Invisibility Cloak: Proactive Defense Against Visual Game Cheating

Chenxin Sun, Kai Ye, Liangcai Su, **Jiayi Zhang**, Chenxiong Qian
The University of Hong Kong

Research Background

The revenue of video game markets is expected to reach **$282.3 billion** in 2024 [1]

8.76% Until 2027

Counter-Strike 2

Valorant

Crossfire

[1] https://www.statista.com/outlook/dmo/digital-media/video-games/worldwide

Aimbots Are the Most Notorious Type Among All Cheats

- **Cheat Types**
  - **Aimbots** – Assist cheaters to auto-aim and auto-shoot
  - Wallhacks
  - Macro-Settings
  - Bug Exploits
  - Smurfing
  - Boosting
  - …

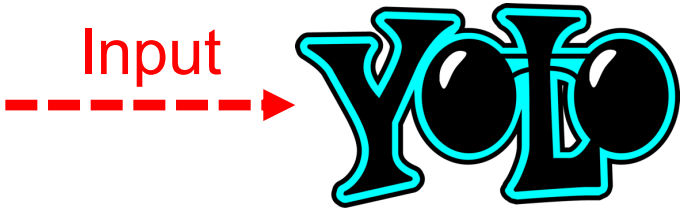
Aimbots' Categories

- **Aimbots**
  - Memory-Access
  - Visual

Memory-Access Aimbots

Visual Aimbots

Input



Windows APIs

Visual Aimbot is Trending

Guide to AI aimbot

Guide to making your own AI aimbot

Why?
I decided to make this guide because of countless people asking me for help with making their aimbot that uses AI instead of memory reading.

What?
I will show you how to make an AI aimbot using YOLOv4 and OpenCV library (train and code the program).

How?
https://memegenerator.net/img/instances/66639307.jpg

Process of making it:
The making of your aimbot I see as a two-step process:

- paste together the code that will run the AI and aim for you.
- create a dataset that you will use to train your AI to recognise whatever the fuck you want it to recognise, in most cases this will probably be the enemy player. - This is more time consuming

A few expressions:
weights / Neural network (NN) / AI / model / the stuff that actually does the magic

Making Aimbot:

So how do you make the aimbot?
I will go off of my previously posted cheats. YOLO Aim Augmentation v2.0, Valorant Cheat w Arduino and Yolov5 AI

How it works?

1. we take a screenshot
2. make a blob from it
3. pass the blob through the model
4. read the output of the model
5. convert the output into useful stuff
6. (Optional) draw the output
7. calculate mouse movement from the useful stuff
8. move the mouse
9. repeat as fast as possible

https://www.unknowncheats.me/forum/general-programming-and-reversing/485725-guide-ai-aimbot.html

Visual Aimbot is Trending

I tried to make a Valorant AI using computer vision

1.5M views

41K

I went down a rabbit-hole of trying to make a Python program that can play Valorant using computer vision and some radio shenanigans.

...more

https://www.youtube.com/watch?v=LXA7zXVz8A4

Visual Aimbot is Trending

Chinese police arrest 10 individuals related to VALORANT AI cheat provider

by Juandi | September 22, 2023

Ten people have be
cheat, with profits **US$4.1 million** **VALORANT**

https://valo2asia.com/chinese-police-arrest-10-individuals-related-to-valorant-ai-cheat-provider/

Challenges in Visual Aimbots

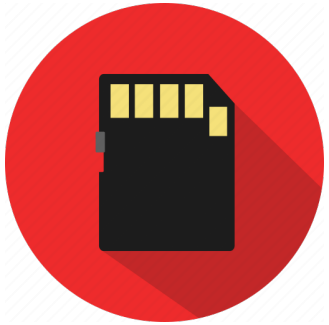
Game Screen

Normal Players

Challenges in Visual Aimbots

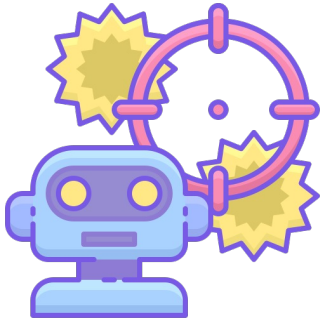
Game Screen

Normal Players



Memory Data

Extract



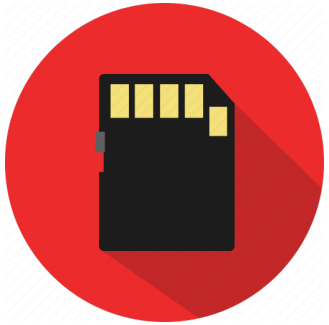
Memory-access Aimbots

Challenges in Visual Aimbots

Game Screen

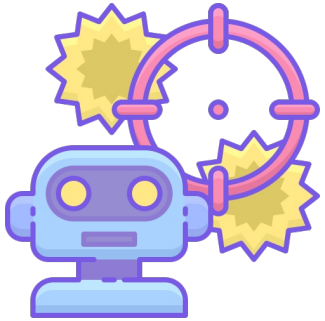
Normal Players

Traditional Anti-cheats (e.g.,VAC [1], EAC [2], BattleEye [3])



Memory Data

Prevented or Detected



Memory-access Aimbots

[1] Valve Anti-Cheat: https://help.steampowered.com/en/faqs/view/571A-97DA-70E9-FF74
[2] Easy Anti-Cheat: https://www.easy.ac
[3] BattleEye: https://www.battleye.com/

