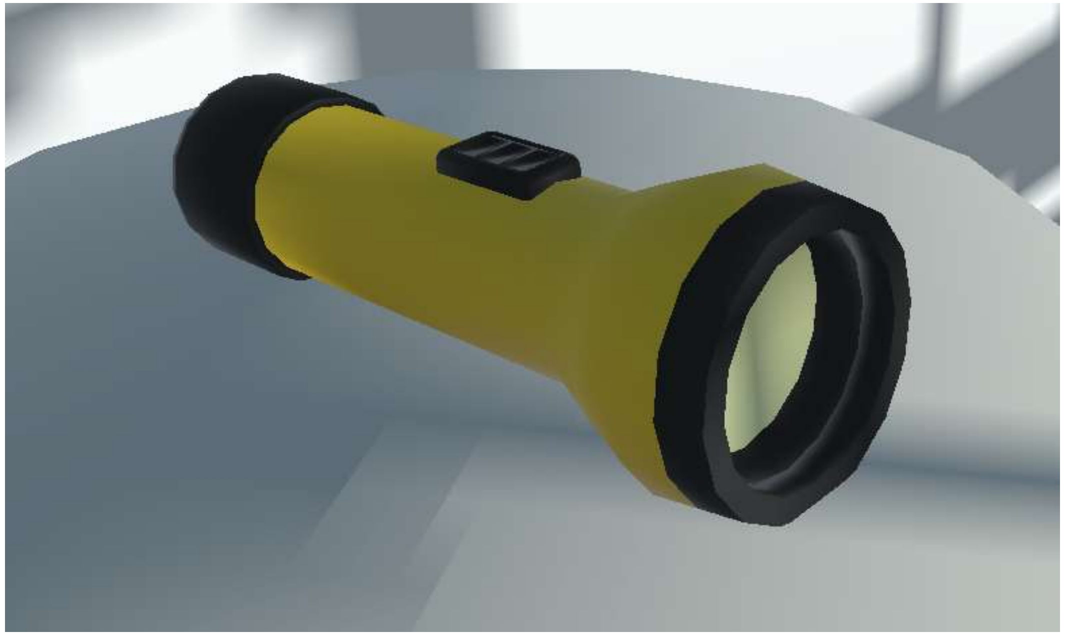
- **Complex3D virtual environments**



Introduction

Challenges

- **Sequence of tasks** (find a key ->open a door -> turn a handle -> finally pressing a button to escape; Grab a Lighter -> light a candle, Grab a light source -> press buttons)



Approach

Contribution and Novelty

- To solve these challenges

   1. Model Abstraction

   2. EAT Framework (based on OOP)

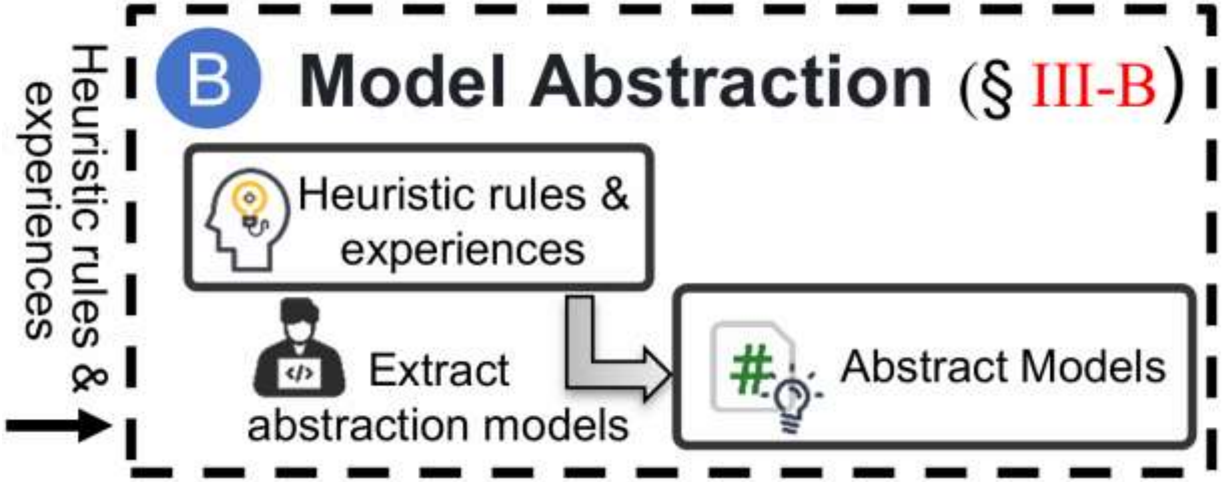
   3. VRExplorer Agent (Automated Exploration of Scenes)

Approach

■ Contribution and Novelty

➢ Model Abstraction

Project Analysis (by Experienced Test Engineers)

