Proceedings on Privacy Enhancing Technologies ; 2021 (3):433–452

Abdul Wajid, Nasir Kamal, Muhammad Sharjeel, Raaez Muhammad Sheikh, Huzaifah Bin Wasim, Muhammad Hashir Ali, Wajahat Hussain, Syed Taha Ali*, and Latif Anjum

A First Look at Private Communications in Video Games using Visual Features

Abstract: Internet privacy is threatened by expanding use of automated mass surveillance and censorship techniques. In this paper, we investigate the feasibility of using video games and virtual environments to evade automated detection, namely by manipulating elements in the game environment to compose and share text with other users. This technique exploits the fact that text spotting in the wild is a challenging problem in computer vision. To test our hypothesis, we compile a novel dataset of text generated in popular video games and analyze it using state-of-the-art text spotting tools. Detection rates are negligible in most cases. Retraining these classifiers specifically for game environments leads to dramatic improvements in some cases (ranging from 6% to 65% in most instances) but overall effectiveness is limited: the costs and benefits of retraining vary significantly for different games, this strategy does not generalize, and, interestingly, users can still evade detection using novel configurations and arbitrary-shaped text. Communicating in this way yields very low bitrates (0.3-1.1 bits/s) which is suited for very short messages, and applications such as microblogging and bootstrapping off-game communications (dialing). This technique does not require technical sophistication and runs easily on existing games infrastructure without modification. We also discuss potential strategies to address efficiency, bandwidth, and security constraints of video game environments. To the best of our knowledge, this is the first such exploration of video games and virtual environments from a computer vision perspective.

Keywords: video-games, virtual environments, automated surveillance, censorship circumvention

DOI 10.2478/popets-2021-0055
Received 2020-11-30; revised 2021-03-15; accepted 2021-03-16.

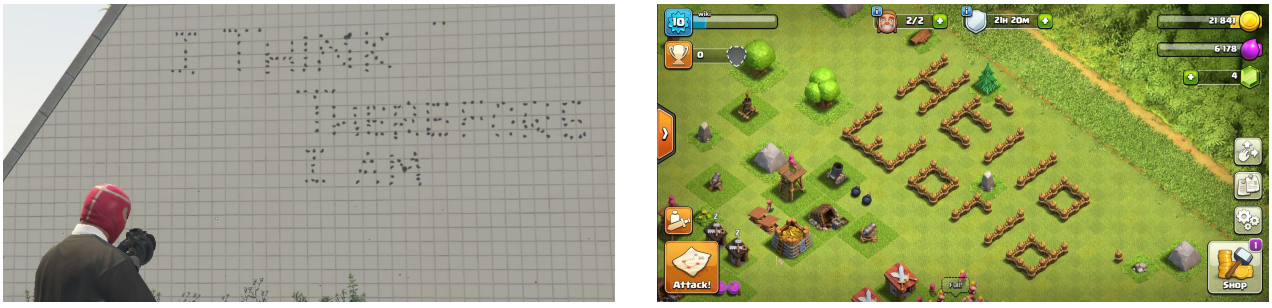
1 Introduction

Over the last decade, the Internet has emerged as a critical enabler for journalism, political organization and grass-roots activism. This was most evident during the Arab Spring, and recently in France, where citizens leveraged social media to organize large-scale protests [28]. Today major political figures actively maintain Twitter accounts, protests and demonstrations are routinely organized using social media, and online grass-roots campaigns can influence real world change [89].

Governments and organizations are consequently tightening their control on this medium. In 2019 Freedom House marked the ninth consecutive year of decline in Internet freedom [27]. The Egyptian government recently filtered 34,000 Internet domains to counter a political opposition campaign [57]. Internet connectivity in Belarus has suffered frequent large-scale disruptions after controversial elections recently [87]. Turkey has a history of blocking Facebook, Twitter, Wikipedia, and WhatsApp for political purposes [61]. Pakistan recently banned YouTube for three years on charges of hosting blasphemous content [91]. China's Great Firewall is the most extensive effort in the world to monitor and restrict cyberspace [52].

This has led to significant uptake of anonymity tools and encrypted messaging applications, and a contentious arms race between users and censors. Some governments react by blocking these services en masse [4]. Various governments are also legislating bulk data collection and mass surveillance [45], Big Brother laws [66] [86], and advocating government-controlled backdoors in communications platforms [78].

Abdul Wajid: National University of Sciences & Technology (NUST), Pakistan. E-mail: awajid.msee17seecs@seecs.edu.pk
Nasir Kamal: NUST, E-mail: 14mseenkamal@seecs.edu.pk
Muhammad Sharjeel: NUST,
E-mail: msharjeel.bee18seecs@seecs.edu.pk
Raaez Muhammad Sheikh: NUST,
E-mail: rsheikh.bee18seecs@seecs.edu.pk
Huzaifah Bin Wasim: NUST,
E-mail: hwasim.bee18seecs@seecs.edu.pk
Muhammad Hashir Ali: NUST,
E-mail: mali.bee18seecs@seecs.edu.pk
Wajahat Hussain: NUST,
E-mail: wajahat.hussain@seecs.edu.pk
***Corresponding Author: Syed Taha Ali:** NUST,
E-mail: taha.ali@seecs.edu.pk
Latif Anjum: NUST, E-mail: latif.anjum@seecs.edu.pk



(a) 'I think therefore I am' (Grand Theft Auto V)

(b) 'Hello World' (Clash of Clans)

Fig. 1. Messages generated in video game environments

In this paper we explore the use of video games to preserve user privacy and counter censorship. Prior research has used games to host covert channels. These approaches include concealing messages in the network traffic of games [82], and encoding data in player moves [22] or in the motion of avatars [98].

In contrast, we propose a very different, simple, and intuitive approach: two parties, Alice and Bob, interact with the game environment to visually communicate text. In a first-person shooter game, Alice uses a gun to spell out messages on walls (Fig. 1(a)[1]). In a strategy game, she forms text by constructing buildings or moving units accordingly (Fig. 1(b)). We focus specifically on the following questions:

- *Is it feasible for parties to use visual features in games as a privacy enhancing technology?*
- *Can these communications be detected using state-of-the-art automated tools?*
- *What communication modes and security properties does this medium allow?*

The key insight here is that these communications do not exist at the code level or in transmitted bits and therefore evade detection by traffic analysis and deep packet scanning methods. These messages exist in the virtual geometry of a scene as visual artefacts, which can be easily deciphered by a human subject. However, automated detection of these messages lies in the domain of computer vision, and, whereas, considerable progress has been made regarding text recognition in certain formats (e.g. documents), text spotting in the wild remains an open and challenging problem, due to environment complexity and context [68] [41] [1].

In this paper, we make the following contributions:

1. We demonstrate that leading text spotting tools used out of the box fail to detect text created in various game environments. For this evaluation, we generate a novel dataset[2] of over 3000 in-game text samples from three popular video games and scan them using four state-of-the-art text spotting tools.
2. We investigate the potential of retraining these tools specifically to detect in-game text. Our results show significant improvement in some cases. However, the costs and benefits of retraining varies and does not generalize to games in different genres. We also find that monitoring players' in-game activities via keystrokes and controls logs has limited effectiveness in detecting in-game communication.
3. Our experiments indicate, interestingly, that even with retrained tools, users can evade detection using novel text configurations. The flexibility afforded by certain game environments allows creation of arbitrary-shaped text which exploits fundamental design limitations of text spotting tools.
4. We demonstrate various ways to use this communication technique, adapted from real-world examples, and we suggest strategies to address efficiency, bandwidth, and security limitations of this medium.

This approach has considerable advantages: video games are a popular pastime, they are available on a wide range of devices, and they feature considerable diversity in format, gameplay, and user experience.

Our particular contribution differs from prior work in important ways: to the best of our knowledge, we are the first to explore the video games medium from a computer vision perspective and exploit the problem of text-spotting in the wild. Communicating in this way is intuitive and simpler. In contrast, most systems in the literature require extensive client-side modifications or technical sophistication on the part of users.

1 We recommend that images in this paper be viewed on screen in colour and with appropriate magnification.

2 This dataset is an independent contribution and is accessible at https://github.com/seecswajid/gametextpets

However, there are various disadvantages. Manually composing characters in a virtual environment is cumbersome, laborious, and not conducive to interactive conversation. As we demonstrate with examples, this approach is more suited for short messages, microblogging, and sharing tweet-length posts. Short messages can also be used to bootstrap out-of-game communications channels, by sharing URLs, Tor bridge addresses, VPNs, proxies, and real-world rendezvous points.

Moreover, our results lead to questions which require further investigation. For instance, in-game text may evade automated tools but is easily detected by human adversaries. Defending against these may require significantly more coordination and technical skills. Furthermore, adversaries may develop larger datasets and text spotting tools for specific game environments and various text configurations. These are resource-intensive tasks, but not impossible for very powerful adversaries. These aspects require further research.

Our paper is organized as follows: §.2 presents background material on games and text spotting. We detail a communications scenario in §.3. §.4 describes our dataset, experiments, and results. In §.5, we consider various strategies to enhance communications. We discuss related work in §.6 and conclude in §.7.

2 Background

Here we describe the video games ecosystem followed by a primer on text spotting and a demonstration of the tools we use in our experiments.

2.1 Video Games

Video games have emerged as a popular and highly lucrative industry over the last few decades. Leading video game titles now routinely contend with blockbuster Hollywood movies in terms of hype and sales. A market research study estimates the worth of the global video games market at $159.3 billion and the number of gamers worldwide at 2.7 billion [90].

The gaming ecosystem includes PCs, dedicated consoles, and smartphones with a multitude of diverse game genres, including action, adventure, sports, strategy, and roleplay. A recent survey notes that 64% of US households own a device on which they play video games, and 60% of Americans play video games on a daily basis [14]. Google claims that every month viewers watch more than 144 billion minutes worth of gameplay videos and livestreams on YouTube, where some gaming channels generate view counts comparable to those of Hollywood celebrities and musicians [65].

Online multiplayer games are an exceedingly popular genre that attracts players from different walks of life, engaging over long periods of time, giving rise to close-knit online communities and rich subcultures [76].