Challenges in Visual Aimbots

Game Screen

Normal Players

Visual Aimbots

Challenges in Visual Aimbots

Game Screen

Normal Players

Visual Aimbots

① **Undetectability**

AC

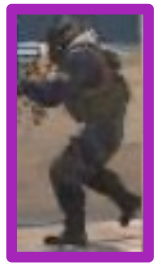
Traditional Anti-cheats (e.g.,VAC, EAC, BattleEye)

② **High Cheating Success Rate**

③ **Generalizability**

Our Solution: Invisibility Cloak

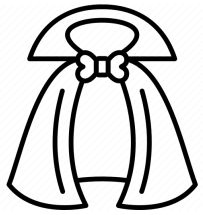
Game Screen



Normal Players

Visual Aimbots

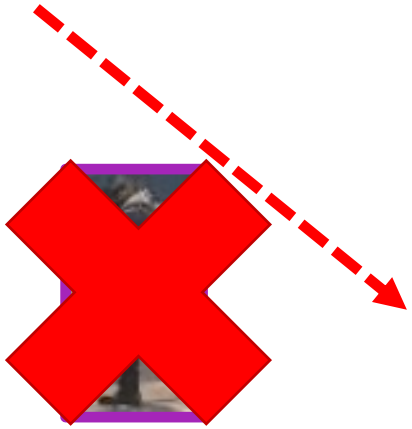
Our Solution: Invisibility Cloak



Cloak

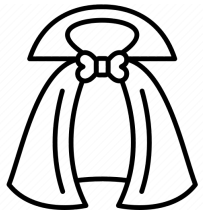
