None of My Pixel is Your Business: Active Watermarking Cancellation Against Video Streaming Service

Authors

- Wang Kang
  - Alibaba Group
  - TUNA
- Hui Yi-Qun
  - Tsinghua University, Master Candidate
  - TUNA

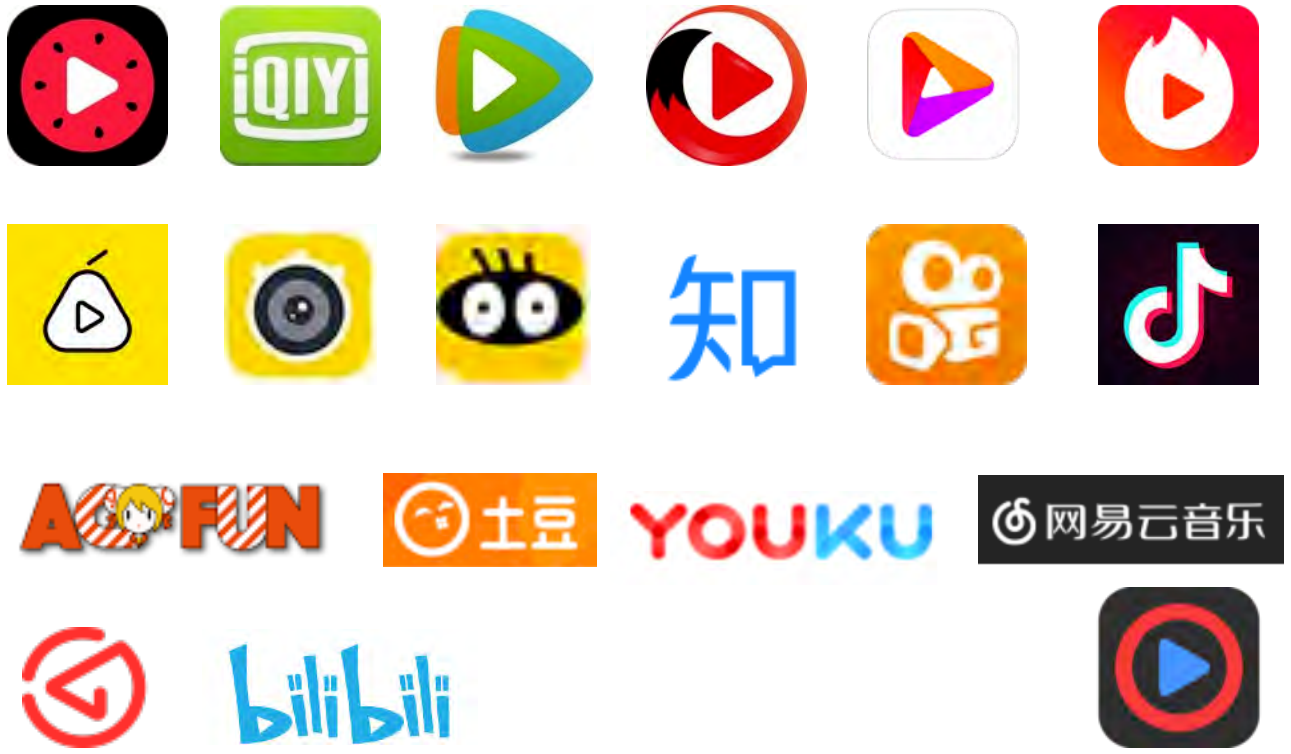
(FamilyName_GivenName)

Outline

- Background
  - Problem & Market Overview
  - Visible Watermark
  - Invisible Watermark (Forensic Watermark)
- (0/3)Theoretical Estimation
    - Principles
    - Method #1: Pre-distort
    - Method #2: Post-process
    - Lossless Recovery Range
- (1/3)Logo Extraction
    - Principles
    - Code
- (2/3)Static Experiments
- (3/3)Realtime Processing
  - FFmpeg & halide
  - libtmblock
  - Experiment Results
    - Discussion
- Related & Future Work
  - With AI
- Our Goal
  - Drop Visible Watermark
  - Use DRM, or like a real man.

Landscape: Video Streaming Services in China

http://v.163.com/
http://music.163.com/
http://www.56.com/
http://www.acfun.tv/
http://tieba.baidu.com/
http://www.baomihua.com/
http://www.bilibili.com/
http://www.dilidili.com/
http://www.douban.com/
http://www.douyutv.com/
http://www.panda.tv/
http://v.ifeng.com/
http://www.fun.tv/
http://www.iqiyi.com/
http://www.joy.cn/
http://www.ku6.com/
http://www.kugou.com/
http://www.kuwo.cn/
http://www.le.com/
http://www.lizhi.fm/
http://www.miaopai.com/
http://www.miomio.tv/
https://www.douyin.com/
https://www.pixnet.net/
http://www.pptv.com/
http://v.iqilu.com/
http://v.qq.com/
http://live.qq.com/
http://video.sina.com.cn/
http://video.weibo.com/
http://tv.sohu.com/
http://www.tudou.com/
http://www.xiami.com/
http://www.isuntv.com/
http://www.yinyuetai.com/
http://www.youku.com/
http://www.zhanqi.tv/lives
http://www.cntv.cn/
http://huaban.com/
http://tvcast.naver.com/
http://www.mgtv.com/
http://www.huomao.com/
http://www.quanmin.tv/
http://www.365yg.com/
https://www.ixigua.com/
https://www.kuaishou.com/



See Also:
https://github.com/soimort/you-get/blob/develop/README.md#supported-sites

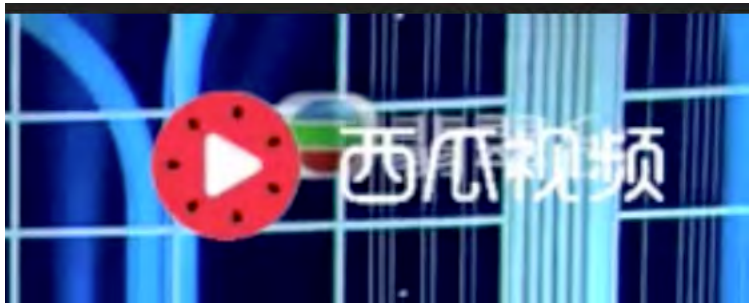
```
# youtube-dl --list-extractors
```

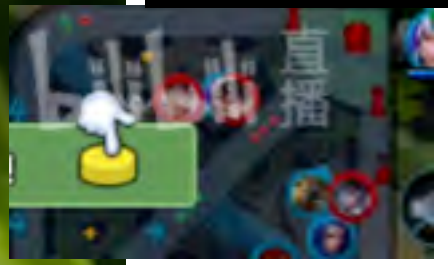
Watermark Technology Overview

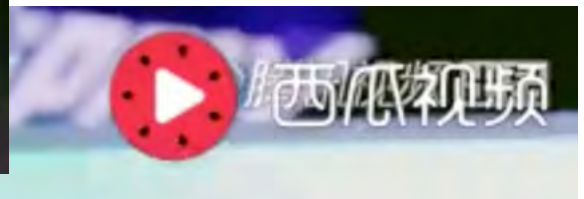
- Forensic/Invisible Watermark
  - Transform-domain
    - Wavelet-based
    - Fourier-based
  - Compressed-domain
    - Bitstream
  - Spatial-domain
    - LSB
    - Blue Channel (simple but effective)
- Embedding and Extracting method
  - Additive
  - Multiplicative