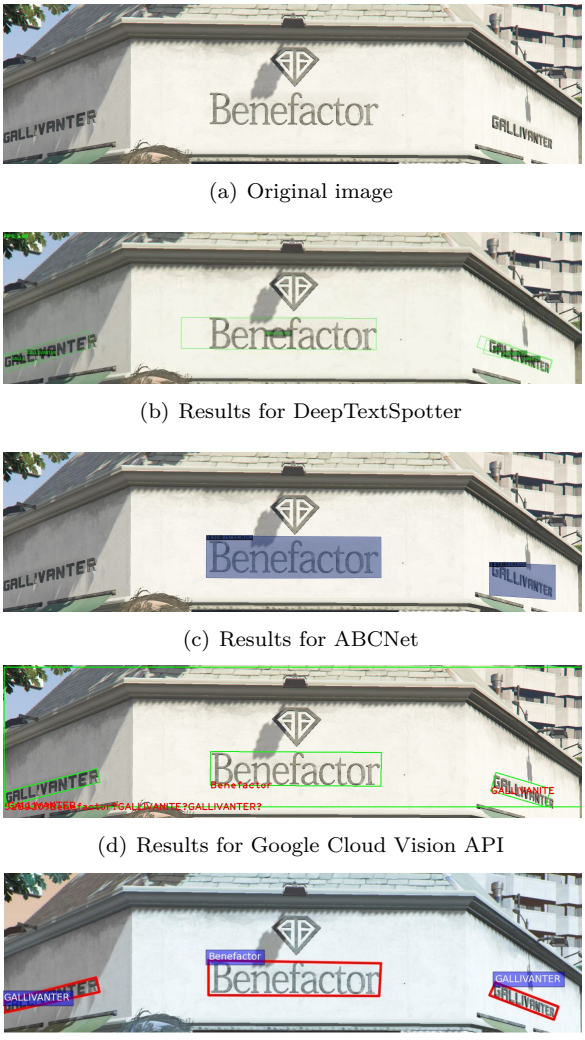
Political and activist trends have also started to surface in these environments. Political campaigns have crossed over into games and virtual environments [88] [64]. In 2017, players in Minecraft and Second Life customized avatars to protest anti-immigration policies [10]. Players in Nintendo's Animal Crossing recently set up memorials and organized virtual protests coinciding with real life demonstrations [69]. Nintendo has now formally requested organizations to refrain from bringing politics into Animal Crossing [58].

2.2 A Primer on Text Spotting

Text embedded in natural images and videos provides automated systems with vital contextual and semantic information, and stands to enable a variety of beneficial applications, including video annotation, forensics, assistance for visually impaired people, and navigation for robot vehicles. However, whereas automated text recognition in scanned documents has witnessed significant advances in recent years, text spotting in scene images remains an open problem.

The reason is that whereas text in documents is usually uniform and ordered, text in natural scenes suffers from various constraints. Text may include unusual fonts. The scene may contain background objects, textures, and clutter that are visually similar to the text itself. The image may be of poor quality or include distortion or noise. Viewing angles and lighting conditions also complicate detection. These factors make text spotting a formidable challenge [67] [68] [41].

Chen et al. first proposed breaking the text spotting problem into two subproblems: text detection (or localization) and text recognition (or classification) [7]. Several works subsequently focused on these problems individually. Text detection solutions (e.g. [95] [21] [79] [40] [46]) typically identify and isolate text into a 'candidate bounding box'. Text recognition solutions (e.g. [60] [30] [71]) then attempt to classify the words. Some recent solutions connect text detection and recognition modules into an end-to-end pipeline to yield a complete text spotting solution (e.g. [6] [41] [42] [68]).



(a) Original image

(b) Results for DeepTextSpotter

(c) Results for ABCNet

(d) Results for Google Cloud Vision API

(e) Results for Azure Cloud Vision

Fig. 2. Output for textspotting tools

We next introduce four text spotting tools consisting of prominent solutions from the research literature and popular commercial solutions. We choose these detectors for their cutting-edge performance which is widely acknowledged in the computer vision community. We demonstrate their results using native text from Grand Theft Auto V (GTAV) in Fig. 2.

DeepTextSpotter, presented by Bušta et al. in 2017, exploits the complementarity between the detection and recognition processes to enhance efficiency [6]. The system is trained on 1229 words in real scenes and ∼9 million in synthetic scenes. DeepTextSpotter outperformed competing solutions on the ICDAR 2013 and 2015 datasets. Word detection in a scene is marked by bounding boxes, as shown in Fig. 2(b). The word 'Benefactor' and the instances of 'GALLIVANTER' placed at oblique angles on either side are detected successfully.

ABCNet, proposed by Liu et al. in 2020, improves upon precision of previous systems in more challenging scenarios where text may be non-uniform and arbitrarily shaped [42]. The system is trained on a dataset consisting of 150 thousand synthesized images, 15 thousand images extracted from COCO-Text, and 7 thousand from the ICDAR-MLT dataset. ABCNet gives state-of-the-art results on benchmark datasets Total-Text and CTW1500. Detected words are highlighted in bounding boxes as shown in Fig. 2(c). Notably, it misses the instance of 'GALLIVANTER' on the left.

Google launched **Google Cloud Vision (GCV)** API in 2017, a wide-ranging commercial cloud-based computer vision suite with extensive pretrained models [9] to classify images efficiently into thousands of categories, detect individual objects and faces, and detect and classify text from a variety of languages. The GCV API is proprietary and there are little published details on the underlying technology. GCV successfully detects 'GALLIVANTER' in bounding boxes as shown in Fig. 2(d), and suggests spelling corrections.

Our final tool, **Azure Cloud Vision (ACV)** from Microsoft, launched in 2020, is part of the Azure cloud computing platform. ACV identifies a wide variety of features, content, and text (including hand-written text) from images and documents. As with GCV, there is little published information on the underlying algorithms. ACV also successfully detects and highlights the text in bounding boxes as depicted in Fig. 2(e).

3 Writing on the Wall

Here we describe how Alice and Bob can communicate in a multiplayer video game, like Grand Theft Auto V. To play, Alice logs on to the public GTA Online server or dedicated servers run by other entities. Inside a session, Alice can play with players of her choice or random players in a public session. Gameplay consists of fights with gangs, robbing armoured trucks, and purchasing items for the character. The maps are typically big and highly detailed, affording considerable opportunity for Alice and Bob to rendezvous.

Alice and Bob locate a blank surface, like a wall or a large object, and spell out their messages on this surface using bullets. They can even leave messages for each other to view later using an in-game location and make public posts which other players can see.

We assume there is an adversary, Eve, who relies on automated surveillance to detect in-game commu-

(a) Graffiti (Istanbul, 2014) (b) Replicated message (GTAV)

Fig. 3. Sharing alternate DNS server information

Fig. 4. Replicating WikiLeaks tweet with server IP address

nications. Our threat model derives from the standard warden and prisoner scenario encountered in the literature on steganography and covert channels in games [82] [22]. The warden, Eve, in our case, seeks to detect and/or censor any communications apart from legitimate game information. Eve can take on many roles: she may be an individual hacker or the administrator of the network Alice or Bob use. She may be an employee of a game company or a government or intelligence agency.

We assume Eve undertakes bulk data collection, including network traces of game traffic, players' voice and text logs, gameplay footage, and may even deploy surveillance bots in the game. All collected data is analyzed using automated tools. We consider the specific case of human adversaries in game environments in §.5. We also assume Alice and Bob can conduct a one-time secure exchange to communicate information, such as user ID, avatar, rendezvous location and time, etc., to coordinate in-game activities.

There already exists a subculture of intricate art and graffiti created using weapons within GTAV and other games [20]. This approach is also extensible: games within a genre share common structure and elements, such as weapons, environment, and gameplay, enabling us to generalize message composition techniques. If one game is therefore unavailable or censored, Alice and Bob can easily migrate to a similar one.

However, generating in-game text is laborious, time-consuming, and therefore more suited for very short messages, such as microblogging or bootstraping off-net communications by advertising rendezvous points, URLs, Tor bridges and proxies. We suggest that this communication mode is comparable to real-world graffiti and street art in terms of message content and effort involved in creating and policing it. As with graffiti, users can also broadcast political statements, share hashtags, and organize meetings and political activity.

We find pertinent use-cases in real-world examples. For instance, when the Turkish government put a DNS block on Twitter prior to the general elections of 2014 [26], users posted graffiti with IP addresses for alternative Google DNS servers, depicted in Fig. 3, along with our replication in GTAV. In Fig.4, we replicate a Wikileaks tweet from 2016 with proxy information after Turkey blocked the Wikileaks website.

4 Experiments and Discussion

We now describe our dataset and experiments to investigate the effectiveness of leading text spotting tools in detecting in-game communications.

4.1 The GameText Dataset

We have modeled our dataset after the Street View Text (SVT) dataset [83]. SVT is one of the earliest and most well-known datasets in the computer vision community for text spotting and benchmarking purposes. The dataset, compiled in 2010, consists of outdoor images harvested from Google Street View with annotated text of signage on buildings and walls. There are 350 images with 904 labeled words (of which 571 are unique), with a total of 2047 characters. Fig. 5 depicts the distribution of labelled characters in the images. We replicate the complete SVT word list in three video games.

We chose games that are exceedingly popular with an active online multiplayer community, are rich in detail, and with variation in genre. Details are as follows:

1. **Grand Theft Auto V (GTAV)** is an action-adventure game and one of the highest rated video game titles in the world. Upon release, it earned over $6 billion, breaking global sales records, including those of top Hollywood movies, to become the most profitable entertainment property in the world [48]. Over 33.8 million players have logged on to play GTAV. Players in GTAV complete missions to progress through the game. A player may freely roam open maps, run, jump, swim, use vehicles to navigate cities, and fight enemies with weapons and explosives.

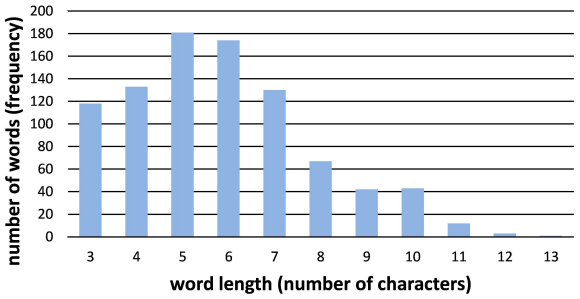


Fig. 5. Distribution of word length in SVT dataset

2. **Call of Duty IV: Modern Warfare (CoD4)** is a first-person shooter released in 2007. It is one of the best selling games of the last decade, it has proved highly influential, with two sequels released thus far. CoD4 still retains considerable popularity: in 2020, the number of monthly online players has peaked at 62.7 million [47]. In the game, players undertake missions in a realistic live combat setting in war zones with access to a variety of modern military weapons.