Game Screen

Normal Players



Visual Aimbots

Our Solution: Invisibility Cloak



Cloak

Game Screen

Normal Players

Visual Aimbots

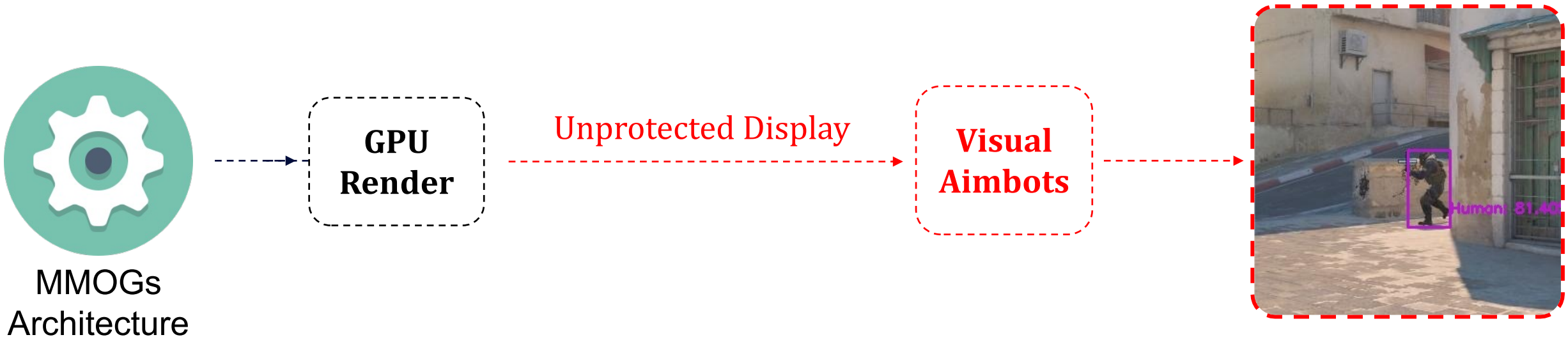
① Imperceptibility

② Real-Time Performance

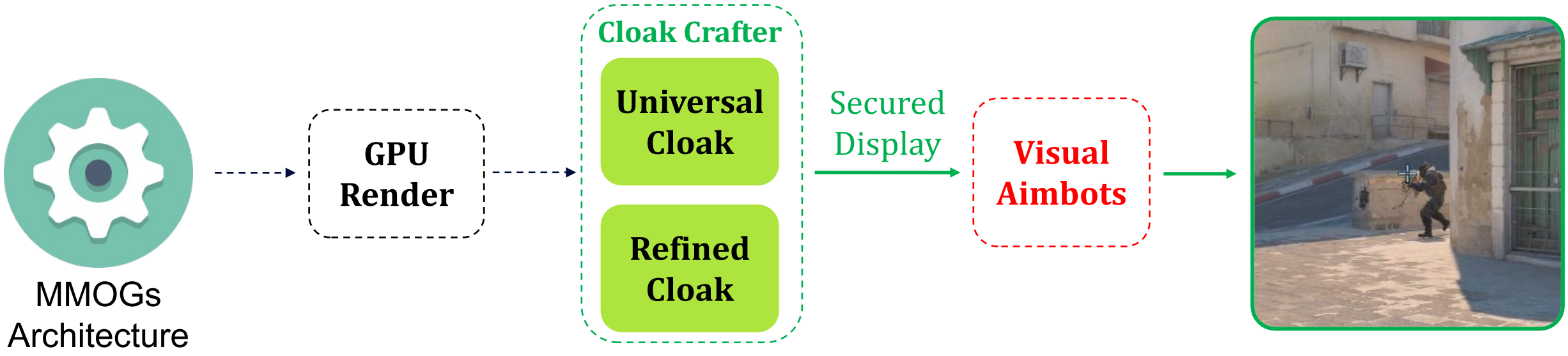
③ Transferability

④ Robustness

Overview of Invisibility Cloak

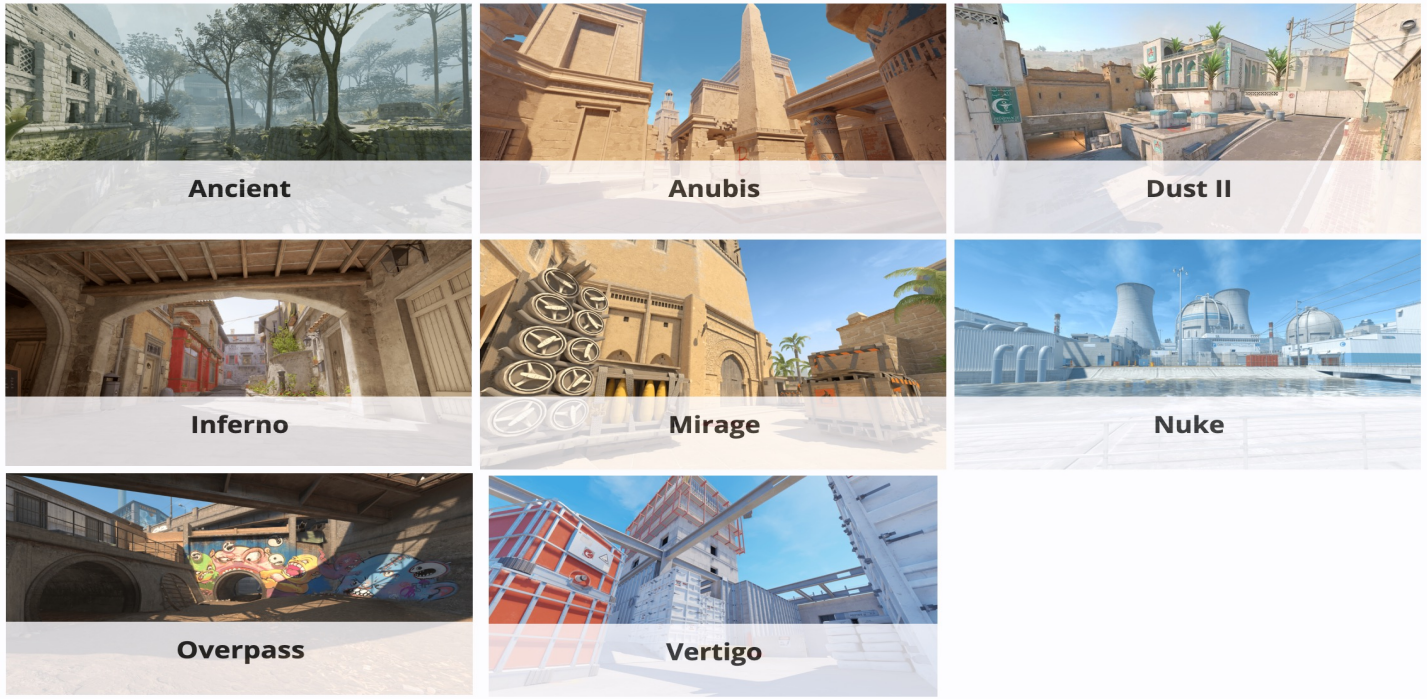


Overview of Invisibility Cloak



Universal Cloak

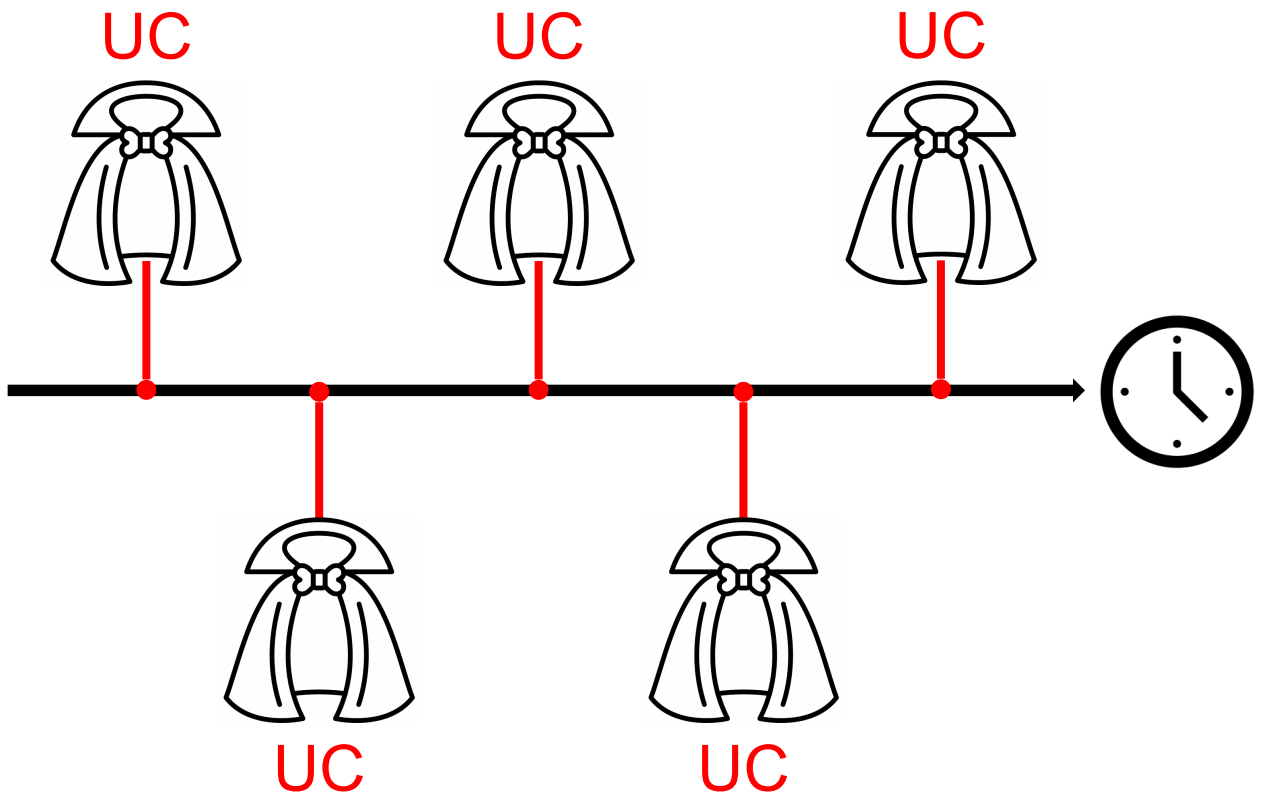
- Pre-crafted **offline** with various data and models
- Customized by scenarios and regularly updated