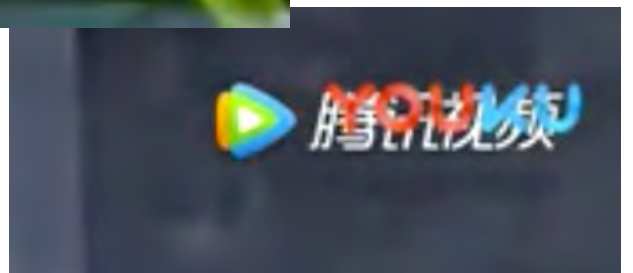
Fw:Fw:Fw:Fw:Fw:Fw: Watermark

- Wild west of copyright
- A highly competitive market
- Forwarded and reposted.. ..
- Just don’t care.. UX





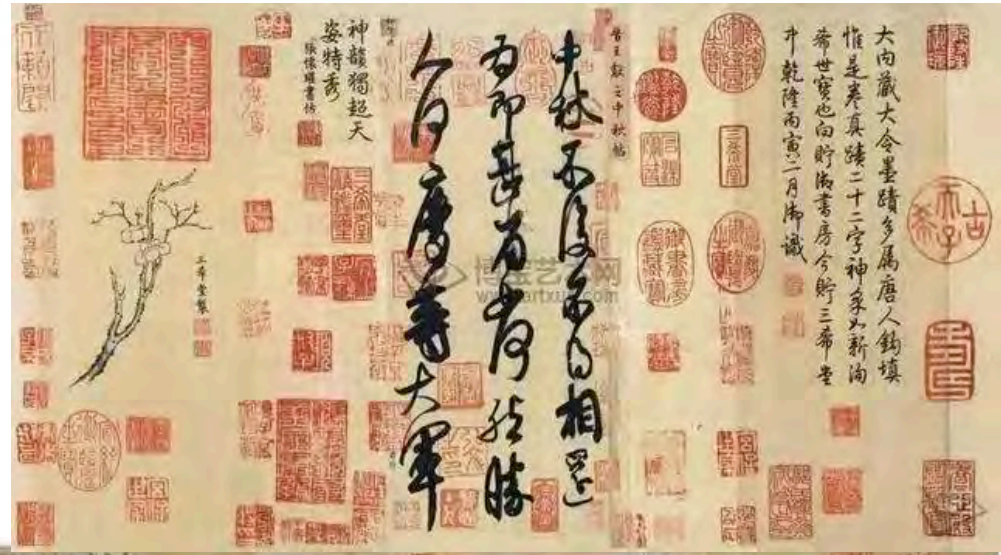


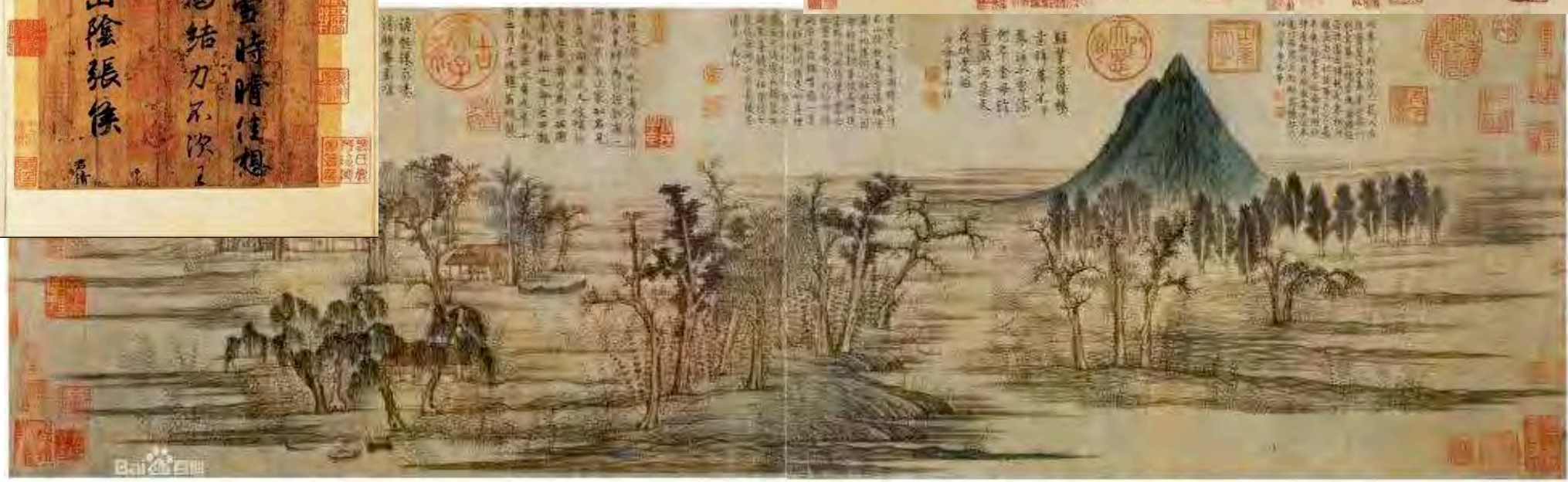


A Historical Tradition?