3. **Minecraft** is a 3D sandbox environment released in 2009 on various platforms. It is one of the most influential and best-selling games to date, selling over 200 million copies. Minecraft currently attracts 126 million monthly active users, and it dominates game viewership stats on YouTube [85]. The game has no specific goals, but offers players an infinite terrain with objects and materials such as wood, water, and stones which may be used to construct items of varying complexity. Various spin-off games have been developed using Minecraft, including educational applications.

In each game, we generated text samples by making marks on surfaces or rearranging objects within the environment. In GTAV and CoD4, we use bullets on walls and other surfaces. Minecraft offers more possibilities with use of different materials like wood, water, etc. Samples of SVT images and our corresponding images are presented in Fig. 6. We refer to this created text as *in-game text* to distinguish it from *native text*, i.e. text occurring naturally within the game, i.e. text on street signs and billboards, as in Fig. 2(a).

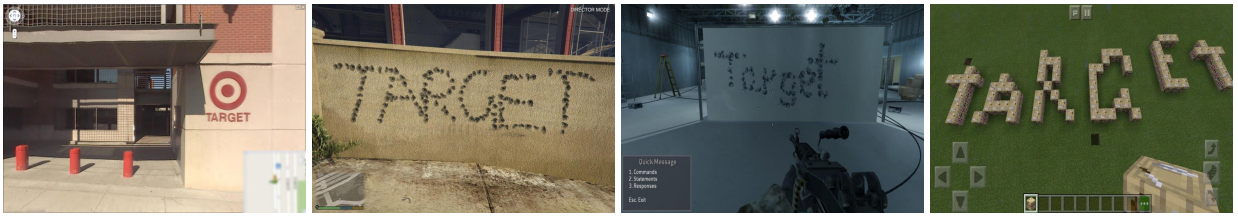
Dataset specifications are listed in Table. 1. The SVT images are mostly low resolution and exhibit high variability of lighting, perspective, and diversity of fonts. In contrast, our images are high-resolution and fonts are uniform within games. We captured each image from three angles (listed as (x3) in Table. 1): a front-facing view, and from the left and right sides to study the effect of perspective and curvature. We also generated images on plain and textured surfaces and under dark and bright lighting conditions.

For control purposes, we collected 50 images featuring native text from GTAV. We also generated 50 images of curved and arbitrary-shaped text, as typically found in logos and monograms.

This dataset was generated and labeled by 6 undergrad students over the course of two months. In contrast to other popular datasets (e.g. Streetview [97], MS COCO [81]) where the effort lies in labeling images, the bulk of our time and effort went into locating appropriate surfaces and generating the text itself within the game. The entire exercise took approximately 250 hours spread out over a 2 month period.

To the best of our knowledge, this is the first public image dataset focusing specifically on this type of adversarial text, and, as such, is an independent contribution also of interest to researchers outside the security domain, notably in the computer vision community[3].

3 The dataset is accessible at https://github.com/seecswajid/gametextpets



(a) SVT - TARGET (b) GTAV - TARGET (c) CoD4 - Target (d) Minecraft - TARGET

(e) SVT - MURPHY (f) GTAV - MURPHY (g) CoD4 - MURPHY (h) Minecraft - Murphy

Fig. 6. Adversarial fonts generated in games using bullets and constructions

Game	Category	Text/Font Type	Image Count	Word Count	Character Count
GTAV	Action-adventure	Bullets	350(x3)	904	2047
CoD4	First Person Shooter	Bullets	350(x3)	904	2047
Minecraft	3D Sandbox	3D Object Constructions	350(x3)	904	2047
GTAV	Action-adventure	Native text	50	250	1323
GTAV	Action-adventure	Arbitrary-shaped text	50	50	283

Table 1. Details of GameText dataset

4.2 Experiments and Results

In the first stage, we scan our dataset using the text spotting tools and report our results. Our objective is to empirically measure out-of-the-box performance of these tools in different environments. Alongside this exercise, we also run these tools on the original SVT dataset and on the set of 50 native text images in GTAV to give us a point of comparison. Our metric is text spotting accuracy, i.e. the percentage of images for each individual game in the set where the tools correctly report the in-game text.

Table 2 presents results. DeepTextSpotter and ABCNet fail completely in detecting in-game text in all games. Azure Cloud Vision performs marginally better: it detects less than 10 images for each game. Google Cloud Vision performs an order of magnitude better on CoD4 but peaks at 5.98% for Minecraft.

These results contrast strongly with those for GTAV native text and for the SVT dataset in the last two columns. All four tools successfully detect over 65% of native text. Azure Cloud Vision performs best at 85.6%. Detection is also very high for the SVT dataset with a minimum of 74.2% for DeepTextSpotter and 91.8% for Azure Cloud Vision. Overall, the commercial tools perform significantly better than the research solutions.

To explain this marked divergence in performance between user-generated and native text, we draw on insights from the computer vision domain. Our text spotting tools, as documented in §. 2 are originally trained on millions of images of scenes with naturally occurring (or native) text which usually consists of continuous strokes, as depicted in Fig. 7 (Natural Font). Discontinuous text is typically encountered in LED displays (LED Font).

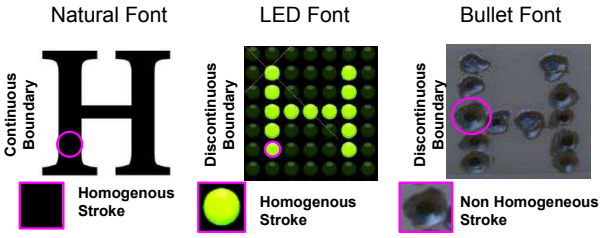


Fig. 7. Comparison of Natural, LED and Bullet Font. The Bullet Font is both discontinuous and non-homogeneous.

The challenges of detecting this type of text have been highlighted in the literature [32] [80]. Since each character in the sample is composed of discontinuous segments, it is critical that all segments be detected before the character can be correctly classified. Moreover, the discontinuity of the segments adds to the complexity of the classification.

A second property of native text is that strokes are usually homogeneous, i.e. character edges appear on the boundaries of strokes as depicted. Adding clutter within the region of the stroke disrupts this homogeneity [15]. Our in-game text samples violate both properties of native text (e.g. Bullet Font in Fig. 7), which, we theorize, makes it difficult for current text spotting tools to intelligently group these segments together into characters.

These results have interesting implications from a privacy perspective: we are able to create a kind of *adversarial text* using the tools available within the game itself. We emphasize here that our notion of adversarial text differs significantly from its usage in the literature. Standard adversarial attacks on deep learning systems typically work by adding small perturbations to images at the pixel level which deceive detectors but do not significantly impact the perception of a human viewer

Text Spotting Tools	GTAV Images: (350x3)	CoD4 (350x3)	Minecraft (350x3)	GTAV (Native Text) (350x3)	SVT Dataset (350)
DeepTextSpotter	0%	0%	0%	66.4%	74.2%
ABCNet	0%	0%	0%	74.8%	74.3%
Google Cloud Vision	0.2%	2.7%	5.98%	78%	86.5%
Azure Cloud Vision	0.44%	0.33%	0.48%	85.6%	91.8%

Table 2. Accuracy of text spotting tools on GameText dataset

[75] [17] [93]. In our case, however, the constraints of the game environment itself lead to creation of text which defeats leading text spotting tools out of the box.

This approach has advantages. Standard adversarial attacks require a specialized understanding of deep learning methods and a mechanism to undertake fine-grained modification of images. Game users and laymen typically do not have this specialist knowledge or the tools or even the necessary level of code access and skills to modify popular game environments. Using visual features to spell out text however is intuitive and requires little technical knowledge or skills.

4.3 Retraining Tools for In-game Text

We now investigate the impact of retraining these tools specifically using in-game text samples. The creators of DeepTextSpotter and ABCNet have released open-source end-to-end pipelines of their solution, enabling us to retrain these using our dataset.

We retrain the tools individually for every game. We isolate 300 common images from all three games as the test set and use the rest of the images as the respective training set. The objective is to construct a test set with identical text samples for every game, where each text sample is captured in two different states for three specific properties: with bright and dark lighting in the scene, against plain and textured backgrounds, and a frontal and perspective angle. This allows us to estimate the improvement in performance due to retraining these tools with regards to these particular properties.

We retrained DeepTextSpotter using Google Colab. It took approximately 6 hours per game when allocated an NVIDIA k80 GPU. We retrained ABCNet using a Core I-9 machine with an NVIDIA RTX 2080 Ti (11GB) GPU and 64 GB RAM. The process took approximately 3 hours per game. Results are presented in Table 3.

Accuracy improves for both tools, but is significantly greater for ABCNet. We speculate that this is due to fundamental differences in design. DeepTextSpotter has a conventional text spotting architecture designed to detect lateral text in rectangular bounding boxes [6]. ABCNet has a more modern design with custom layers for novel functionality, including the ability to accurately localize, align, and classify instances of oriented or curved text [42]. This is better suited to the irregular alignment of text in our dataset (e.g. Fig. 6).

Text Spotting Tools	GTAV	CoD4	Minecraft
DeepTextSpotter	6.67%	6.33%	0.67%
ABCNet	64.67%	42.33%	17.33%

Table 3. Detection of in-game text after retraining tools

Fig. 8 presents accuracy for various features. We note that the dramatic difference in results of both tools persists for all three features. Moreover, the observed improvement for Minecraft across both tools is considerably less than that for GTAV and CoD4. We reason that this may be due to the freedom and flexibility in open world environments. In GTAV and CoD4, our surfaces are limited and text is always rendered using bullets. In Minecraft, however, there is a rich variety of materials, surfaces, and textures. Prior research suggests that this can result in high dataset diversity, i.e. there is high variability of appearances and viewpoints and therefore less repetition in training images as compared to the other games [99]. In this case, it is possible that significantly more training data may be needed to attain an accuracy comparable to the other games.