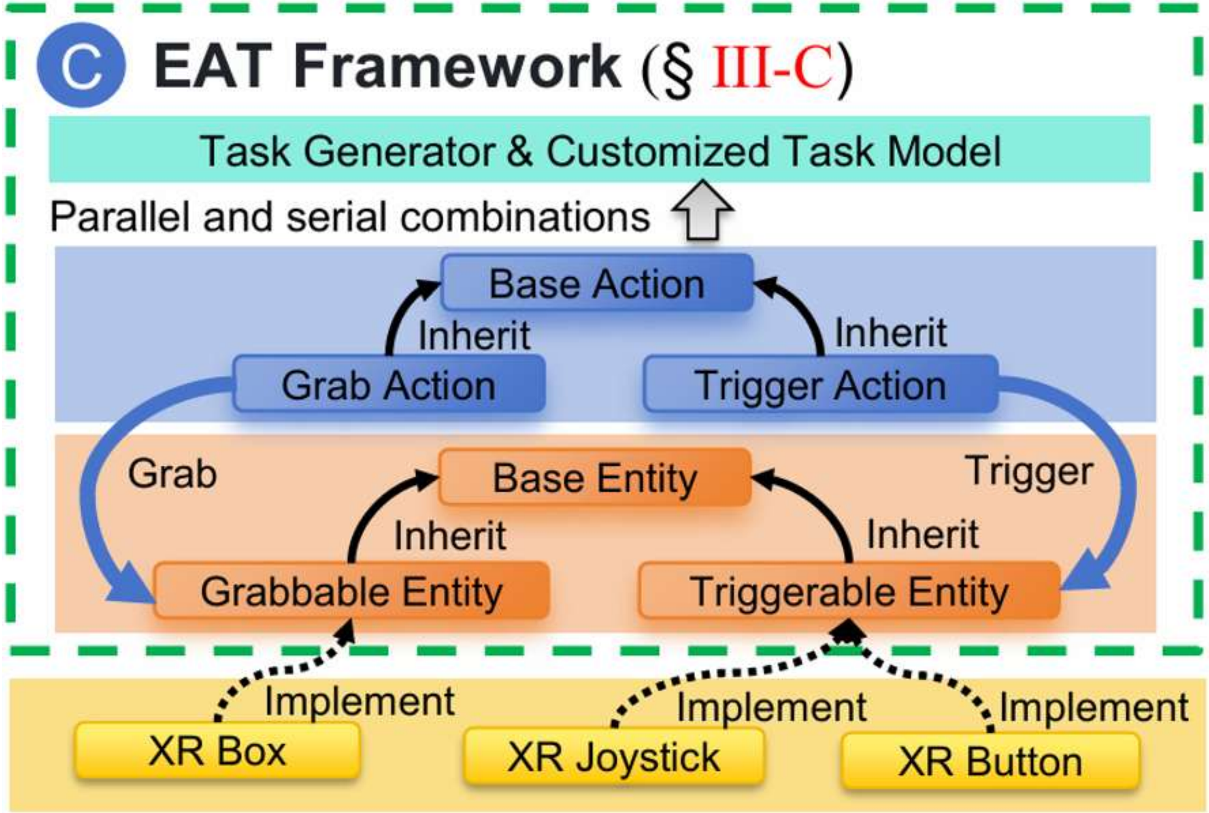

Heuristic rules & experiences
B
Model Abstraction (§ III-B)
Heuristic rules & experiences
Extract abstraction models
Abstract Models

Approach

■ Contribution and Novelty

➢ EAT Framework (based on OOP)