Cloak Refinement

- Refine the Universal Cloak online
- Frames are sampled **at intervals**

UC: Universal Cloak

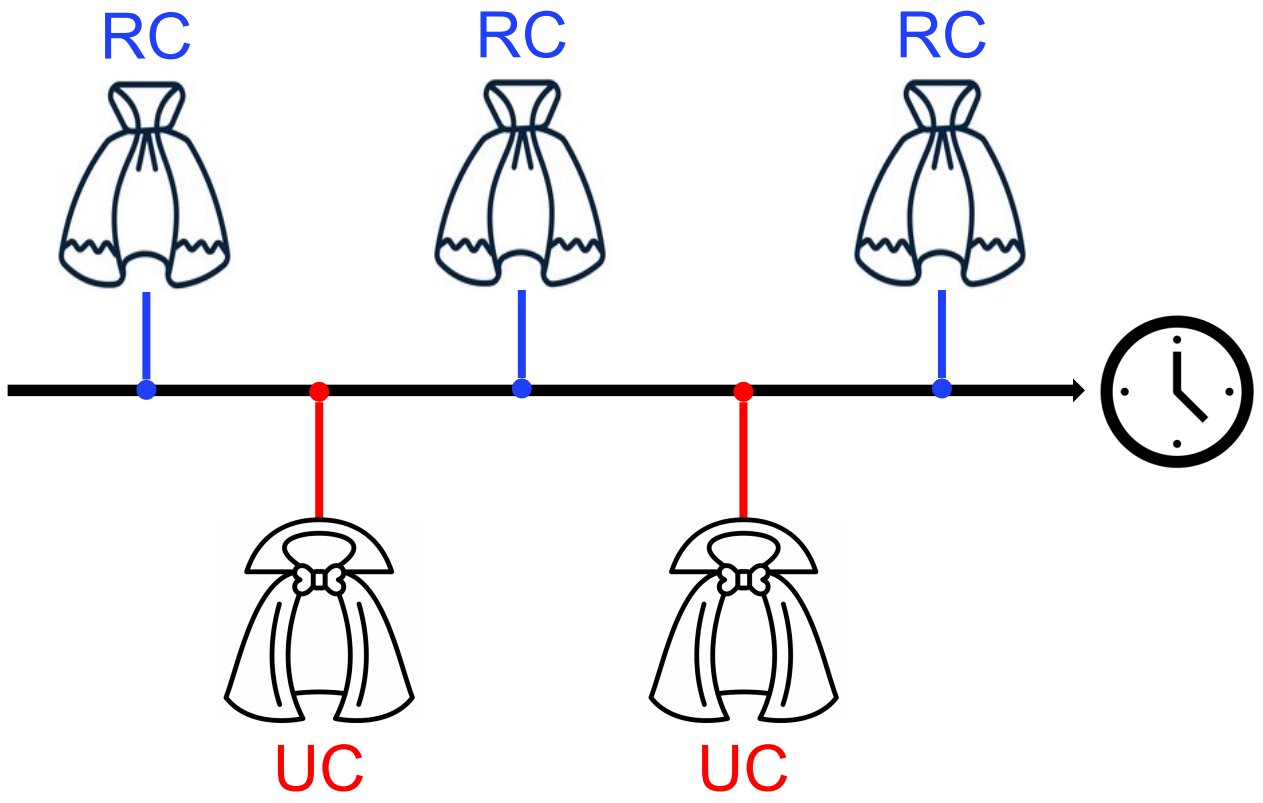


Cloak Refinement

- Refine the Universal Cloak online
- Frames are sampled **at intervals**

UC: Universal Cloak

RC: Refined Cloak



Cloak Generation Process

$$\arg\min_{\delta} \mathcal{S} = \mathcal{M}^{C}(x+\delta), s.t., \|\delta\|_{\infty} < \varepsilon,$$

Original →

Evaluation

RQ1: How **effective** is our approach in proactive defending against vision-based game cheating methods?

RQ5: How effective is our approach when **cheaters try to counter** our defense?

RQ6: How effective is our approach **in real-world** games at preventing aimbots while ensuring invisibility?

For more details, please refer to our paper!

Dataset

The dataset consists of **72,150** screenshots from real matchmaking gameplay in Counter-Strike 2 (CS2) and Crossfire (CF).