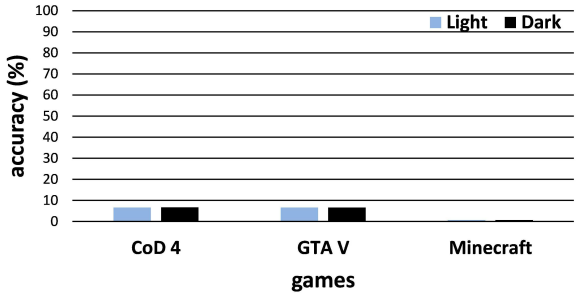
Lighting has little impact on accuracy. This is expected as the underlying scene structure does not significantly change. Prior work demonstrates that enhancements in image contrast provide light invariance, enabling comparable performance for scenes with different lighting conditions [37], as typically observed during the day/night cycle [50] and seasonal changes [56].

Performance drops considerably for textured surfaces. This is contrary to findings in the literature. Generally machine learning techniques have made considerable progress in processing additional as well as adversarial clutter in scenes (as evidenced in notable results on solving visual CAPTCHAs [53] [18] [94]).

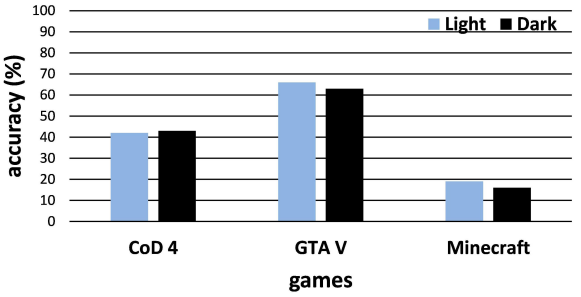
However, we note that different properties (lighting, surface type and perspective) are not mutually exclusive here, i.e. a scene with textured background may also feature text viewed from an oblique perspective. To further distinguish between features, we analysed only the frontal images for a textured-plain analysis. In this case, we observe in Fig. 8 that ABCNet's performance declined by 16% (plain 83%, textured 67%) for GTAV, and only 5% (plain 57%, textured 52%) for CoD4.

Not surprisingly, the biggest difference in performance correlates with perspective. ABCNet performance declined by 20% (frontal 76%, perspective 56%) for GTAV, and 18% (frontal 54%, perspective 36%) for CoD4. Perspective is a known challenge in text spotting as it may change the underlying structure of the scene, i.e. patterns and image gradients, resulting in decreased accuracy [31] [29] [72].

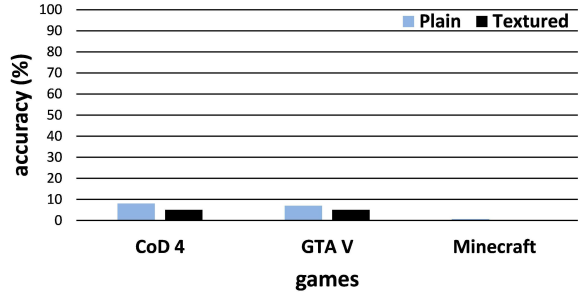
We also observe that generalization is poor, especially across genres. Tools trained on either GTAV or CoD4 show 10-20% accuracy when tested on the



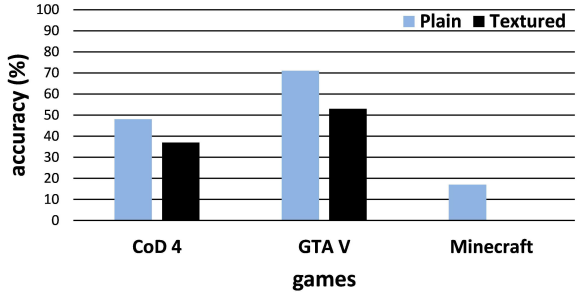
(a) DeepTextSpotter - light vs. dark scenes



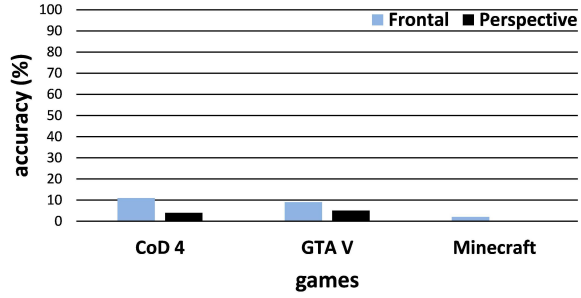
(b) ABCNet - light vs. dark scenes



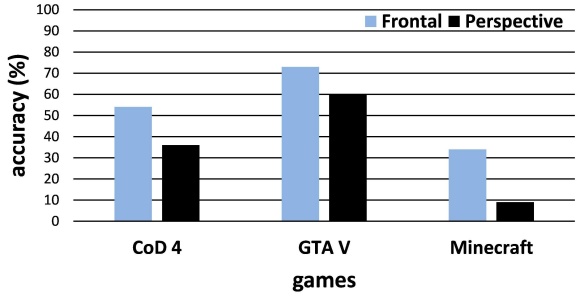
(c) DeepTextSpotter - plain vs. textured backgrounds



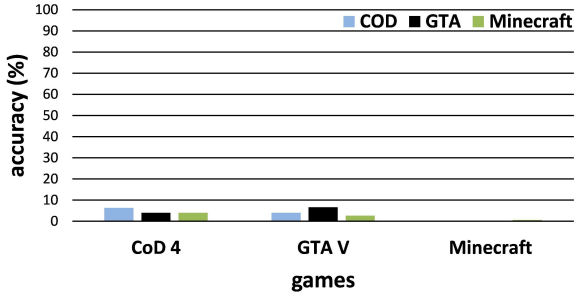
(d) ABCNet - plain vs. textured backgrounds



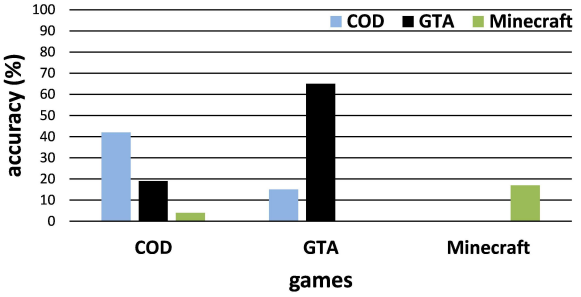
(e) DeepTextSpotter - frontal vs. perspective angle



(f) ABCNet - frontal vs. perspective angle



(g) DeepTextSpotter - cross-domain detection



(h) ABCNet - cross-domain detection

Fig. 8. Detection results after retraining DeepTextSpotter and ABCNet

other game and no improvement on Minecraft. Likwise, retraining on Minecraft yields negligible improvement when tested on the other games. This is likely due to the similarities between GTAV and CoD4 and the considerably different visual themes, textures, and structures encountered in Minecraft.

We would note here that these results are binary comparisons and not an ablation study. There is significant overlap in the sets for all three properties of lighting, texture, and perspective. A significantly larger dataset is required to weight the impact of each attribute in an exclusive manner. These results should therefore be considered early findings that are indica-

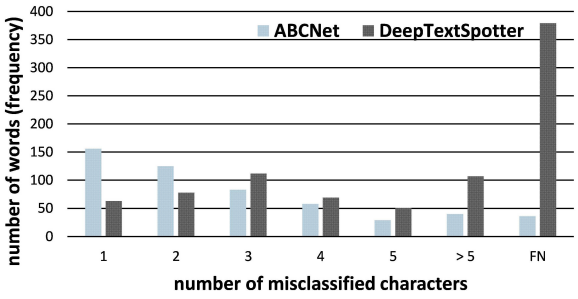


Fig. 9. Results for character misclassification and false negatives

(a) Synthetic sample (b) Sample in GTAV

Fig. 10. Examples of arbitrary-shaped text

tive of general trends that fit with our expectations from surveying the literature.

Fig. 9 breaks down the accuracy results of our retrained tools in terms of number of characters misclassified in a given word. We observe that ABCNet reliably spots text with minor transcription errors, i.e. the majority of failures are due to errors in classifying 1-2 characters. ABCNet also has a low rate of false negatives (FN in the graph), i.e. those words for which no character is detected at all. False negatives dominate in the case of DeepTextSpotter (almost ten times greater). We have already discussed possible reasons for the comparatively poor performance of this tool earlier.

To further evaluate the reliability of ABCNet, we tested the retrained tool on game scenes without text. We collected 150 images from the Internet containing random bullet marks for GTAV and CoD4 and structures in Minecraft. This resulted in a large number of false positives (47.33%, 71 out of 150), i.e. the tool identified text where there wasn't any.

It appears that retraining these tools on small datasets improves recall and recognition but overall reliability is still poor. Experiments with much larger datasets are needed to properly quantify this effect.

4.4 Arbitrary-shaped Text

In recent years, text spotting research focused on detection of curved and arbitrary-shaped text [44] [43] [42]. Progress, however, is limited to very simple curves. This is for two main reasons: first, spotting arbitrary-shaped text entails significantly more effort than lateral text because the bounding boxes are larger and there is potentially greater overlap with other objects or clutter in the scene. Second, current datasets contain few images of curved text. Specialized datasets have been released recently to address this gap (e.g. Total Text Dataset [8] and [96]), and these consist primarily of simple curves with one change in direction (i.e. single bends).

For this reason, we find that leading text spotting tools struggle when confronted with text written in more complicated shapes. Fig. 10 includes a synthetic text sample and a sample generated in GTAV, both of which evade detection by ABCNet. Our dataset includes 50 such examples generated in GTAV, which are easy to generate, and easily readable by humans but are not detected by any of the tools.

This creates an interesting situation: as games and virtual environments become more immersive and expressive, players get even more freedom to manipulate the environment, and can potentially create arbitrary shapes of ever increasing variety and complexity.

4.5 Analyzing Controls and Keystrokes

A legitimate question arises as to whether Eve can track Alice and Bob's activity patterns within the game to check for suspicious behavior. We expect that spelling out a message in the game will entail significantly different behavior than normal gameplay. These patterns are captured in players' controls logs (keystrokes and mouse cursor positions) and can be parsed from their network traffic by sophisticated attackers [16].

Keystrokes dynamics have been widely used for identifying and authenticating parties (e.g. [34]), but we have been unable to find any research studies specific to gamers. We therefore devise our own experiment to use players' keystrokes to detect anomalous behavior which correlates with communications.

We define the following features:

Cursor position variance: This is the 2D on-screen variance in the cursor position. Text writing is predominantly a lateral movement so we expect these values to be higher during covert communications.