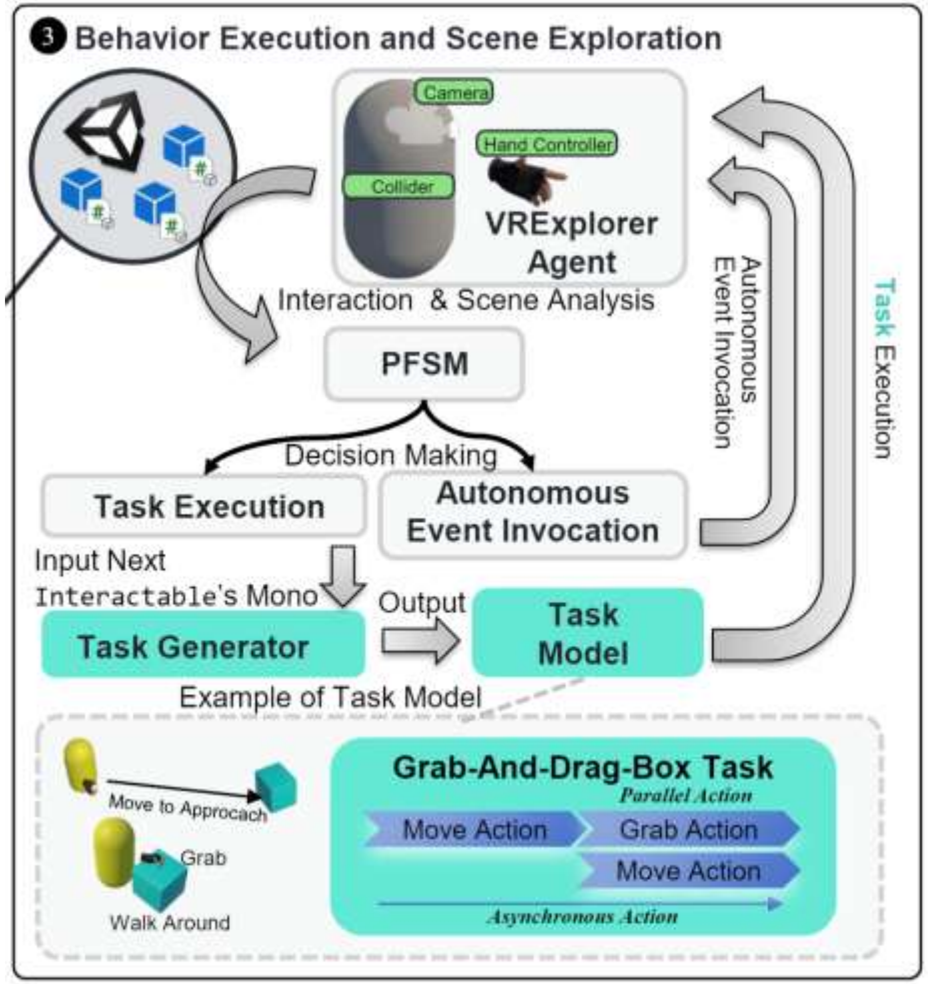
- ➢ **Entity:** Abstract Interactables
- ➢ **Action:** Abstract VR Actions
- ➢ **Task:** Abstract Action Sequences