Conter-Strike 2

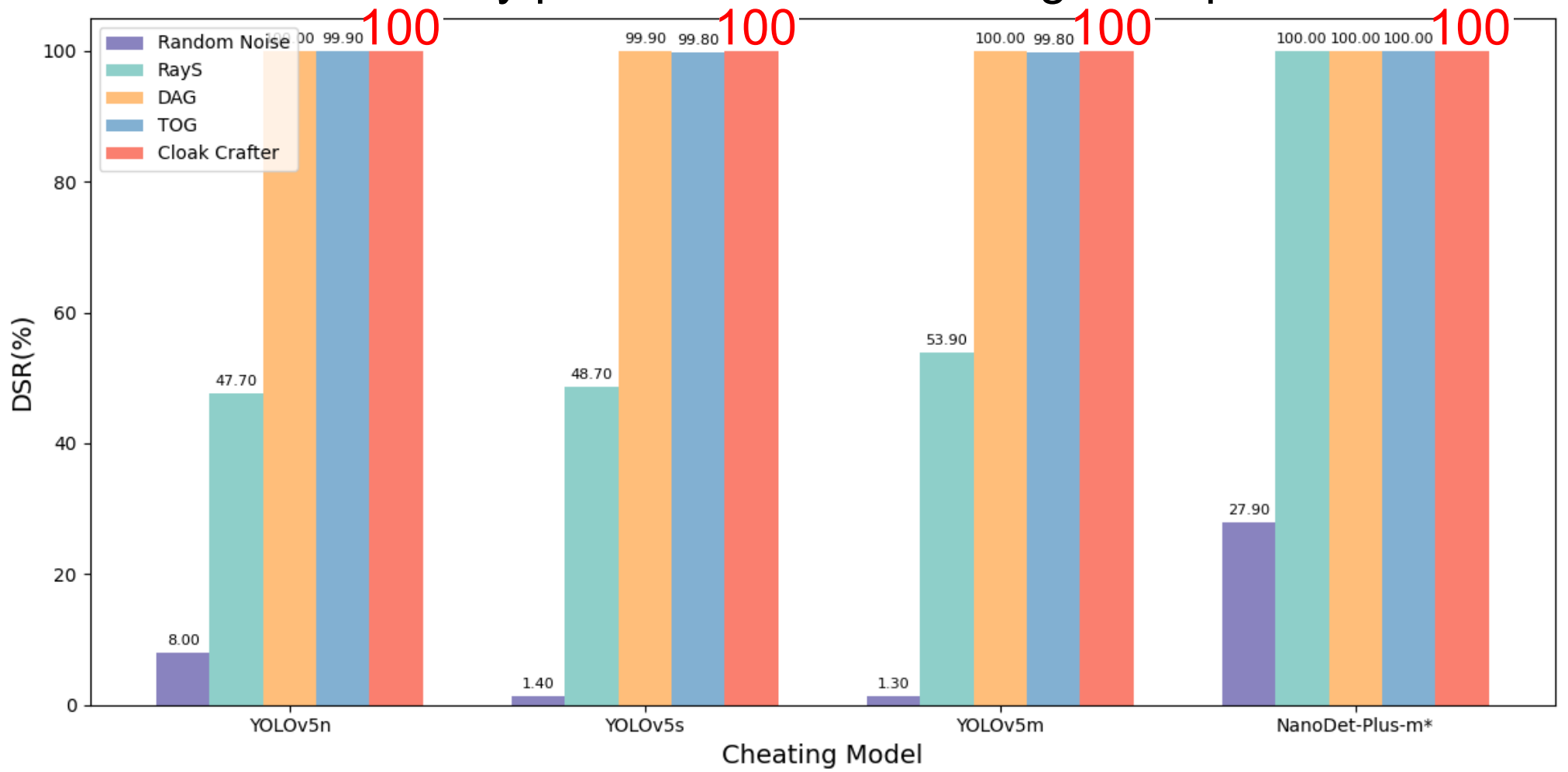
Crossfire

Dataset	Map	#frames[1]	#eng.[2]	#targets[3]	aTPF[4]
ACVC-CS2	dust2	10,500	350	13,877	1.322
	anubis	480	16	495	1.031
	mirage	570	19	612	1.074
	vertigo	1,140	38	1,184	1.039
	inferno	420	14	448	1.067
	nuke	960	32	993	1.034
	desert_atrium	150	5	150	1.000
	repository	330	11	331	1.003
	desert_town	270	9	302	1.119
	Total	14,820	494	18,392	1.241
ACVC-CF	coconut_island	16,110	537	16,404	1.018
	aquarium	10,260	342	11,587	1.129
	ship	15,840	528	17,130	1.081
	pyramid	3,360	112	3,557	1.059
	training_ground	2,730	91	3,375	1.236
	stable	7,650	255	7,804	1.020
	dust	1,380	46	1,502	1.088
	Total	57,330	1,911	61,359	1.070

[1]**#frames**: the number of frames;
[2]**#eng.**: the number of engagements (30 frames window per engagement);
[3]**#target**: the number of detected targets (i.e., YOLOv8n detected as human);
[4]**aTPF**: the averaged number of targets per frame.

Defensive Performance: Effectiveness

- Defense Success Rate (DSR) (↑): The proportion of instances where the method successfully prevents visual cheating attempts.