Cursor position variance during active mode: This feature records variance in movement when clicking the left mouse button consecutively, i.e., active mode. For instance, in GTAV or CoD4, when aiming at a

target, the gun usually remains still whereas for writing text it will exhibit high variance in either X- or Y-directions, corresponding to horizontal and vertical strokes of letters. This feature is meaningful in shooting games (GTAV, CoD4). For Minecraft one has to move while constructing structures, i.e., mouse clicks are separated by other keystrokes. To capture a similar effect in Minecraft, we use the count of double left clicks.

Average click count in active mode: If the player is writing text in addition to game play, he will likely use more bullets than usual in shooting games. The corresponding feature in Minecraft is the total click count. We use both left and right mouse click counts in Minecraft, as both indicate different actions.

Isolated Single Clicks: The player can even compose a character by firing individual bullets. This feature captures clandestine behaviour.

For Minecraft, considering that gameplay and composing messages are similar activities involving building structures, we also include the mean x-y position of the cursor. This results in six features for shooting games (GTAV, CoD4) and seven features for strategy (Minecraft) games respectively. We record in-game keystrokes using the Mini Mouse Macro tool and gather data for four activity modes consisting of combinations of text creation and normal gameplay (Fig. 11).

We start with the simple case where we record 100 minutes of keystrokes data each for normal gameplay and text generation. We segment this data into 1-minute units and for each we calculate the six dimensional feature vector. We use these to train multiple binary classifiers using the Weka machine learning toolbox [24]. The training set size is 55 units (feature vectors) and the test set size is 45 units (features) for each activity.

Results in Tab. 4 demonstrate that detection accuracy for gameplay and text generation (Fig. 11a-b) is near perfect (almost 100%) in GTAV. We undertook a similar training exercise using each feature individually and obtained near identical accuracy results (not listed here). This experiment validates our choice of features.

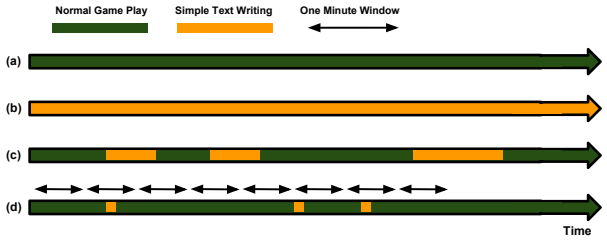


Fig. 11. Game modes for studying keystrokes dynamics (a) normal gameplay (b) normal communications (text writing) (c) gameplay and communications (d) clandestine communications

Classifier	Gameplay	Text Writing	Average
Random Forest	45/45 (100%)	45/45 (100%)	90/90 (100%)
Random Trees	42/45 (93.3%)	40/45 (88.8%)	82/90 (91.1%)
Logistic	42/45 (93.3%)	45/45 (100%)	87/90 (96.6%)
Multilayer Perceptron	45/45 (100%)	45/45 (100%)	90/90 (100%)
SMO (SVN)	45/45 (100%)	32/45 (71%)	77/90 (85.5%)

Table 4. Keystroke analysis to detect activities in GTAV

Classifier	Game Play	Text Writing	Average
Random Forest	15/30	15/30	30/60 (50%)
Random Trees	12/30	18/30	30/60 (50%)
Logistic	10/30	20/30	30/60 (50%)
Multilayer Perceptron	11/30	19/30	30/60 (50%)
SMO(SVN)	10/30	20/30	30/60 (50%)

Table 5. Keystroke analysis to detect intermittent text writing

Classifier	Game Play	Text Writing	Average
Random Forest	27/30	3/30	30/60 (50%)
Random Trees	30/30	0/30	30/60 (50%)
Logistic	30/30	0/30	30/60 (50%)
Multilayer Perceptron	30/30	0/30	30/60 (50%)
SMO(SVN)	30/30	0/30	30/60 (50%)

Table 6. Keystroke analysis to detect clandestine text writing

The performance of these classifiers degrades considerably when players try more creative and clandestine strategies for writing text. We experimented with two such strategies in GTAV. In the first mode, over a 30-minute period, the player plays the game for the most part and intermittently writes text on walls (Fig. 11c). Results in Tab. 5 indicate poor performance in detecting communications. The performance of all classifiers is 50%, which is the same as random guessing.

The second strategy is more clandestine (Fig. 11d). Here the player distributes the composition of text over a large time epoch. The majority of the keystrokes, therefore, are of normal gameplay and this obscures the communication activity entirely. As we expect, most classifiers flag this data as normal gameplay (Tab. 6).

Keystroke analysis in Minecraft reveals interesting results. Even without using any clandestine strategy, most classifiers perform poorly (Tab. 7). The best performing classifier (Multilayer Perceptron) achieves only 61.1% accuracy which is slightly better than random guessing. Even for human observers it is difficult to visually discriminate this behaviour. Can one guess whether the player is constructing text in Fig. 12? Here the player spells out the letter 'H', one stroke at a time, while undertaking other nearby activities. Detecting such behaviour requires consistent attention.

Classifier	Gameplay	Text Writing	Average
Random Forest	30/45 (67%)	23/45 (51%)	53/90 (58.8%)
Random Trees	25/45 (56%)	21/45 (47%)	46/90 (51.1%)
Logistic	28/45 (62%)	23/45 (51%)	51/90 (56.6%)
Multilayer Perceptron	25/45 (56%)	30/45 (67%)	55/90 (61.1%)
SMO (SVN)	33/45 (73%)	12/45 (27%)	45/90 (50%)

Table 7. Keystroke analysis for text writing in Minecraft

Classifier	Gameplay	Text Writing	Average
Random Forest	26/30 (87%)	15/30 (50%)	41/60 (68.3%)
Random Trees	22/30 (73%)	14/30 (47%)	36/60 (60%)
Logistic	22/30 (73%)	10/30 (33%)	32/60 (53.3%)
Multi-LayerPerceptron	20/30 (67%)	14/30 (47%)	34/60 (56.6%)
SMO (SVN)	30/30 (100%)	1/30 (3%)	31/60 (51.6%)

Table 8. Keystroke analysis fusion with image data for Minecraft

Finally, we analyse if fusion of image and keystroke features improves results for Minecraft. In addition to keystrokes, we captured screenshots at one minute intervals. From these images, we extract well-known deep features of FC7 layer of Alexnet [36]. Each image is converted into a 4096 dimensional vector. To maintain balance between image and keystroke features, we reduced the dimension from 4096 to 8 using principal component analysis (PCA). We retrained the classifier using image and keystroke features. Results in Tab. 8 indicate that fusion actually leads to further deterioration in results, indicating that image features could not distinguish between text construction and structure construction.

4.6 Adversarial Capabilities

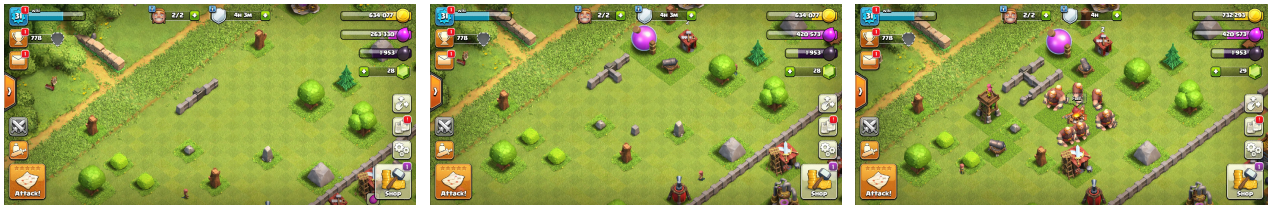
Here we summarize our results and theorize in more depth about Eve's capabilities. We have demonstrated that leading text spotting tools used out of the box perform poorly on in-game text. These tools can be retrained for individual environments by constructing appropriate datasets. However, the gains of retraining do not generalize effectively, so a new dataset should ideally be constructed for every game environment that is to be surveilled. The effort required for this exercise is specific to the game (we include basic time estimates in §. 5). We expect the costs of developing specific datasets, to retrain tools, render and scan hours of gameplay video streams will prove prohibitive for adversaries with limited resources, such as individual hackers and network administrators in small and mid-sized organizations.

Our messages exist in the visual domain and therefore Eve cannot detect them directly by examining the code-base or parsing raw network traffic of games. Eve can potentially reconstruct player actions from transmitted keystrokes and control logs in network traffic, but this approach has limited benefits. Accuracy improves for games such as GTAV and CoD4 but not Minecraft, and only for the naive case where there is a clear partition between message writing and gameplay activity. Players can blend the two activities to successfully evade detection. It is also an open question if it is practical for an adversary to undertake bulk data collection and real-time keystrokes analysis of very large volumes of game traffic.

If Eve is particularly resourceful, she could overcome certain technical limitations, such as modifying scene lighting, transforming surface textures and image perspectives, and selective rendering of bullet marks and items of interest. Theoretically a video game company could implement such a system on its gaming servers to detect undesirable communications. Eve could deploy bots to periodically scan game environments. These steps might incur a significant cost, but would certainly simplify the problem of spotting in-game text.

However, as we note in §. 4.4, text spotting tools have fundamental design limitations in that they fail to detect curved and arbitrary-shaped text, despite retraining, which is easy for human subjects to read.

Here, it may be possible for an attacker with considerable resources, such as a state actor or an intelligence agency, to push the state-of-the-art in text spotting to developing new tools catering to specific environments and able to handle text in more complex shapes. An attacker with such resources could even deploy human agents in games to search for communications.



(a) First stroke of letter (b) Some activity and second stroke (c) More activity and final stroke

Fig. 12. Fragmented construction of letter *H*

Indeed, there is evidence from the Snowden leaks, that several of the world's elite intelligence agencies, including the NSA, FBI, CIA, and GCHQ have actively investigated popular video games to search for potential terrorists, undertake criminal investigations, and recruit for counter-intelligence operations [49]. This effort included the development of custom tools to parse and analyze raw video game traffic as well as engage in active human infiltration in environments such as Second Life, World of Warcraft, and XboxLive.

The effectiveness of communicating via in-game text therefore ultimately come down to the capabilities of our adversary, Eve. We believe our results thus far justify this approach as an alternative to prior covert channel and steganographic solutions (documented in §. 6) where the threat model focuses on low-resource adversaries and Eve's capabilities are primarily restricted to analysis of application code and network traffic.