Approach

Contribution and Novelty

- VRExplorer Agent



Approach

■ Overview

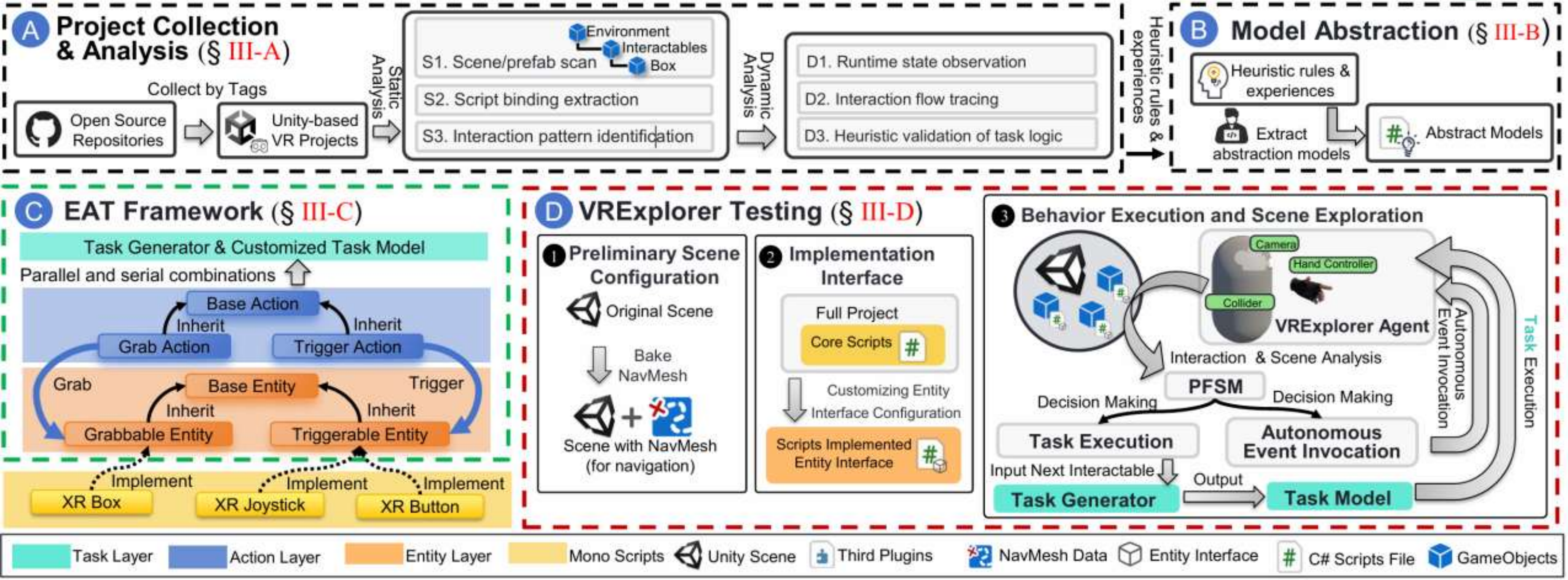

A Project Collection & Analysis (§ III-A)
Collect by Tags
Open Source Repositories
Unity-based VR Projects
Static Analysis
S1. Scene/prefab scan
Environment
Interactables
Box
S2. Script binding extraction
S3. Interaction pattern identification
Dynamic Analysis
D1. Runtime state observation
D2. Interaction flow tracing
D3. Heuristic validation of task logic
Heuristic rules & experiences
B Model Abstraction (§ III-B)
Heuristic rules & experiences
Extract abstraction models
Abstract Models
C EAT Framework (§ III-C)
Task Generator & Customized Task Model
Parallel and serial combinations
Base Action
Inherit
Inherit
Grab Action
Trigger Action
Grab
Trigger
Base Entity
Inherit
Inherit
Grabbable Entity
Triggerable Entity
Implement
Implement
Implement
XR Box
XR Joystick
XR Button
D VRExplorer Testing (§ III-D)
1 Preliminary Scene Configuration
Original Scene
Bake NavMesh
Scene with NavMesh (for navigation)
2 Implementation Interface
Full Project
Core Scripts
Customizing Entity Interface Configuration
Scripts Implemented Entity Interface
3 Behavior Execution and Scene Exploration
Camera
Hand Controller
Collider
VRExplorer Agent
Interaction & Scene Analysis
PFSM
Decision Making
Decision Making
Task Execution
Autonomous Event Invocation
Autonomous Event Invocation
Task Execution
Input Next Interactable
Output
Task Generator
Task Model
Task Layer
Action Layer
Entity Layer
Mono Scripts
Unity Scene
Third Plugins
NavMesh Data
Entity Interface
C# Scripts File
GameObjects

Fig. 1: Overview of *VRExplorer*

Evaluation of VRExplorer

Evaluation Projects

- We evaluate VRExplorer on **11** representative VR projects
- +122.8% Executable Lines of Code Cov. vs SOTA
- +52.8% Method Cov. vs SOTA
- successfully detected **3** real-world bugs.