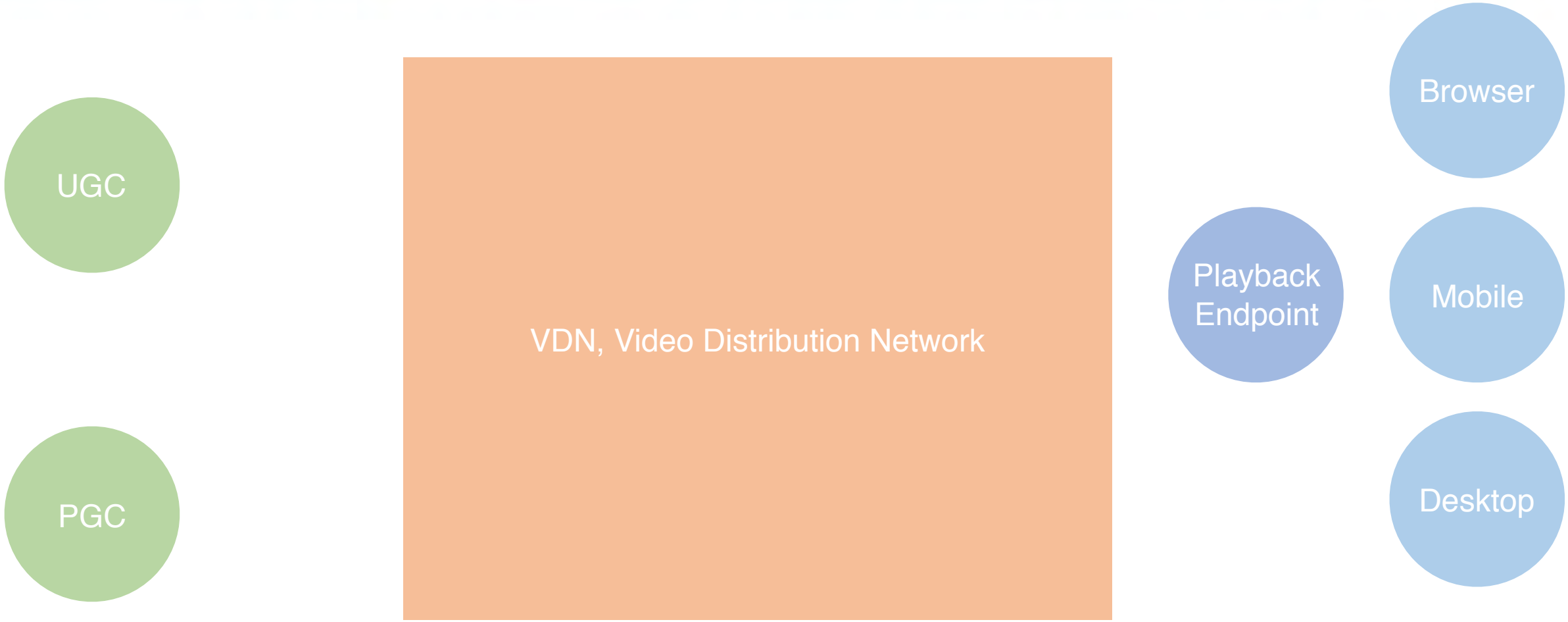
by Emperor 乾隆 (1735-1795)

神: *Oh God*

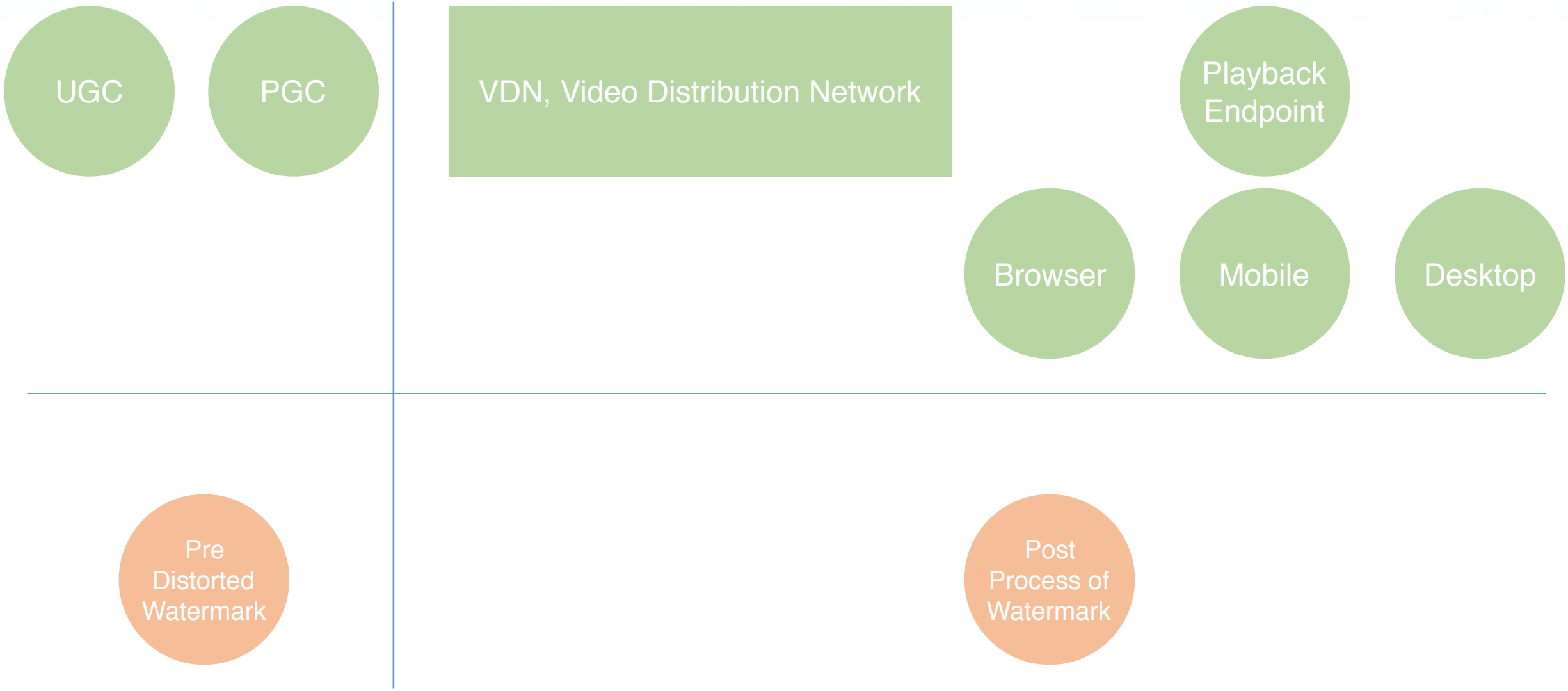




Full Life Circle of Video



Full Life Circle of Video



Wish List

- A library that collects all logos
  - high-precision, with alpha channel, size, position
- A FFmpeg filter that is able to do:
  - pre-distortion with specified logo before uploading/streaming
  - post-process with specified logo
- A browser plugin/mobile app that can:
  - perform post-process against watermarked video

(0/3)

Theoretical Estimation

to get the probability of recovery

Definition of Alpha Compositing

$$(R_{Output}, G_{Output}, B_{Output}) = (1 - \alpha)(R_{Input}, G_{Input}, B_{Input}) + \alpha(R_{Logo}, G_{Logo}, B_{Logo})$$

https://en.wikipedia.org/wiki/Alpha_compositing#Composing_alpha_blending_with_gamma_correction