However, more research is required to theorize for the case of a resource-rich adversary who can modify the technical functioning of game and create large datasets and powerful new text spotting tools. In this regard, we consider our paper a preliminary exploration of this domain, which opens up lines of inquiry for future work.

5 Enhancements

Here we recap several limitations of this approach and explore various practical strategies to address them.

Generating in-game text is slow and inefficient and not conducive to interactive conversation or exchange of large amounts of data. Moreover, if messages are written on surfaces, other players might view them too, thereby potentially compromising privacy. This also applies to human adversaries in the game. It may also be difficult for Alice and Bob to authenticate each other's messages. Traditional security solutions, such as encryption, message authentication codes, and digital signatures cannot be applied here in a straightforward manner due to the severe bandwidth limitations of this medium.

Efficiency: Composing messages in games is laborious and time consuming compared to other communications tools. To develop a preliminary estimate, we requested three game players to write out all 36 characters of the alphanumeric character set in each of the three gaming environments while we timed them. There were 15 writing instances in all. The players were free to compose the characters any way they chose. Results are averaged and presented in boxplot format in Fig. 13. Composing

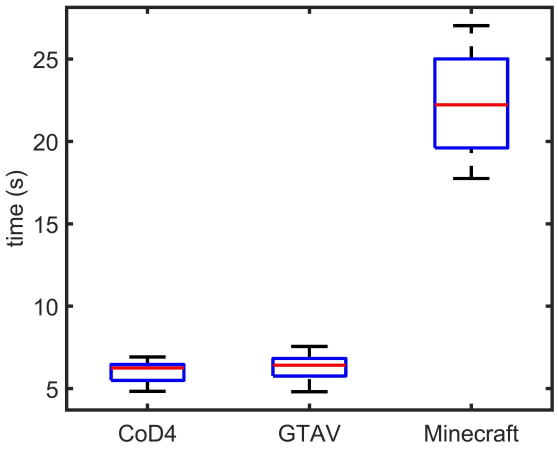


Fig. 13. Timing for individual characters per game

a character in GTAV and CoD4 is a point-and-shoot operation whereas Minecraft requires more effort. We also observe greater variation for Minecraft. Overall, our experiment yields bitrates of 1.11 bits/s, 1.17 bits/s, and 0.31 bits/s for GTAV, CoD4, and Minecraft respectively.

One strategy to ease this load is to use **encoding schemes** which reduce the character count of a message, such as Base58 and Base85, which strike a balance between human readability and efficiency. Second, there are **compaction strategies and tools** for specific content. For instance, IP addresses may be shorter and easier to type than URLs. Another popular option is URL shortening services.

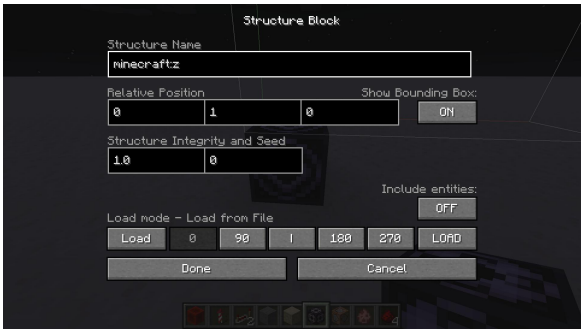
A third option is to **automate message generation** within games using tools which automate keystrokes and movement sequences. For instance, Minecraft includes a 'structure' block, which players may use to replicate built structures (depicted in Fig. 14(a)). Using this approach, we constructed a structure (e.g. the letter 'z'), saved it as a structure block (*minecraft:z* in Fig. 14(b)), and then deployed it when needed in one step (Fig. 14(c)). The structure block can also save entire words and sentences.

For shooter games, keystrokes can be automated using scripts. We successfully used Python's PyAutoGUI library to simulate key presses and create letters of our choice on surfaces in CoD4 and GTAV.

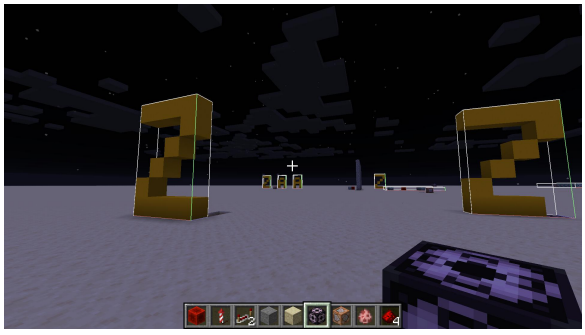
These techniques considerably reduce the effort and time involved in generating text. It may be possible to further automate the process, using computer vision techniques as deployed in robotics.

However, some tools may actually aid detection by an adversary. For instance, if Eve were to access key replay scripts, she could monitor players' keystrokes for the scripted patterns and thereby detect text creation. We intend to investigate this aspect in future work.

(a) structure block



(b) options to load saved structure



(c) result

Fig. 14. Structure block feature in Minecraft

Additional Stealth and Privacy: As noted earlier, a well-resourced adversary may send human agents to infiltrate games. In this context, we define *stealth* as the ability to obscure and conceal in-game messages from other human agents. We propose three practical strategies for this purpose:

First, several games feature large maps where Alice and Bob can interact without being spotted by Eve's agents. In GTAV, for instance, game designers have sought to replicate the city of Los Angeles accurately down to the district level, incorporating famous buildings, landmarks, and tourist spots, offering Alice and Bob various options for private meetings [11].

Second, some games offer flashlights to players to illuminate dark locations on a map [77]. Leaving messages in dark spots can further obscure them from other players in the game. Detecting these messages would require agents to have an in-depth knowledge of the map.

Third, in most shooter games, markings in the environment have a temporal lifetime. In older games like Counterstrike and early titles in the Call of Duty series, these marks would fade away after a few minutes.

(a) Original message (b) Destruction using grenade

Fig. 15. Destroying messages after viewing in Battlefield 4

In GTAV, we found that our text persists for more than six hours, over the entire day-night cycle of the game. In other games, marks disappear once the player changes his weapon. Another option, depicted in Fig. 15, is for Alice and Bob to destroy messages after reading. Minecraft features a '/fill' command which allows players to efficiently select and replace structures.

Alice and Bob may also use encryption to preserve confidentiality as an extra layer of defense in the chance that their messages are discovered. Modern stream ciphers, such as Salsa20, are suited for short messages because they do not add to message length.

Authentication: A pre-shared key between Alice and Bob can be used to authenticate messages and verify integrity using message authentication codes. We consider an example using HMAC-MD5, a popular keyed-hash message authentication code. Alice writes her message on the wall (*'When in doubt use bruteforce'*) and the corresponding HMAC code (in. Fig. 16). Compressing the HMAC using Base58 encoding reduces the character count from 32 to 22 characters.

However, a MAC does not enable non-repudiation. A corresponding public-key solution for such a scenario is an interesting problem for future work. Additionally, if we consider in-game communication primarily as a boostrap or dialling application, we can perhaps justify the diminished security guarantees and implement more rigorous checks at the next level.

Fig. 16. Authentication using message authentication codes

Offloading for Efficiency and Security: A convenient strategy is for users to **offload larger messages** to a different medium. This approach borrows from prior work on anti-censorship [5] [51] and botnets [35][2]. Within the game, Alice advertises a short pointer (a URL) to her real (and much larger) message. Such messages can be hosted on text storage sites, online clipboards, cloud-based file storage services, or even emailed to disposable email services. These services are largely anonymous and some even advertise a 'self-destruct' option which deletes the message after a time interval.

With this offloading technique we can also deploy rigorous cryptographic solutions. In-game, Alice shares links to text dumps, clipboards, and resources where she posts messages encrypted and signed by her private key. Bob navigates to the link, downloads, decrypts and verifies the messages using Alice's public key.

In this case, even if a human adversary were to discover the pointer in the game, she would not be able to decrypt the larger message. It is also unlikely that an attacker has control over the variety of offloading options to censor them en masse without inconveniencing legitimate users of those services.

6 Prior Work

There is considerable literature on **encoding communication within strategic moves** in a game. Winkler et al. have proposed bidding strategies in bridge, enabling a player to communicate information about his/her hand to a partner [92]. Hernandez-Castro et al. devise a game-theoretic framework for hiding data in strategic moves in chess, backgammon and Go [25]. Diehl refines the security notions and computes the data players can exchange undetected in multiplayer games [13]. Murdoch et al. propose covert channels via choice of game strategy and timing of moves in an online Connect-4 contest [55]. Smed et al. describe covert channels in poker and Age of Empires [74].

Johnson et al. formalized the notion of a **behavior-based covert channel** where two or more parties purposefully modify the internal states or behavior of an application to communicate information [33]. Such channels have been proposed using various applications, including anti-virus updates [3], web browsing patterns [70], and video games [33]. Hahn, et al. propose Castle, which encodes covert information in player activities within real-time strategy game (such as moving units and constructing buildings) [22]. Castle achieves up to 50-200 B/s bandwidth with a popular game like 0-A.D, which is practical for political organizational communications, such as email, SMS messages, and tweets.

Zander et al. encode covert bits as slight and constant variations in the movements of player avatars which are visually imperceptible to human players in the game but are deciphered by modified game clients [98]. They demonstrate their method using Quake III Arena, and achieve communication rates of 7-18 bit/s.

Steganographic approaches have been devised for simple computer games, such as Solitaire, Pong, Pac-Man [54], Soduku [73] [84], Tetris [62] [63], maze games [59] [39] [38], and chess [12], etc. A representative example is StegoRogue, a solution in which secret information is embedded in the design of maps (such as the number of rooms in the map, the number and types of items in the rooms) [19]. Other players accessing these maps compare this information against a dictionary to recover the original message.

A prominent contribution is Rook, which encodes communications within encrypted game packets [82]. This solution, demonstrated using Team Fortress 2, achieves a rate of approximately 30 b/s, suited to IRC-style chat applications. Since network traffic is encrypted, traffic analysis techniques cannot distinguish Rook traffic from normal game traffic. However, Rook requires sophisticated modifications to game clients and also suffers from communication 'outages' (when communicating players are not in range of each other).