Evaluation of VRExplorer

Evaluation Projects

- Evaluation on Complex 3D VR Projects

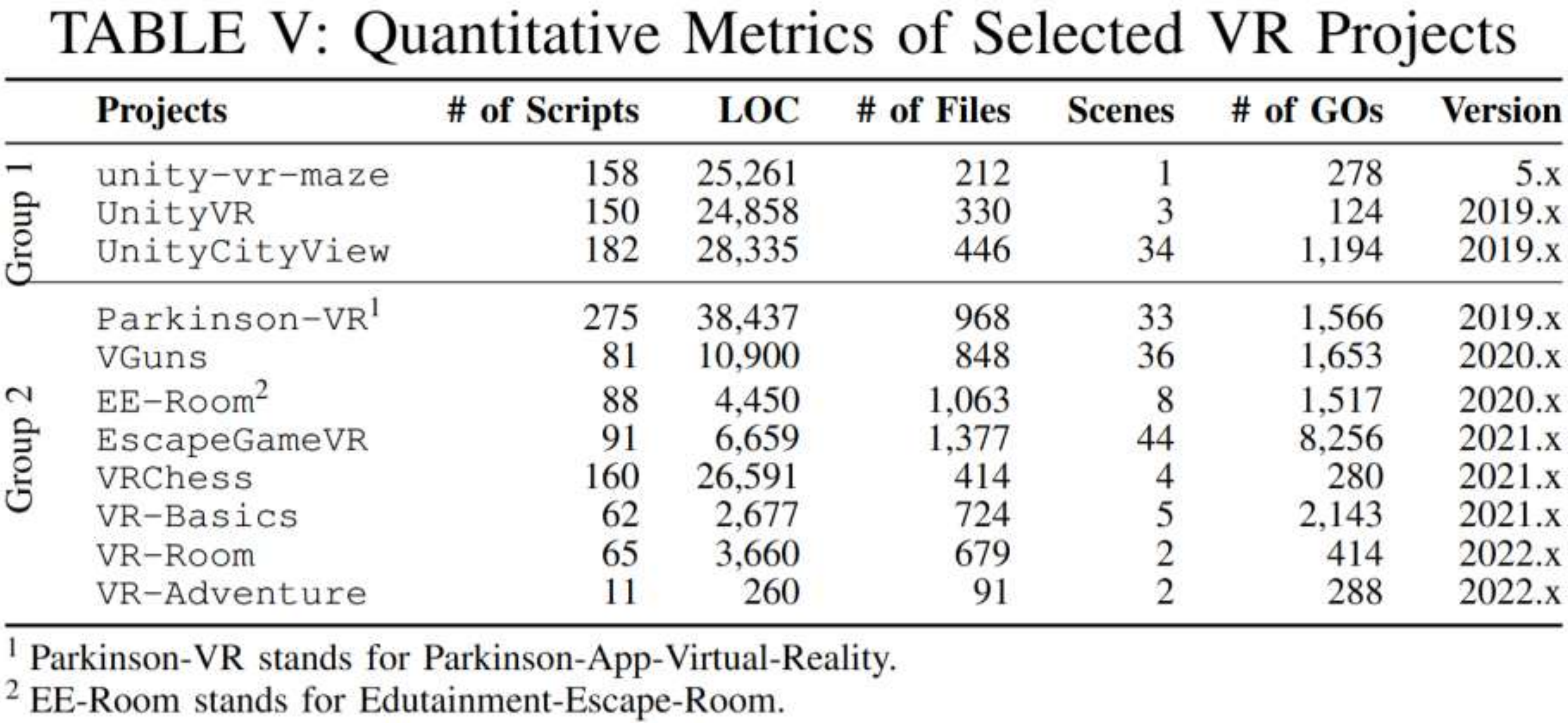


TABLE V: Quantitative Metrics of Selected VR Projects

	Projects	# of Scripts	LOC	# of Files	Scenes	# of GOs	Version
Group 1	unity-vr-maze	158	25,261	212	1	278	5.x
	UnityVR	150	24,858	330	3	124	2019.x
	UnityCityView	182	28,335	446	34	1,194	2019.x
Group 2	Parkinson-VR[1]	275	38,437	968	33	1,566	2019.x
	VGuns	81	10,900	848	36	1,653	2020.x
	EE-Room[2]	88	4,450	1,063	8	1,517	2020.x
	EscapeGameVR	91	6,659	1,377	44	8,256	2021.x
	VRChess	160	26,591	414	4	280	2021.x
	VR-Basics	62	2,677	724	5	2,143	2021.x
	VR-Room	65	3,660	679	2	414	2022.x
	VR-Adventure	11	260	91	2	288	2022.x

[1] Parkinson-VR stands for Parkinson-App-Virtual-Reality.

[2] EE-Room stands for Edutainment-Escape-Room.