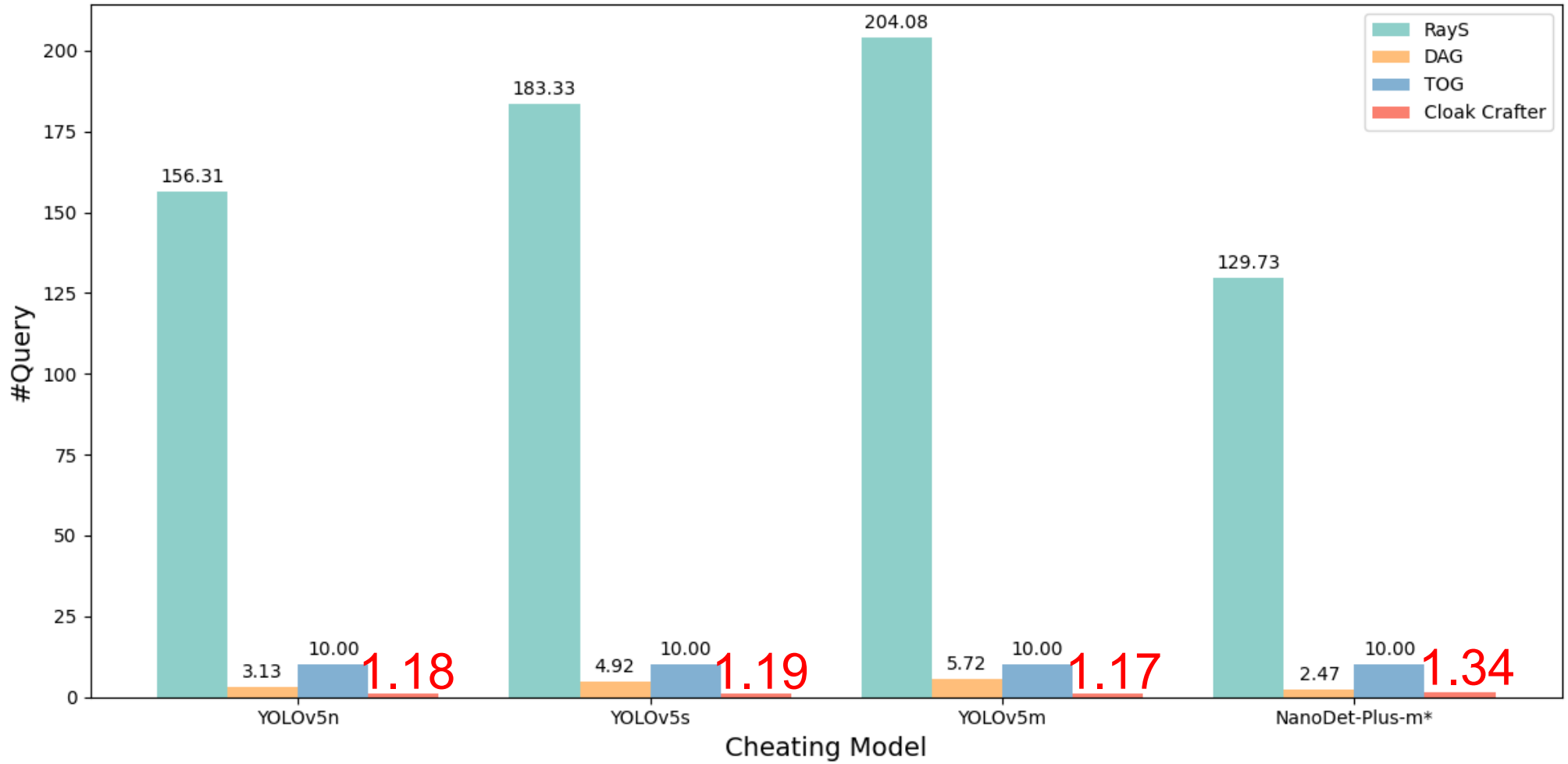
*: Anchor-free model

Defensive Performance: Efficiency

- Frames Per Second (FPS) (↑): The number of frames that Cloak Crafter can process per second.

Defensive Performance: Efficiency

- Query (↓): The number of queries to the proxy model required to generate a perturbation.



Adaptive Defense Evaluation

- Performance under adaptive attack using **adversarial training** with fine-tuned YOLOv5n as a local proxy model.

Dataset	Cheating Model	ε = 8/255			ε = 16/255		
		DSR(%)	FPS	#Query	DSR(%)	FPS	#Query
AVCA-CS2	YOLOv5n^{+}	100.00	31.30	2.26	100.00	86.05	1.05
	YOLOv5n$^{+}_{AT}$	81.70	31.30	2.26	86.00	93.49	1.05

$^{+}$: Models that have been fine-tuned;

AT: Models that have been adversarial trained.

100% ⟶ 81.70%

100% ⟶ 86%

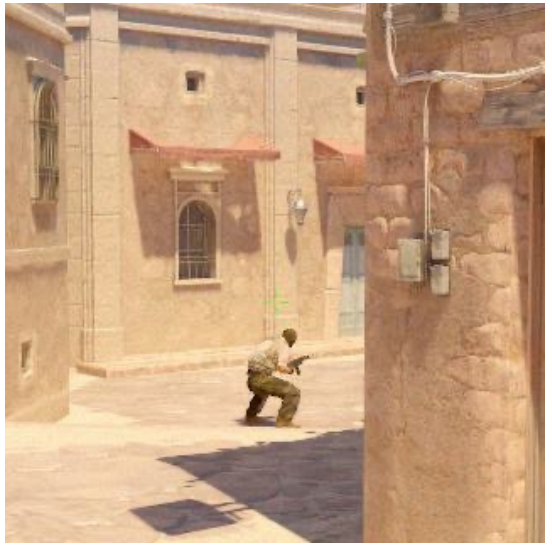
User Study

122 participants, 80% have shooting games experience.