Our work differs fundamentally from these solutions in important ways: prior work primarily relies on covert channels and steganography, whereas we exploit a challenging problem in computer vision. Our messages exist purely in the visual domain and cannot be tracked at the byte or code level. In contrast, the solutions described above can be compromised if an attacker knows about the covert channel or obtains the dictionary and/or encryption keys. In our case, a human viewer is needed to watch gameplay videos and identify communications. Another important difference is that our approach does not require technical sophistication on the part of users. However, our technique is laborious and time-consuming, and therefore best suited for low-bandwidth applications.

The work closest to ours is that of Hale, who proposes to embed information visually within games [23]. Users include text messages in the map itself using brushes, overlays, or by embedding image files. This technique is equivalent to use of native text that we discuss in our own work, and which is spotted with high accuracy by tools (Table 2). We include a sample gener-

(a) text sample (b) text spotting results

Fig. 17. Text sample and result (Hale)

ated using Hale's approach along with the transcribed output in Fig. 17.

This approach also requires parties to customize and share maps (using the Steam platform), which requires technical sophistication. If an attacker were to join the same server or obtain the map, she could easily detect the text using automated tools, whereas in our case a human agent has to physically watch gameplay footage to identify text. Furthermore, messages embedded in maps like native text become a permanent feature, whereas in our case, the messages are a temporary artefact which automatically expire, can be destroyed after viewing, or reset when the game concludes.

7 Conclusion and Future Work

In this paper we have explored the potential of using visual features within video games as a privacy enhancing technology against automated surveillance. We build an extensive dataset of such examples from three popular video games and demonstrate that this technique resists out-of-the-box detection by leading text spotting tools. Retraining these tools leads to improvements, but the costs and benefits vary across games, and this approach does not generalize to different environment. We investigate scene properties, including lighting, texture and perspective, and identify their impact on accuracy. We demonstrate that users can still evade detection using arbitrary shaped text. Moreover, mining players' activity data for patterns has limited effectiveness.

These findings allow us to theorize about potential attackers and also suggest various measures for users to further improve security, privacy, and efficiency. We also propose low bandwidth applications.

To the best of our knowledge, this is the first such exploration of video games and virtual environments from a computer vision perspective. In future work, we intend to explore various questions opened up by our inquiry, with regards to quantifying adversary capability, and improving on the usability and security constraints of this approach.

Future trends are difficult to predict: it is reasonable to expect that text spotting technology will likely improve and become more effective in the future. On the other hand, as computing power increases and with further developments in virtual reality technology, we anticipate that video games and virtual environments will become more immersive and interactive with time and offer participants still greater control in manipulating the environment.

8 Acknowledgements

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

We, the authors, wish to thank the anonymous reviewers for highly constructive feedback and also Rob Jansen for shepherding our paper. We also acknowledge the efforts of Waqas Amjad and other student volunteers at NUST-SEECS, for contributing to the first iteration of the GameText dataset, and for crafting the video game images featured in this paper.

References

[1] Haseeb Ahmad, Sardar Muhammad Usama, Wajahat Hussain, and Muhammad Latif Anjum. A sketch is worth a thousand navigational instructions. *Autonomous Robots*, pages 1–21, 2021.

[2] Syed Taha Ali, Patrick McCorry, Peter Hyun-Jeen Lee, and Feng Hao. Zombiecoin 2.0: managing next-generation botnets using bitcoin. *International Journal of Information Security*, 17(4):411–422, 2018.

[3] D Anthony, D Johnson, P Lutz, and B Yuan. A behavior based covert channel within anti-virus updates. In *Proceedings of the International Conference on Security and Management (SAM)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2012.

[4] Matt Burgess. This is why Russia's attempts to block Telegram have failed. https://www.wired.co.uk/article/telegram-in-russia-blocked-web-app-ban-facebook-twitter-google, April 2018.

[5] Sam Burnett, Nick Feamster, and Santosh Vempala. Chipping away at censorship firewalls with user-generated content. In *USENIX Security Symposium*, pages 463–468. Washington, DC, 2010.

[6] Michal Bušta, Lukáš Neumann, and Jirı Matas. Deep textspotter: An end-to-end trainable scene text localization

and recognition framework. In *IEEE International Conference on Computer Vision (ICCV), Venice*, pages 22–29, 2017.

[7] Xiangrong Chen and Alan L Yuille. Detecting and reading text in natural scenes. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. IEEE, 2004.

[8] Chee Kheng Ch'ng and Chee Seng Chan. Total-text: A comprehensive dataset for scene text detection and recognition. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 935–942. IEEE, 2017.

[9] Google Cloud. Cloud Vision. https://cloud.google.com/vision/.

[10] Samantha Cole. Second Life Users Are Protesting With Their Avatars. https://www.vice.com/en/article/kbgnwa/second-life-users-are-protesting-with-their-avatars, February 2017.

[11] Sarah Deen. Nice city! Photo series captures real life GTA V locations. https://metro.co.uk/2015/02/06/nice-city-photo-series-captures-real-life-gta-v-locations-5052766/#, February 2015.

[12] Abdelrahman Desoky and Mohamed Younis. Chestega: chess steganography methodology. *Security and Communication Networks*, 2(6):555–566, 2009.

[13] Malte Diehl. Secure covert channels in multiplayer games. In *Proceedings of the 10th ACM Workshop on Multimedia and Security*, pages 117–122. ACM, 2008.

[14] Entertainment Software Association (ESA). Essential facts about the computer and video game industry: 2018 sale, demographic, and usage data. http://www.theesa.com/wp-content/uploads/2018/05/EF2018_FINAL.pdf, 2018.

[15] Boris Epshtein, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2963–2970. IEEE, 2010.

[16] Wu-chang Feng, Francis Chang, Wu-chi Feng, and Jonathan Walpole. A traffic characterization of popular on-line games. *IEEE/ACM Transactions on Networking (TON)*, 13(3):488–500, 2005.

[17] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE, 2018.

[18] Dileep George, Wolfgang Lehrach, Ken Kansky, Miguel Lázaro-Gredilla, Christopher Laan, Bhaskara Marthi, Xinghua Lou, Zhaoshi Meng, Yi Liu, Huayan Wang, et al. A generative vision model that trains with high data efficiency and breaks text-based captchas. *Science*, 358(6368):eaag2612, 2017.

[19] Chance Gibbs and Narasimha Shashidhar. Stegorogue: Steganography in two-dimensional video game maps. *Advances in Computer Science: an International Journal*, 4(3):141–146, 2015.

[20] GTAall. Best Crews in GTA Online: April 2016. https://www.gtaall.com/gta-5/news/27947-gta-online-best-crews-april.html, April 2016.

[21] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2315–2324, 2016.

[22] Bridger Hahn, Rishab Nithyanand, Phillipa Gill, and Rob Johnson. Games without frontiers: Investigating video games as a covert channel. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 63–77. IEEE, 2016.

[23] Christopher Hale, Lei Chen, and Qingzhong Liu. A new villain: Investigating steganography in source engine based video games. In *Proceedings of the 2012 Hong Kong International Conference on Engineering & Applied Science (HKICEAS), Hong Kong, China, December 14*, volume 16, 2012.

[24] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[25] Julio C Hernandez-Castro, Ignacio Blasco-Lopez, Juan M Estevez-Tapiador, and Arturo Ribagorda-Garnacho. Steganography in games: A general methodology and its application to the game of go. *computers & security*, 25(1):64–71, 2006.

[26] Megan Hess. Fighting Turkey's Twitter Ban With DNS Graffiti. https://mashable.com/2014/03/21/twitter-ban-turkey-graffiti/#E7qrq61jAmqA, March 2014.

[27] Freedom House. Freedom on the net. *Washington, DC: Freedom House*, 2019.

[28] Jane Hu. The Second Act of Social-Media Activism. https://www.newyorker.com/culture/cultural-comment/the-second-act-of-social-media-activism, August 2020.

[29] Wajahat Hussain, Javier Civera, Luis Montano, and Martial Hebert. Dealing with small data and training blind spots in the manhattan world. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–9. IEEE, 2016.

[30] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014.

[31] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[32] Kanghyun Jo et al. Led dot matrix text recognition method in natural scene. *Neurocomputing*, 151:1033–1041, 2015.

[33] Daryl Johnson, Peter Lutz, and Bo Yuan. Behavior-based covert channel in cyberspace. In *Intelligent Decision Making Systems*, pages 311–318. World Scientific, 2010.

[34] Kevin S Killourhy and Roy A Maxion. Comparing anomaly-detection algorithms for keystroke dynamics. In *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pages 125–134. IEEE, 2009.

[35] Itzik Kotler and Ziv Gadot. Turbot: a next generation botnet. http://www.hackitoergosum.org/2010/HES2010-ikolter-zgadot-Turbot-Next-Generation-Botnet.pdf, 2010.

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[37] Pierre-Yves Laffont, Zhile Ren, Xiaofeng Tao, Chao Qian, and James Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Transactions on graphics (TOG)*, 33(4):1–11, 2014.

[38] Hasnain Lakhani and Fareed Zaffar. Covert channels in online rogue-like games. In *Communications (ICC), 2014 IEEE International Conference on*, pages 761–767. IEEE, 2014.

[39] Hui-Lung Lee, Chia-Feng Lee, and Ling-Hwei Chen. A perfect maze based steganographic method. *Journal of Systems and Software*, 83(12):2528–2535, 2010.

[40] Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu. Textboxes: A fast text detector with a single deep neural network. In *AAAI*, pages 4161–4167, 2017.

[41] Xuebo Liu, Ding Liang, Shi Yan, Dagui Chen, Yu Qiao, and Junjie Yan. Fots: Fast oriented text spotting with a unified network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5676–5685, 2018.

[42] Yuliang Liu, Hao Chen, Chunhua Shen, Tong He, Lianwen Jin, and Liangwei Wang. Abcnet: Real-time scene text spotting with adaptive bezier-curve network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9809–9818, 2020.

[43] Zichuan Liu, Guosheng Lin, Sheng Yang, Fayao Liu, Weisi Lin, and Wang Ling Goh. Towards robust curve text detection with conditional spatial expansion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7269–7278, 2019.

[44] Shangbang Long, Jiaqiang Ruan, Wenjie Zhang, Xin He, Wenhao Wu, and Cong Yao. Textsnake: A flexible representation for detecting text of arbitrary shapes. In *Proceedings of the European conference on computer vision (ECCV)*, pages 20–36, 2018.

[45] Asaf Lubin. A New Era of Mass Surveillance is Emerging Across Europe. https://www.justsecurity.org/36098/era-mass-surveillance-emerging-europe/, January 2017.