Proposed Method #1: Pre-distortion

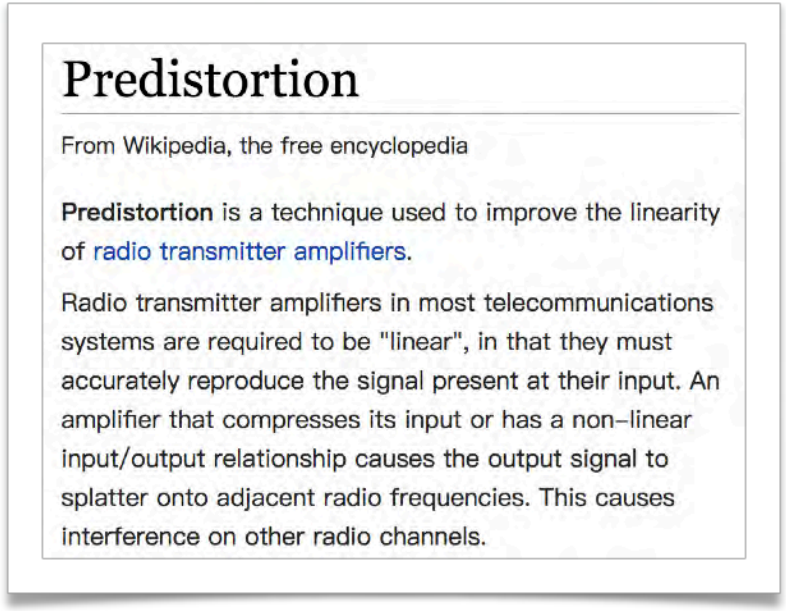
$$P' = \frac{1}{1-\alpha}P - \frac{\alpha}{1-\alpha}X$$

Where:

P: (R,G,B) of Original Pixel
α: the α channel of Extracted Logo
X: (R,G,B) of Extracted Logo
P': (R,G,B) of Predistorted Pixel

Proof: (When watermarked later…)

$$(1-\alpha)P' + \alpha X$$
$$= P - \alpha X + \alpha X = P$$



Predistortion

From Wikipedia, the free encyclopedia

Predistortion is a technique used to improve the linearity of radio transmitter amplifiers.

Radio transmitter amplifiers in most telecommunications systems are required to be "linear", in that they must accurately reproduce the signal present at their input. An amplifier that compresses its input or has a non-linear input/output relationship causes the output signal to splatter onto adjacent radio frequencies. This causes interference on other radio channels.

Lossless Limitation:

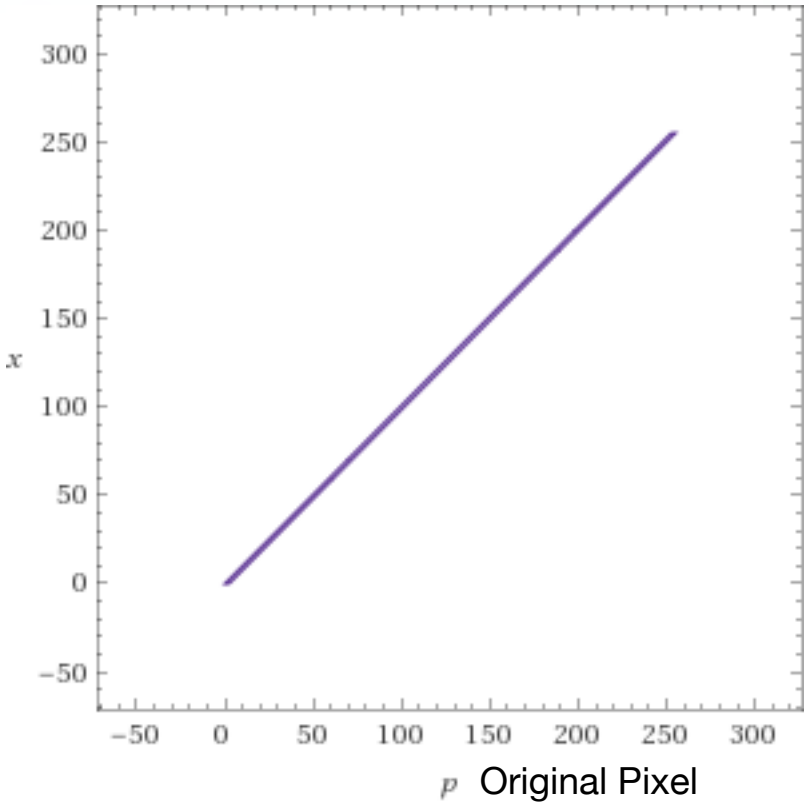
$$\begin{cases} P, P', X \in [0,255] \\ \alpha \in [0,1) \end{cases}$$

Given α and X, the original pixel must be between:

$$P \in [\alpha X, \alpha X + 255(1-\alpha)]$$

Visualizing α=0 .. 1

Logo Pixel

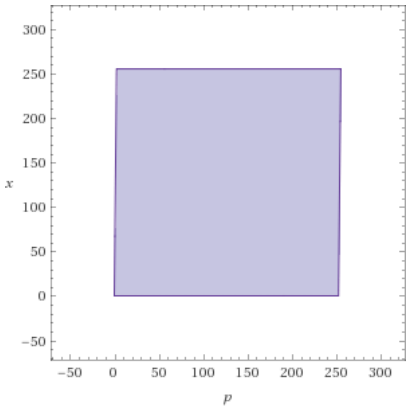


α=0.99

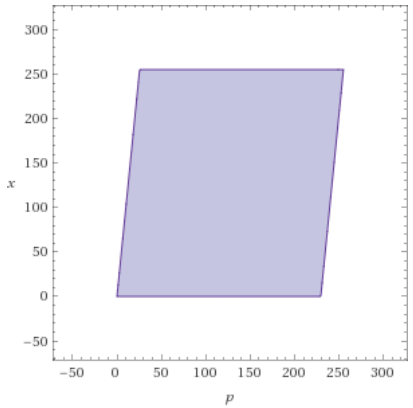
$$P \in [\alpha X, \alpha X + 255(1-\alpha)]$$