Q: Please choose the most realistic gameplay image.

- Binary Choices (Total Acc: 47.87%)

Indistinguishability

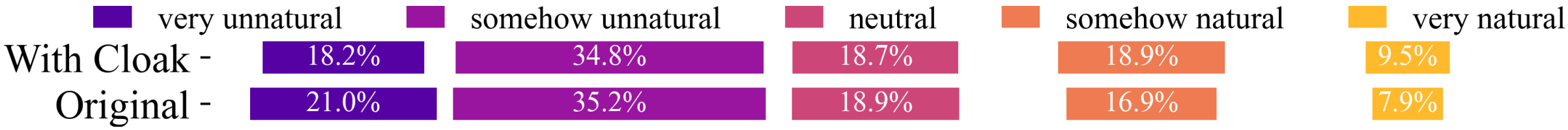


A

B

User Study

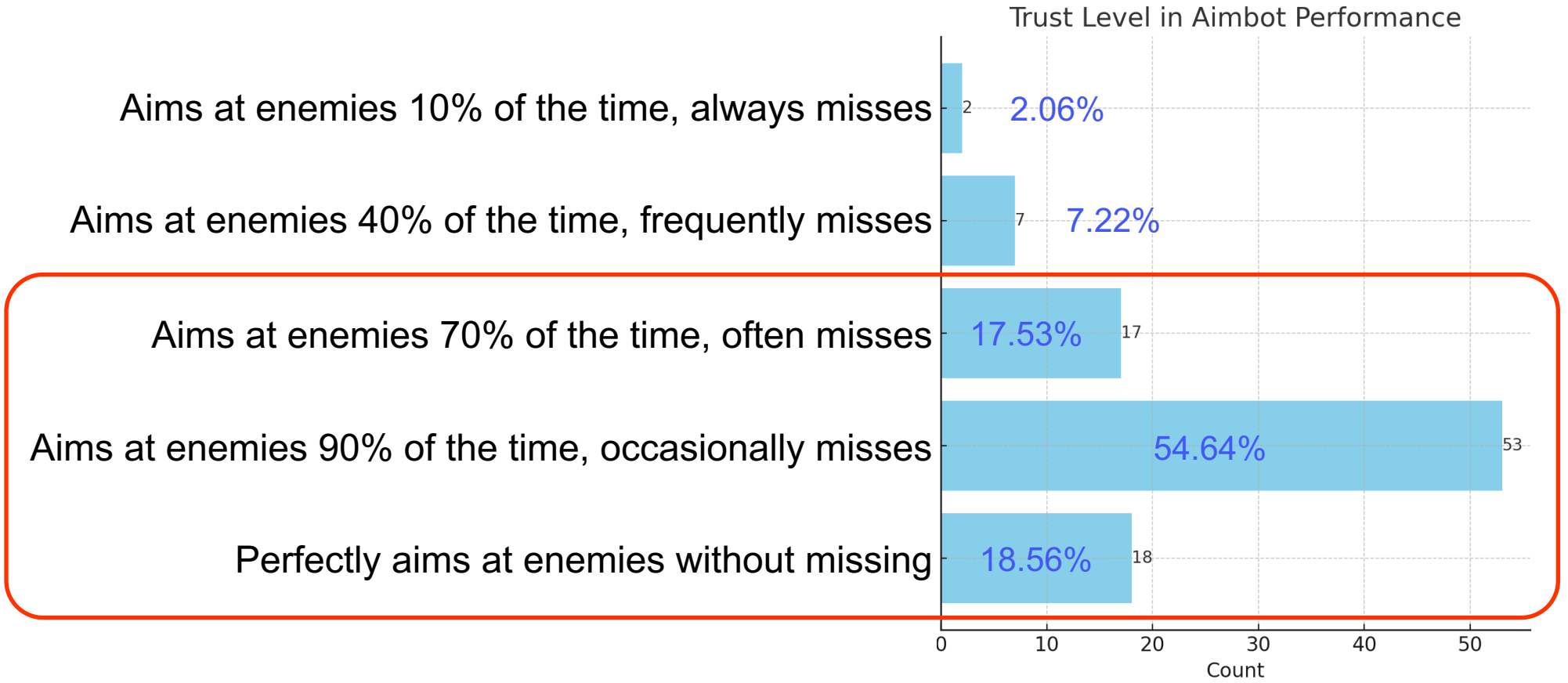
Q: Please choose the naturalness of a randomly selected game screenshot demo from five options.



Naturalness

User Study

Q: What level of performance would make you trust and continue using an aimbot (automatically aims and shoots)?



90.73%

Real-World Effect Verification

Real-World Effect Verification

Game	Duration(sec)	w/o Invisibility Cloak		w/ Invisibility Cloak	
		#Detection	#Auto-aim	#Detection	#Auto-aim
CS2	2,370.61	3,860	3,549	**5**	**5**
CF	1,765.69	2,394	2,197	**2**	**2**