[46] Jianqi Ma, Weiyuan Shao, Hao Ye, Li Wang, Hong Wang, Yingbin Zheng, and Xiangyang Xue. Arbitrary-oriented scene text detection via rotation proposals. *IEEE Transactions on Multimedia*, 2018.

[47] Fabrizia Malgieri. Modern Warfare reaches 62.7 million active monthly users. https://www.gamereactor.eu/modern-warfare-reaches-627-million-active-monthly-users/, April 2020.

[48] Marco Margaritoff. 'Grand Theft Auto V' Is Most Profitable Entertainment Title Ever With $6 Billion in Sales. https://www.complex.com/pop-culture/2018/04/grand-theft-auto-v-5-most-profitable-entertainment-title-ever-6-billion-sales, April 2018.

[49] Mark Mazzetti and Justin Elliott. Spies Infiltrate a Fantasy Realm of Online Games. https://www.nytimes.com/2013/12/10/world/spies-dragnet-reaches-a-playing-field-of-elves-and-trolls.html, December 2013.

[50] Michael J Milford and Gordon F Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *2012 IEEE International Conference on Robotics and Automation*, pages 1643–1649. IEEE, 2012.

[51] Mohsen Minaei, Pedro Moreno-Sanchez, and Aniket Kate. R3c3: Cryptographically secure censorship resistant rendezvous using cryptocurrencies. https://eprint.iacr.org/2018/454.pdf, 2018.

[52] Cheang Ming. China has launched another crackdown on the internet ? but it's different this time. https://www.cnbc.com/2017/10/26/china-internet-censorship-new-crackdowns-and-rules-are-here-to-stay.html, October 2017.

[53] Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: Breaking a visual captcha. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2003.

[54] Anton Mosunov, Vineet Sinha, Heather Crawford, John Aycock, Daniel Medeiros Nunes de Castro, and Rashmi Kumari. Assured supraliminal steganography in computer games. In *International Workshop on Information Security Applications*, pages 245–259. Springer, 2013.

[55] Steven J Murdoch and Piotr Zieliński. Covert channels for collusion in online computer games. In *International Workshop on Information Hiding*, pages 355–369. Springer, 2004.

[56] Tayyab Naseer, Wolfram Burgard, and Cyrill Stachniss. Robust visual localization across seasons. *IEEE Transactions on Robotics*, 34(2):289–302, 2018.

[57] NetBlocks. Egypt filters 34,000 domains in bid to block opposition campaign platform. https://netblocks.org/reports/egypt-filters-34000-domains-in-bid-to-block-opposition-campaign-platform-7eA1blBp, April 2019.

[58] Nintendo. Animal Crossing: New Horizons Usage Guidelines for Businesses and Organizations . https://www.nintendo.co.jp/animalcrossing_announcement/en/index.html, November 2007.

[59] Naomoto Niwayama, Nasen Chen, Takeshi Ogihara, and Yukio Kaneda. A steganographic method for mazes. In *Proc. of Pacific Rim Workshop on Digital Steganography*, 2002.

[60] Tatiana Novikova, Olga Barinova, Pushmeet Kohli, and Victor Lempitsky. Large-lexicon attribute-consistent text recognition in natural images. In *European Conference on Computer Vision*, pages 752–765. Springer, 2012.

[61] Paul Benjamin Osterlund. Turkey marks one year without Wikipedia. https://www.theverge.com/2018/4/30/17302142/wikipedia-ban-turkey-one-year-anniversary, April 2018.

[62] Zhan-He Ou and Ling-Hwei Chen. Hiding data in tetris. In *Machine Learning and Cybernetics (ICMLC), 2011 International Conference on*, volume 1, pages 61–67. IEEE, 2011.

[63] Zhan-He Ou and Ling-Hwei Chen. A steganographic method based on tetris games. *Information Sciences*, 276:343–353, 2014.

[64] Gene Park. Joe Biden's 'Animal Crossing' island was definitely made by a pro gamer. https://www.washingtonpost.com/video-games/2020/10/16/biden-animal-crossing-island/, October 2020.

[65] Sarah Perez. Android Users Can Now Record And Publish Their Video Gameplay From The Google Play Games App. https://techcrunch.com/2015/10/28/android-users-can-now-record-publish-their-video-gameplay-from-the-google-play-games-app/, October 2015.

[66] Pranesh Prakash. Can India Trust Its Government on Privacy? https://india.blogs.nytimes.com/2013/07/11/can-india-trust-its-government-on-privacy/, July 2013.

[67] Siyang Qin. *Text Spotting in the Wild*. PhD thesis, UC Santa Cruz, 2018.

[68] Ahmed Sabir, Francesc Moreno-Noguer, and Lluís Padró. Textual visual semantic dataset for text spotting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 542–543, 2020.

[69] Daisy Schofield. Black Lives Matter meets Animal Crossing: how protesters take their activism into video games. https://www.theguardian.com/games/2020/aug/07/black-lives-matter-meets-animal-crossing-how-protesters-take-their-activism-into-video-games, August 2020.

[70] Yao Shen, Wei Yang, and Liusheng Huang. Concealed in web surfing: Behavior-based covert channels in http. *Journal of Network and Computer Applications*, 101:83–95, 2018.

[71] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2017.

[72] Baoguang Shi, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Robust scene text recognition with automatic rectification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4168–4176, 2016.

[73] M Hassan Shirali-Shahreza and Mohammad Shirali-Shahreza. Steganography in sms by sudoku puzzle. In *Computer Systems and Applications, 2008. AICCSA 2008. IEEE/ACS International Conference on*, pages 844–847. IEEE, 2008.

[74] Jouni Smed, Timo Knuutila, and Harri Hakonen. Can we prevent collusion in multiplayer online games. In *Proceedings of the Ninth Scandinavian Conference on Artificial Intelligence (SCAI 2006)*, pages 168–175, 2006.

[75] Congzheng Song and Vitaly Shmatikov. Fooling ocr systems with adversarial text images. *arXiv preprint arXiv:1802.05385*, 2018.

[76] Keith Stuart. Gamer communities: the positive side. https://www.theguardian.com/technology/gamesblog/2013/jul/31/gamer-communities-positive-side-twitter, July 2013.

[77] Brian Taylor. 9 Great Videogame Flashlights. https://www.pastemagazine.com/articles/2016/04/9-great-videogame-flashlights.html, April 2016.

[78] Iain Thomson. Germany, France lobby hard for terror-busting encryption backdoors ? Europe seems to agree. https://www.theregister.co.uk/2017/02/28/german_french_ministers_breaking_encryption/, February 2017.

[79] Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting text in natural image with connectionist text proposal network. In *European Conference on Computer Vision*, pages 56–72. Springer, 2016.

[80] PG Vandana and Bobbinpreet Kaur. A novel technique for led dot-matrix text detection and recognition for non-uniform color system. In *Advances in Computing, Communications and Informatics (ICACCI), 2016 International Conference on*, pages 2750–2754. IEEE, 2016.

[81] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge J. Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *arXiv preprint arXiv:1601.07140*, 2016.

[82] Paul Vines and Tadayoshi Kohno. Rook: Using video games as a low-bandwidth censorship resistant communication platform. In *Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society*, pages 75–84. ACM, 2015.

[83] Kai Wang and Serge Belongie. Word spotting in the wild. In *European Conference on Computer Vision*, pages 591–604. Springer, 2010.

[84] Zhi-Hui Wang, Chin-Chen Chang, Ming-Chu Li, et al. A sudoku based wet paper hiding scheme. *International Journal of Smart Home*, 3(2):1–11, 2009.

[85] Tom Warren. Minecraft still incredibly popular as sales top 200 million and 126 million play monthly. https://www.theverge.com/2020/5/18/21262045/minecraft-sales-monthly-players-statistics-youtube, May 2020.

[86] Human Rights Watch. Russia: ?Big Brother? Law Harms Security, Rights. https://www.hrw.org/news/2016/07/12/russia-big-brother-law-harms-security-rights, July 2016.

[87] Human Rights Watch. Belarus: Internet Disruptions, Online Censorship. https://www.hrw.org/news/2020/08/28/belarus-internet-disruptions-online-censorship, August 2020.

[88] Sarah Wheaton. Obama Is First in Their Second Life. https://thecaucus.blogs.nytimes.com/2007/03/31/obama-is-first-in-their-second-life/, March 2007.

[89] Brenda K Wiederhold. Social media and social organizing: From pandemic to protests. *Cyberpsychology, Behavior, and Social Networking*, 23(9):579–580, 2020.

[90] Tom Wijman. The World's 2.7 Billion Gamers Will Spend $159.3 Billion on Games in 2020; The Market Will Surpass $200 Billion by 2023. https://newzoo.com/insights/articles/newzoo-games-market-numbers-revenues-and-audience-2020-2023/, May 2020.

[91] Tommy Wilkes. Pakistan lifts ban on YouTube after launch of local version. https://www.reuters.com/article/us-pakistan-youtube-idUSKCN0UW1ER, January 2016.

[92] Peter Winkler. The advent of cryptology in the game of bridge. *Cryptologia*, 7(4):327–332, 1983.

[93] Xing Xu, Jiefu Chen, Jinhui Xiao, Lianli Gao, Fumin Shen, and Heng Tao Shen. What machines see is not what they get: Fooling scene text recognition models with adversarial text images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12304–12314, 2020.

[94] Guixin Ye, Zhanyong Tang, Dingyi Fang, Zhanxing Zhu, Yansong Feng, Pengfei Xu, Xiaojiang Chen, and Zheng Wang. Yet another text captcha solver: A generative adversarial network based approach. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 332–348, 2018.

[95] Xu-Cheng Yin, Xuwang Yin, Kaizhu Huang, and Hong-Wei Hao. Robust text detection in natural scene images. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):970–983, 2014.

[96] Liu Yuliang, Jin Lianwen, Zhang Shuaitao, and Zhang Sheng. Detecting curve text in the wild: New dataset and new solution. *arXiv preprint arXiv:1712.02170*, 2017.

[97] A.R. Zamir and M. Shah. Image geo-localization based on multiple nearest neighbor feature matching using generalized graphs. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PP(99):1–1, 2014.

[98] Sebastian Zander, Grenville Armitage, and Philip Branch. Covert channels in multiplayer first person shooter online games. In *Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference on*, pages 215–222. IEEE, 2008.

[99] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.