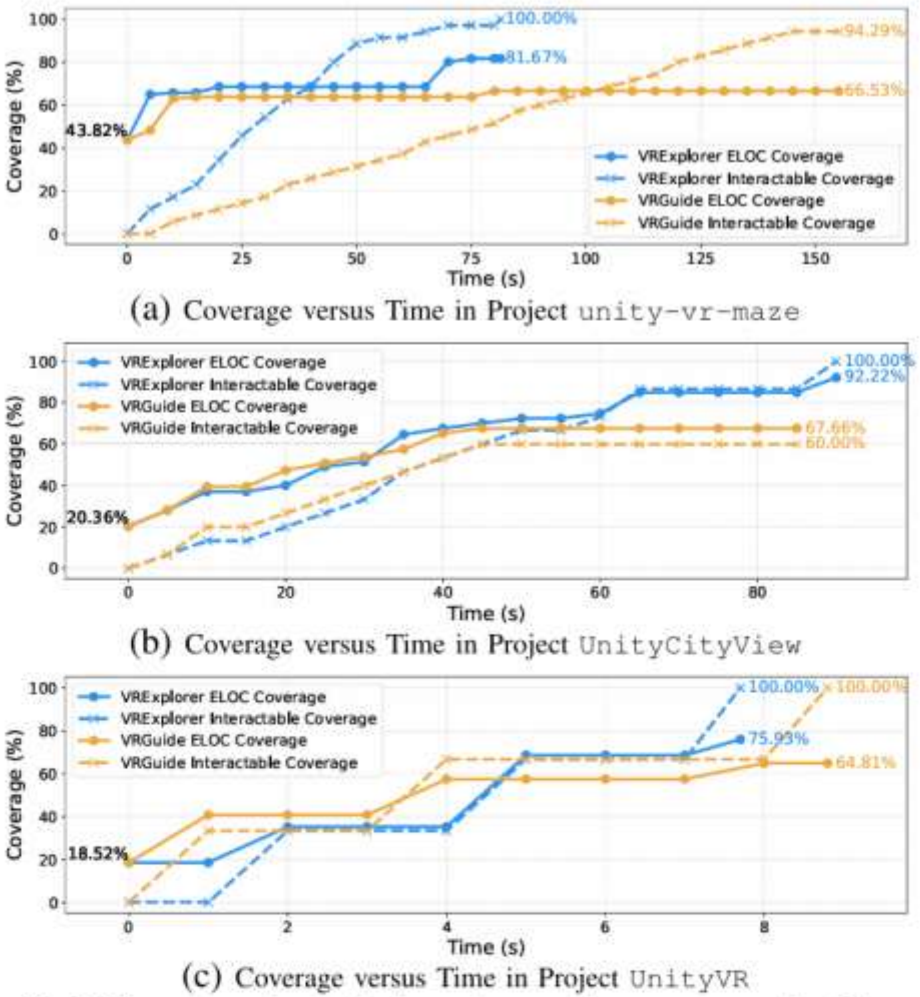
Evaluation of VRExplorer

Types of evaluated applications

- (i)Action and Shooter (VGuns)
- (ii)Simulation (VR-Basics, VR-Room, Unity VR)
- (iii) Adventure (unity-vr-maze, VR-Adventure)
- (iv) Puzzle (Edutainment-Escape-Room, EscapeGameVR)
- (v) Medical Care (Parkinson-VR)
- (vi) Strategy Board Game(VRChess)

Evaluation of VRExplorer

RQ1 (Performance vs SOTA)



(a) Coverage versus Time in Project `unity-vr-maze`

(b) Coverage versus Time in Project `UnityCityView`

(c) Coverage versus Time in Project `UnityVR`

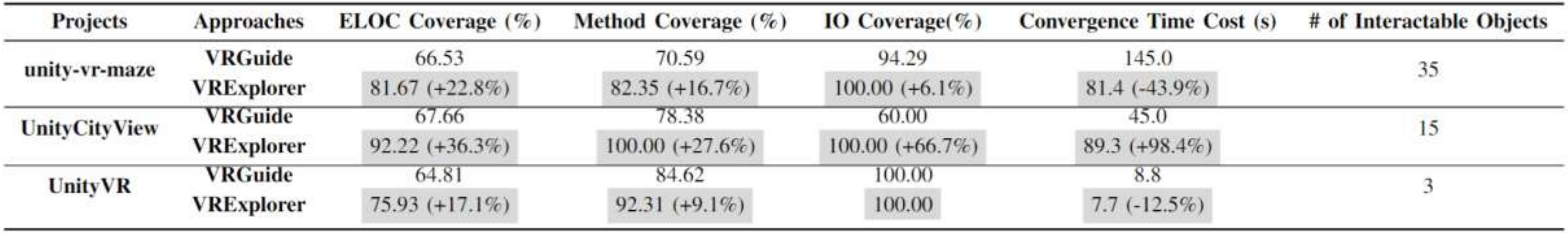
Fig. 8: EC versus time during the testing process in Group 1.

Experiment

■ RQ1 (Performance vs SOTA)

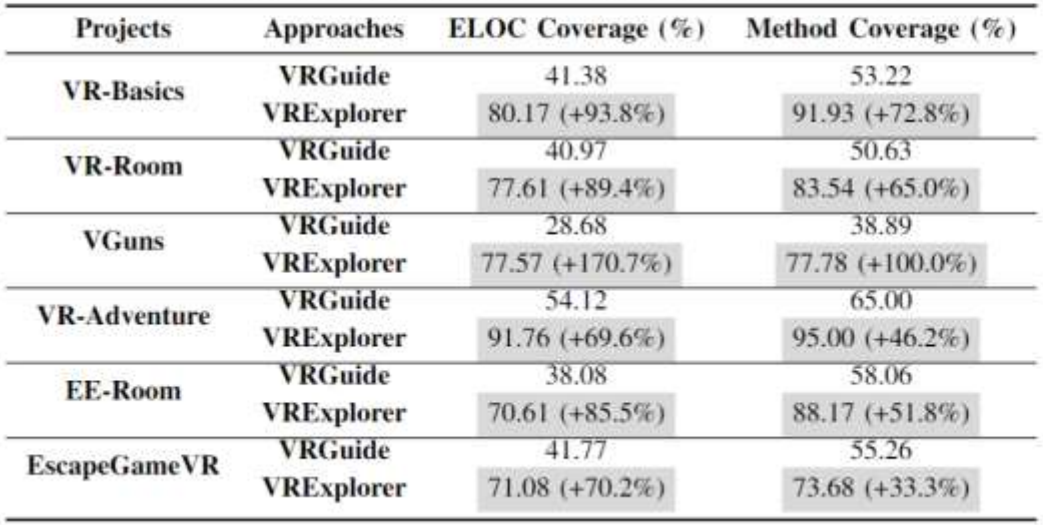
$$G_{AB}(M) = \frac{P_A(M) - P_B(M)}{P_B(M)} \times 100\%,$$