Recoverable Color Area

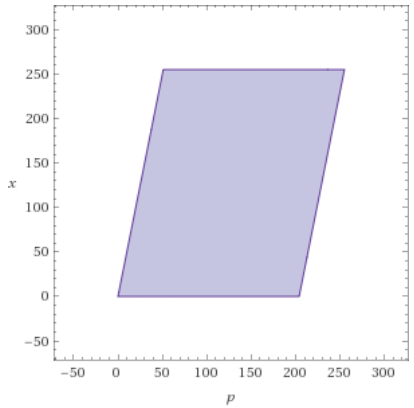
α=0 .. 1



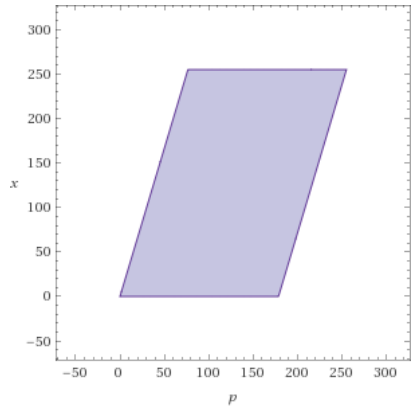
α=0



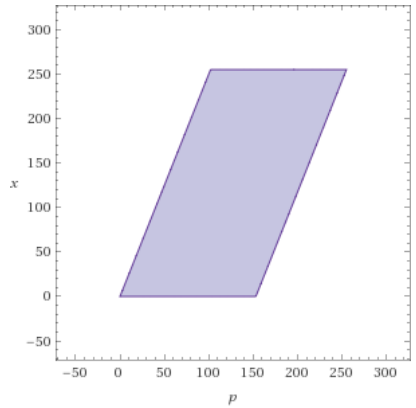
α=0.1



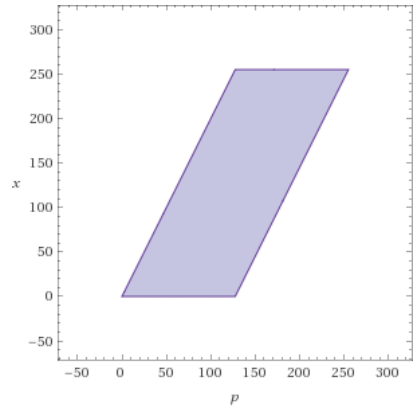
α=0.2



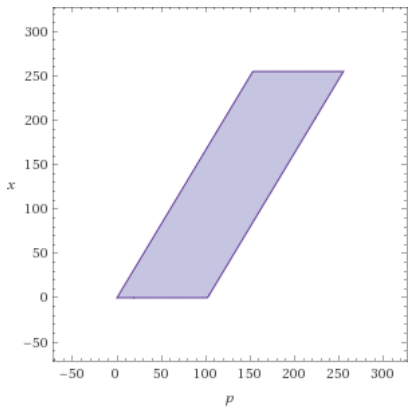
α=0.3



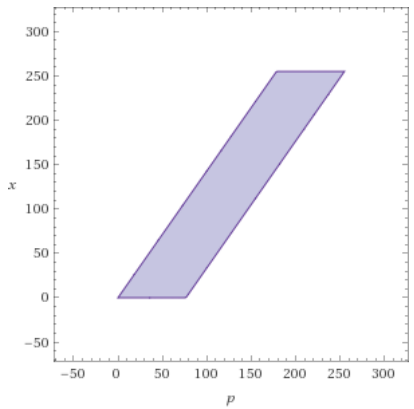
α=0.4



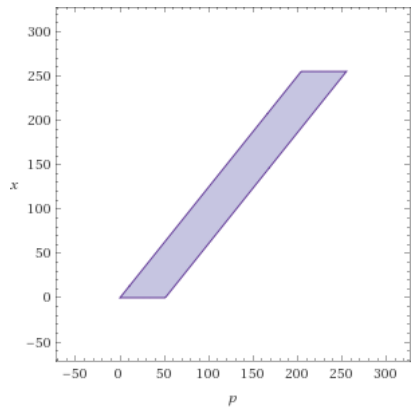
α=0.5



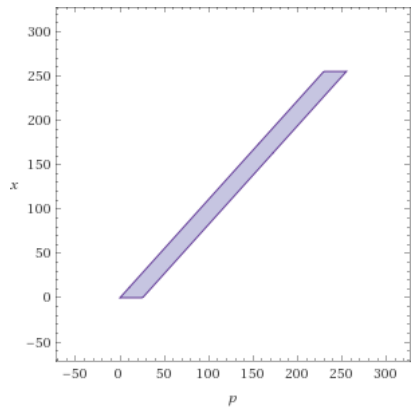
α=0.6



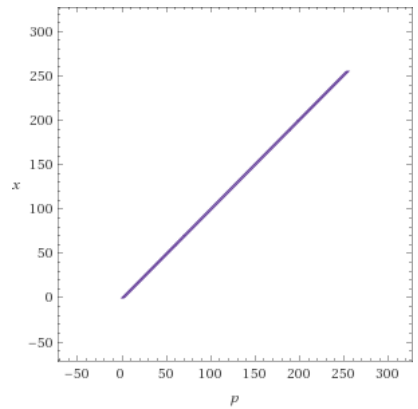
α=0.7



α=0.8



α=0.9



α=0.99

Recoverable Color Area

Proposed Method #2: Post-process

$$RGB_{Output} = (1-\alpha)RGB_{Input} + \alpha RGB_{Logo}$$

$$P = (1-\alpha)P_0 + \alpha X$$

$$P' = P_0 = \frac{1}{1-\alpha}P - \frac{\alpha}{1-\alpha}X$$

Where:

P_0: (R,G,B) of Original Pixel
P: (R,G,B) of Watermarked Pixel
α: the α channel of Extracted Logo
X: (R,G,B) of Extracted Logo
P': (R,G,B) of Recovered Pixel

Lossless Limitation:

$$P \in [0,255]$$

Quantitative loss: int -> float -> int

(compensated by human brain)