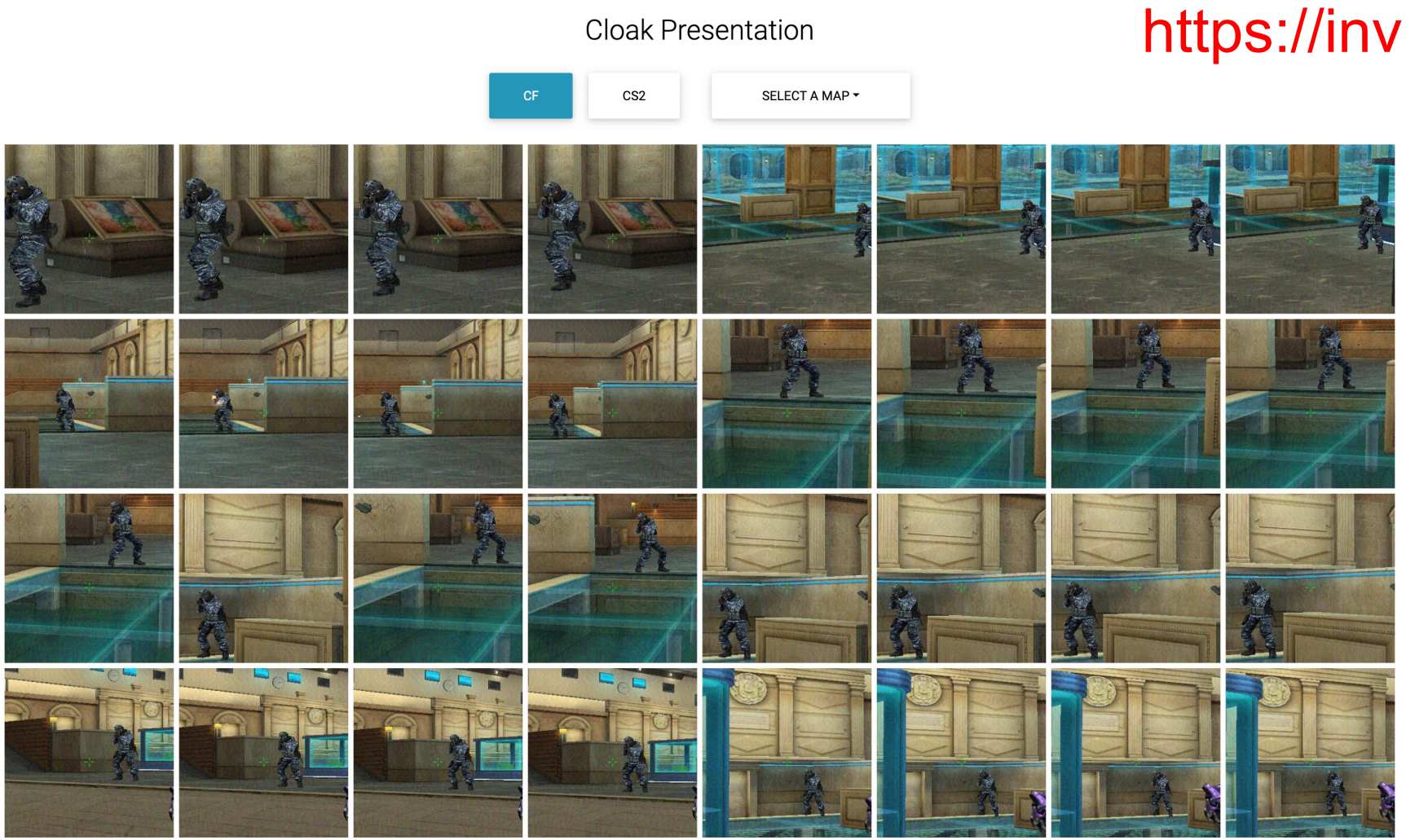
[1]**#Detection**: the number of successful detections by aimbot;
[2]**#Auto-aim**: the number of successful auto-aiming by aimbot.

99.9% Defence Success Rate

3549 ⇢ 5

2197 ⇢ 2

Examples and Demos



https://invisCloak.github.io/

Examples and Demos

Comparison Demo for CS2

https://invisicloak.github.io/

SELECT A SCENARIO

We showcase three perspectives: a honest player's view (left), a cheater using visual aimbots (center), and the cheater's view with our Invisibility Cloak deployed, blocking the aimbots (right).

Questions?

https://inviscloak.github.io/

Q: Resolutions?

Dataset	Resolution	Cheating Model	DSR(%)	FPS	#Query
AVCA-CS2	224×224	YOLOv5n	99.90	40.00	2.28
		YOLOv5m	52.20	21.15	2.75
		YOLOv5x	46.40	16.91	2.54
		Average	66.17	26.02	2.52
	320×320*	YOLOv5n	100.00	77.94	1.51
		YOLOv5m	75.80	45.74	1.57
		YOLOv5x	74.40	44.31	1.44
		Average	83.40	55.99	1.51
	416×416	YOLOv5n	99.90	47.15	1.73
		YOLOv5m	73.60	32.05	2.12
		YOLOv5x	68.50	19.15	2.27
		Average	80.67	32.78	2.04
AVCA-CF	224×224	YOLOv5n	99.90	41.65	2.30
		YOLOv5m	47.70	30.25	2.61
		YOLOv5x	41.40	18.46	2.38
		Average	63.00	30.12	2.43
	320×320*	YOLOv5n	100.00	71.97	1.55
		YOLOv5m	74.20	58.61	1.51
		YOLOv5x	73.10	40.43	1.47
		Average	82.43	57.00	1.51
	416×416	YOLOv5n	100.00	50.96	1.98
		YOLOv5m	61.50	32.96	2.93
		YOLOv5x	54.80	16.91	2.24
		Average	72.10	33.61	2.38

*: The baseline resolution.

Q: Prior Works

- **BotScreen (USENIX 23’)**
  - Detection method (functional post-cheat)
- **BlackMirror (CCS 20’)**
  - Prevents Memory and Rendering Processes
  - Targeting Wallhacks