TABLE V: Results on Projects in Group 1



Projects	Approaches	ELOC Coverage (%)	Method Coverage (%)	IO Coverage(%)	Convergence Time Cost (s)	# of Interactable Objects
unity-vr-maze	VRGuide	66.53	70.59	94.29	145.0	35
	VRExplorer	81.67 (+22.8%)	82.35 (+16.7%)	100.00 (+6.1%)	81.4 (-43.9%)	
UnityCityView	VRGuide	67.66	78.38	60.00	45.0	15
	VRExplorer	92.22 (+36.3%)	100.00 (+27.6%)	100.00 (+66.7%)	89.3 (+98.4%)	
UnityVR	VRGuide	64.81	84.62	100.00	8.8	3
	VRExplorer	75.93 (+17.1%)	92.31 (+9.1%)	100.00	7.7 (-12.5%)	

TABLE VI: Results on Projects of Group 2



Projects	Approaches	ELOC Coverage (%)	Method Coverage (%)
VR-Basics	VRGuide	41.38	53.22
	VRExplorer	80.17 (+93.8%)	91.93 (+72.8%)
VR-Room	VRGuide	40.97	50.63
	VRExplorer	77.61 (+89.4%)	83.54 (+65.0%)
VGuns	VRGuide	28.68	38.89
	VRExplorer	77.57 (+170.7%)	77.78 (+100.0%)
VR-Adventure	VRGuide	54.12	65.00
	VRExplorer	91.76 (+69.6%)	95.00 (+46.2%)
EE-Room	VRGuide	38.08	58.06
	VRExplorer	70.61 (+85.5%)	88.17 (+51.8%)
EscapeGameVR	VRGuide	41.77	55.26
	VRExplorer	71.08 (+70.2%)	73.68 (+33.3%)

Answer to RQ1 (Performance): Faster and Higher Coverage

Experiment

- **RQ2: (Ablation Study)** How do different modules contribute to the performance of VRExplorer?