Engineering Challenges

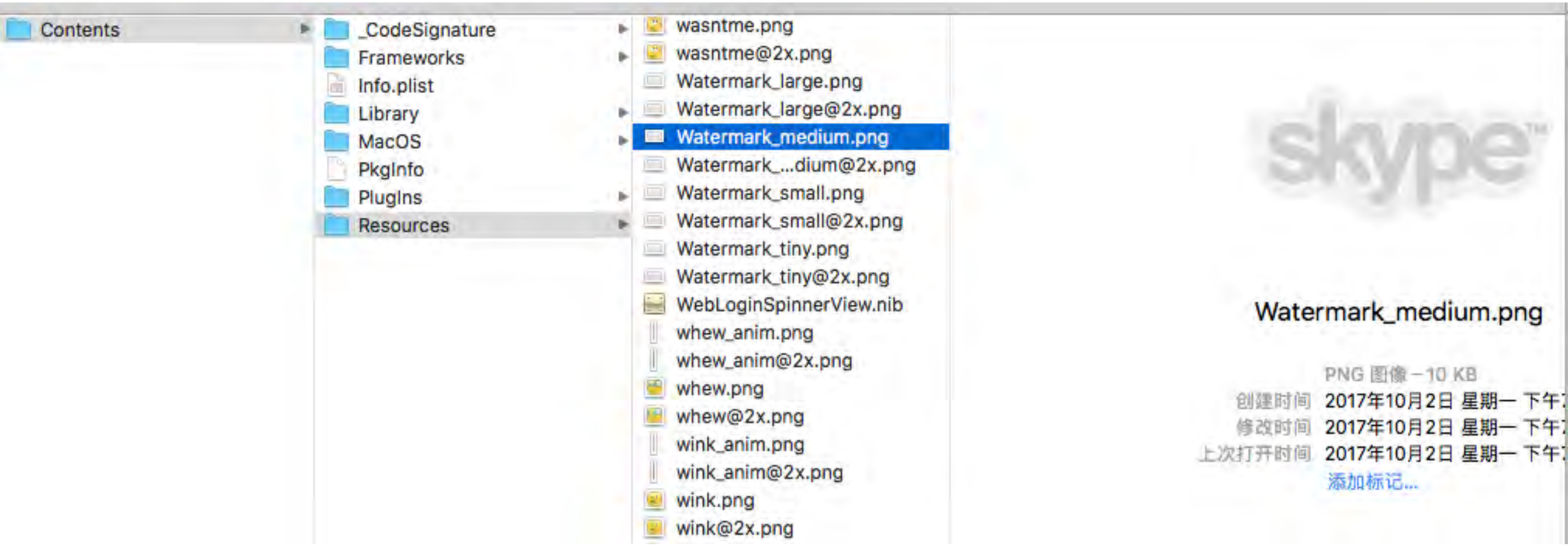
- Extraction of Transparent Watermark
- Theoretical Recoverable Range
- Position Fine-Tuning
- Shadow and Edge Handling
- Color Space Management
  - sRGB / Linear RGB
  - Gamma Correction
- Realtime Performance
  - Framework
  - Live Streaming Protocol
  - Video Codec

(1/3)

Logo Extraction

with high-precision metadata

if lucky enough..



α Channel Recovery of Logo (Watermark)

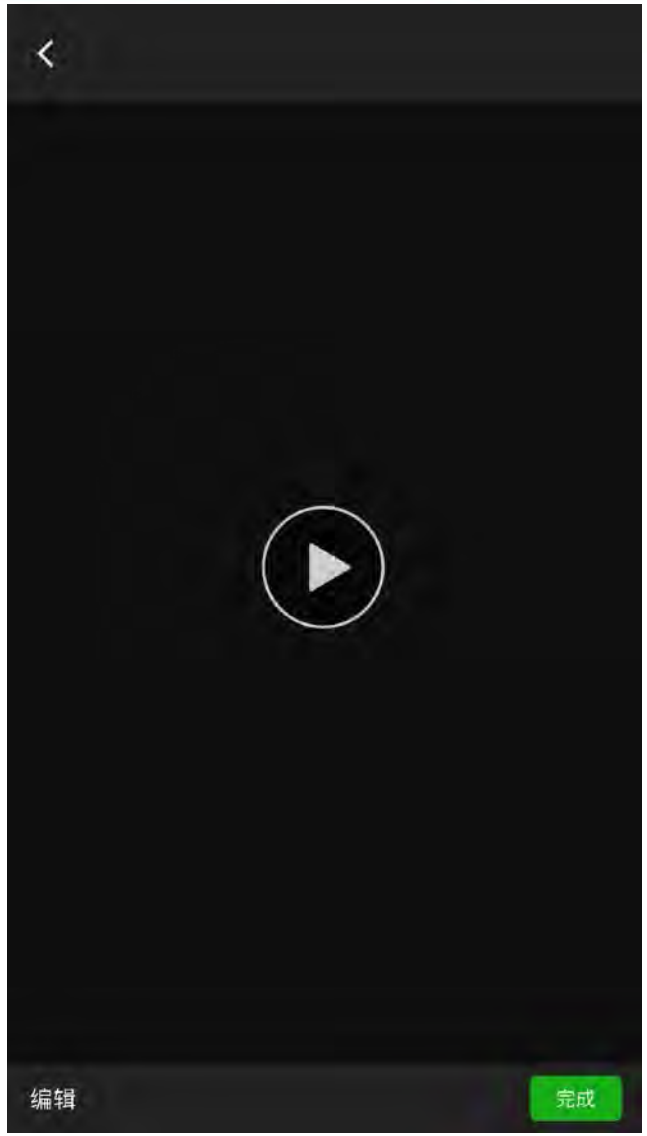
$$Y_1 = (1-\alpha)X_1 + \alpha L$$
$$X_1 = (0,0,0)$$

$$Y_2 = (1-\alpha)X_2 + \alpha L$$
$$X_2 = (255,255,255)$$

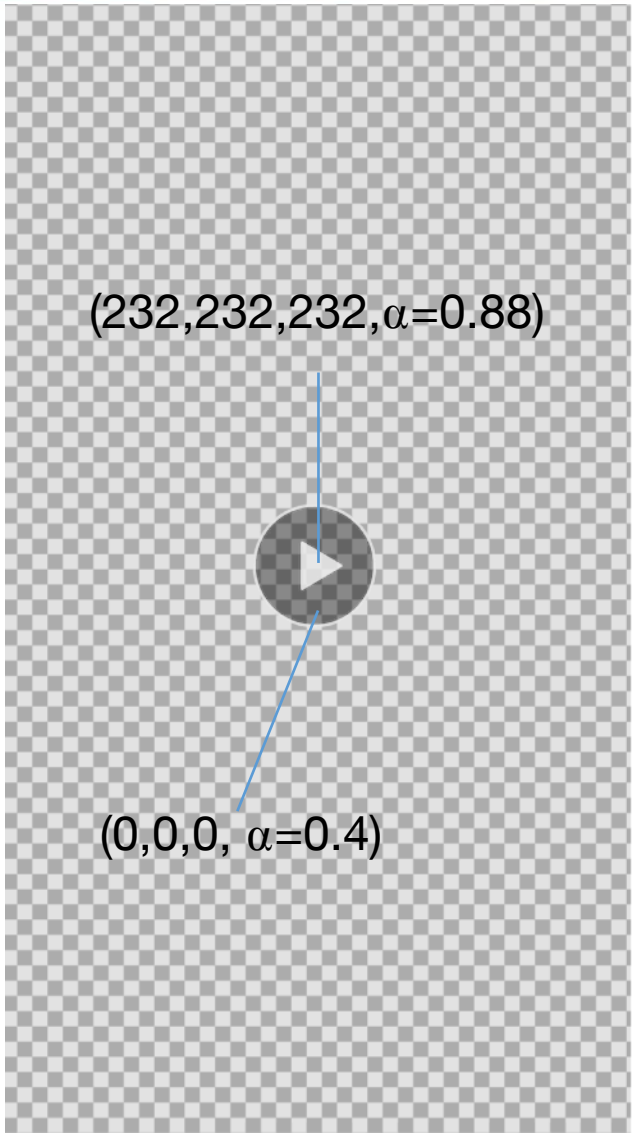
**With enough patience.. just watch and wait for a pure-color scene*

$$\alpha = 1 - \frac{\Delta X}{\Delta Y}$$

$$L = \frac{\Delta X \cdot Y_1 - \Delta Y \cdot X_1}{\Delta X - \Delta Y}$$


编辑
完成


编辑
完成


(232,232,232,α=0.88)
(0,0,0, α=0.4)

Logo Extracting Automation

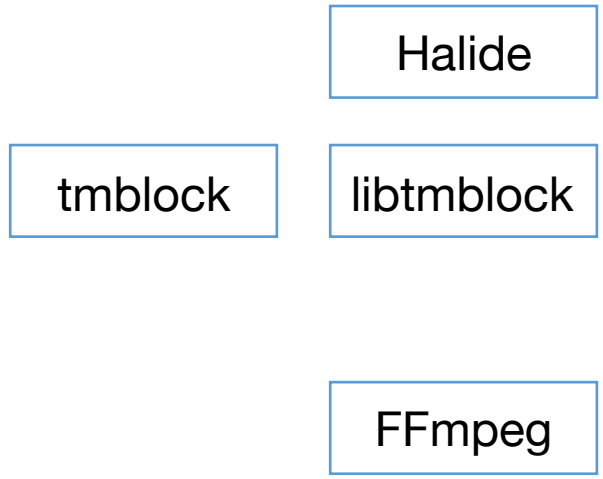
- As a content provider, you can just inject two {black, white} frames..
  - two of any pure color scenes are fine
  - a simple automation script should do well.
  - Or if you really get sufficient funds,
    - deploy an advertisement.
- Other Methods for Extracting
  - Pixel Estimation with AI
  - Long Exposure Algorithm
    - Time-varying part averaged

(2/3)

Static Experiments

Effective Development Approach

- Challenge: Image/video processing is
  - not so easy to develop
  - hard to debug
- Write Python prototype scripts
  - tuning: algorithm, media resources, color, position
  - *life is short, use Python*
  - https://github.com/tmblock/testcase
- Write C++ code with Halide
  - to improve performance
  - to integrate with FFmpeg for realtime processing
  - compare with Python version as test case



Halide: a language for image processing and computational photography

http://halide-lang.org/

Halide is a new programming language designed to make it easier to write high-performance image processing code on modern machines. Its current front end is embedded in C++. Compiler targets include x86/SSE, ARM v7/NEON, CUDA, and OpenCL.

MIT CSAIL
2013

TensorFlow
GNURadio
Apache mxnet: Halide IR(Intermediate Representation)
dmlc/HalideIR

- enough performance by native scheduler
- enough space for future optimization

PRE-01-origin.png

PRE-02-pre-anti-watermarked.png

PRE-03-normal-watermarked.png

PRE-04-final.png

Pre-distorted

POST-03-output.png

Cancelled Result:
Pre-distorted then Watermarked

Method #2: Post

PRE-01-origin.png

PRE-02-pre-anti-watermarked.png

PRE-03-normal-watermarked.png

PRE-04-final.png

POST-03-output.png

PRE-01-origin.png

PRE-02-pre-anti-watermarked.png

PRE-03-normal-watermarked.png

PRE-04-final.png

POST-03-output.png

PRE-01-origin.png

PRE-02-pre-anti-watermarked.png

PRE-03-normal-watermarked.png

PRE-04-final.png

POST-03-output.png

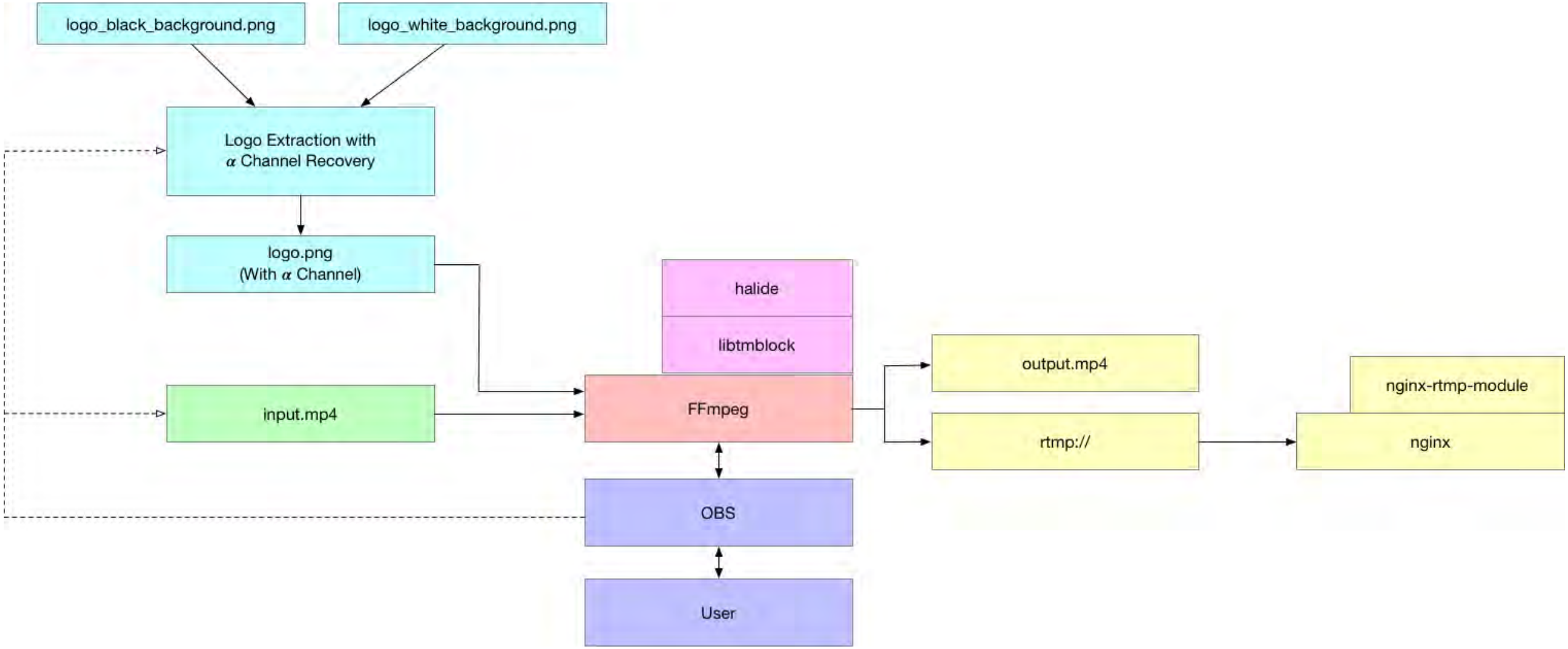
(3/3)