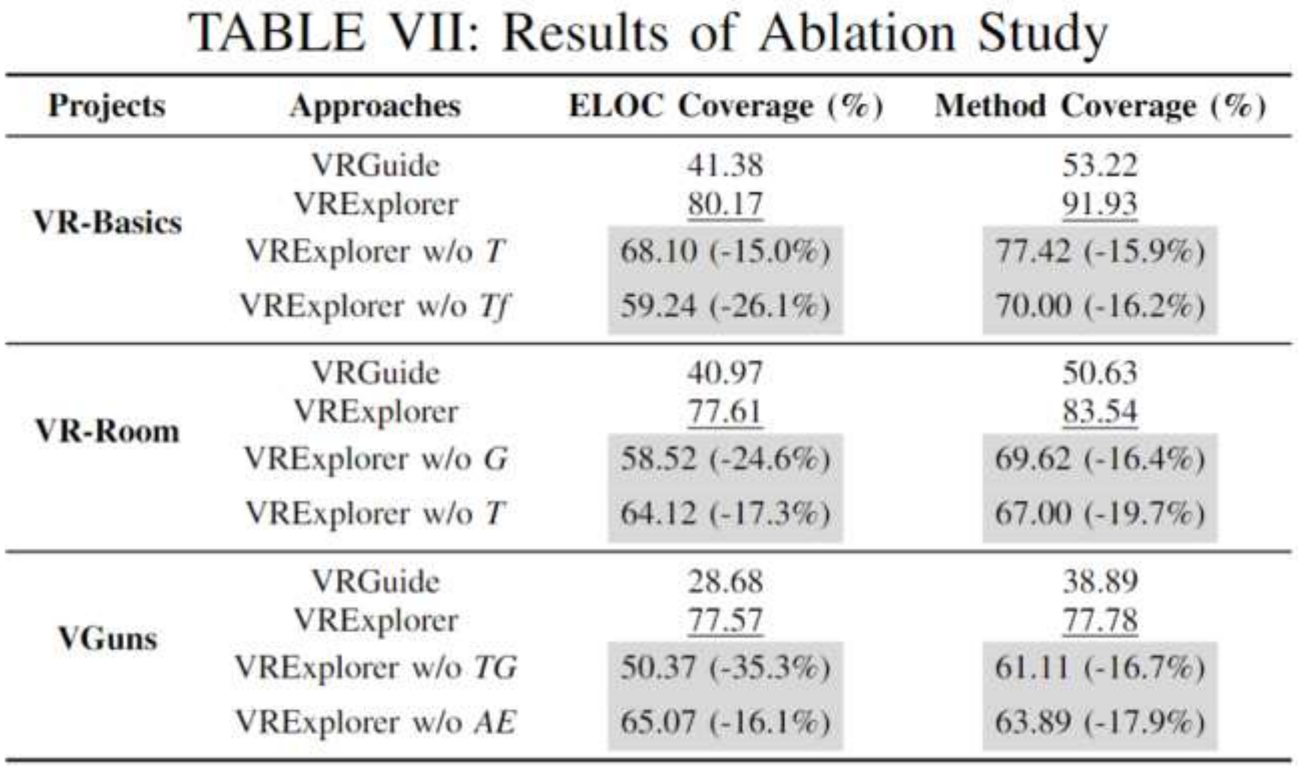


TABLE VII: Results of Ablation Study

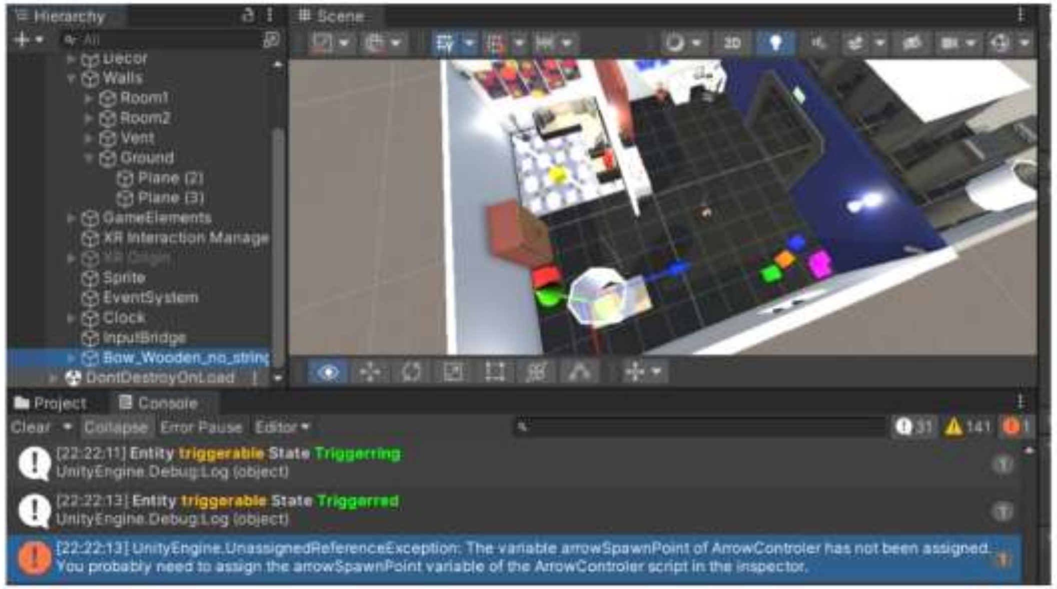
Projects	Approaches	ELOC Coverage (%)	Method Coverage (%)
VR-Basics	VRGuide	41.38	53.22
	VRExplorer	80.17	91.93
	VRExplorer w/o *T*	68.10 (-15.0%)	77.42 (-15.9%)
	VRExplorer w/o *Tf*	59.24 (-26.1%)	70.00 (-16.2%)
VR-Room	VRGuide	40.97	50.63
	VRExplorer	77.61	83.54
	VRExplorer w/o *G*	58.52 (-24.6%)	69.62 (-16.4%)
	VRExplorer w/o *T*	64.12 (-17.3%)	67.00 (-19.7%)
VGuns	VRGuide	28.68	38.89
	VRExplorer	77.57	77.78
	VRExplorer w/o *TG*	50.37 (-35.3%)	61.11 (-16.7%)
	VRExplorer w/o *AE*	65.07 (-16.1%)	63.89 (-17.9%)

Answer to RQ2: Interaction Modules Contribute Marvelously

Experiment

RQ3: Can VRExplorer detect real-world VR bugs?

- 2 functional in projects EscapeGameVR and UnityCityView
- 1 non-functional bug in the project EscapeGameVR.



```
[22:22:11] Entity triggerable State Triggerring
UnityEngine.Debug:Log (object)
[22:22:13] Entity triggerable State Triggerred
UnityEngine.Debug:Log (object)
[22:22:13] UnityEngine.UnassignedReferenceException: The variable arrowSpawnPoint of ArrowControler has not been assigned.
You probably need to assign the arrowSpawnPoint variable of the ArrowControler script in the inspector.
```

Fig. 7: A Detected Bug in EscapeGameVR

Answer to RQ3: capable of detecting complex VR bugs

Future Work

- **Future Work**
  - **LLM + VRExplorer**
    - Fully automated VR testing agent
    - Using LLM to generate testing actions

Acknowledgement

Acknowledgement

- This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFB2704100, the National Natural Science Foundation of China (No. 62032025), the Seed Funding for Collaborative Research Grants of HKBU (with Grant No. RC-SFCRG/23-24/R2/SCI/06), the Major Key Project of Peng Cheng Laboratory under Grant PCL2025AS07.

Contact Me

- **Contact Me** (Feel free to reach out)**:**
  - **Homepage**: https://tsingpig.github.io/
  - **Email**:
    - tsingpig@163.com
    - zhuzhy57@mail2.sysu.edu.cn
  - **PDF Url:** [PDF] VRExplorer (ASE '25)