Realtime Processing

with FFmpeg

Project ™Block

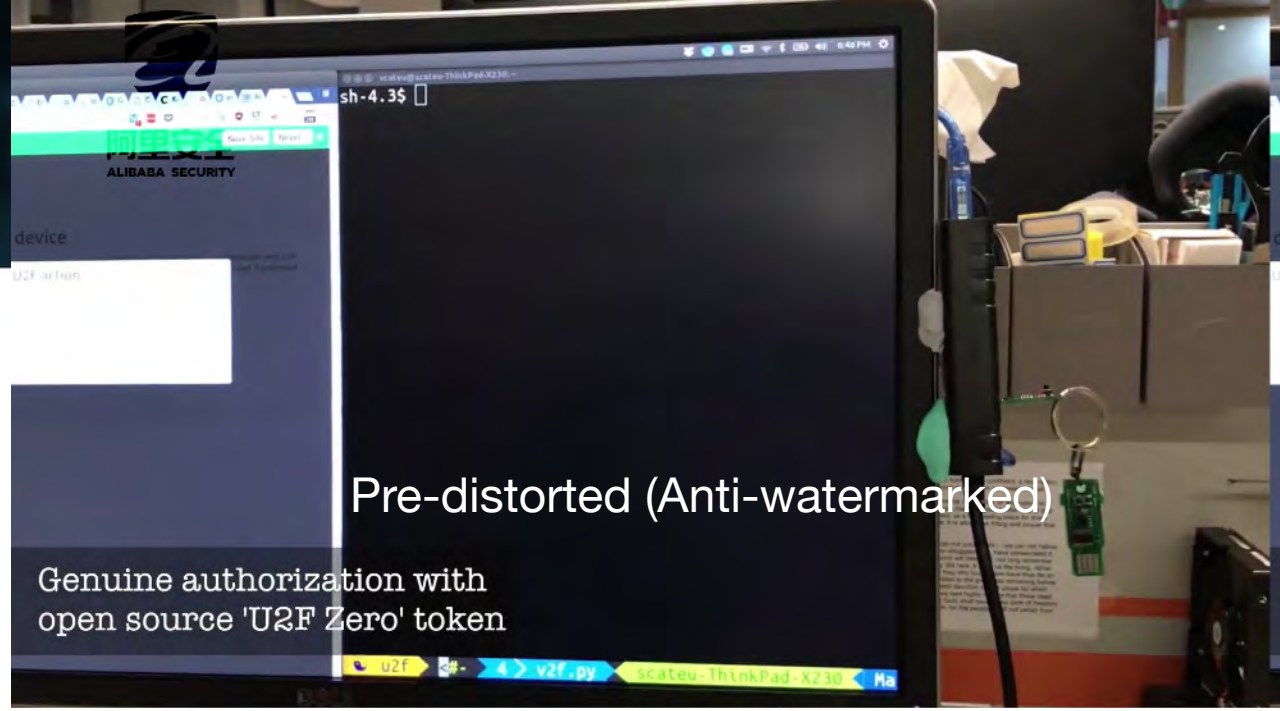
System Architecture

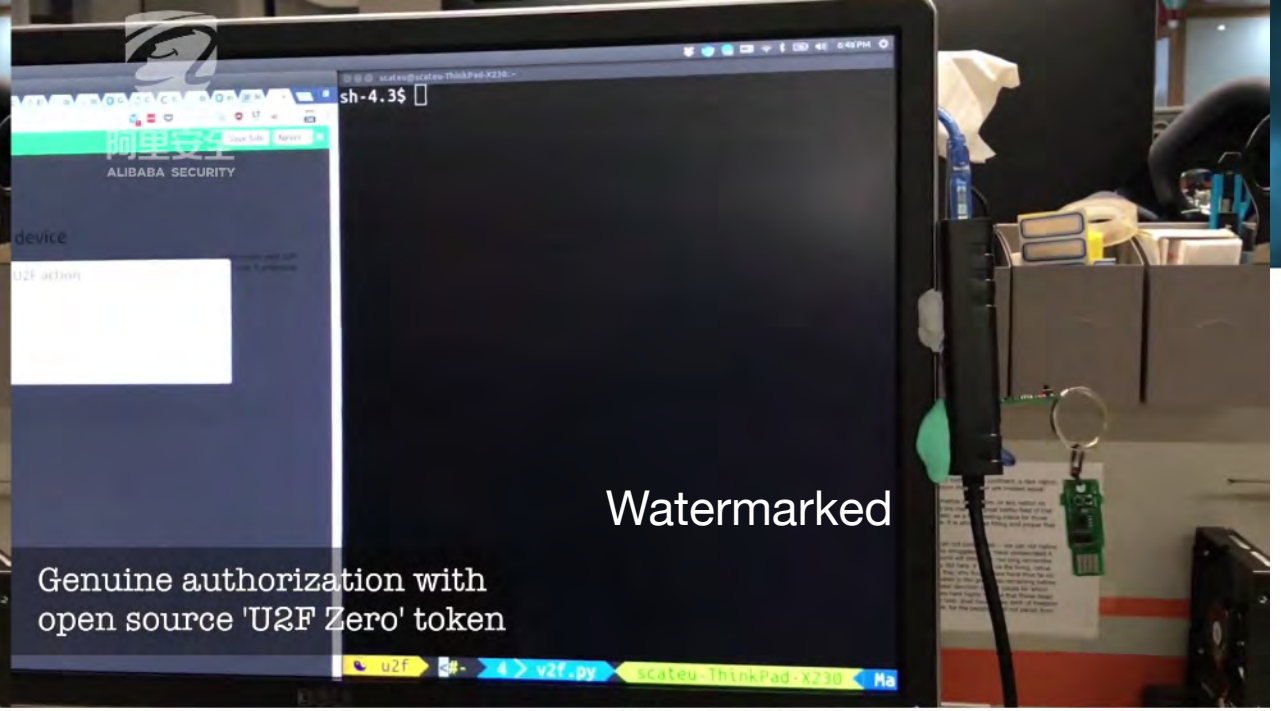


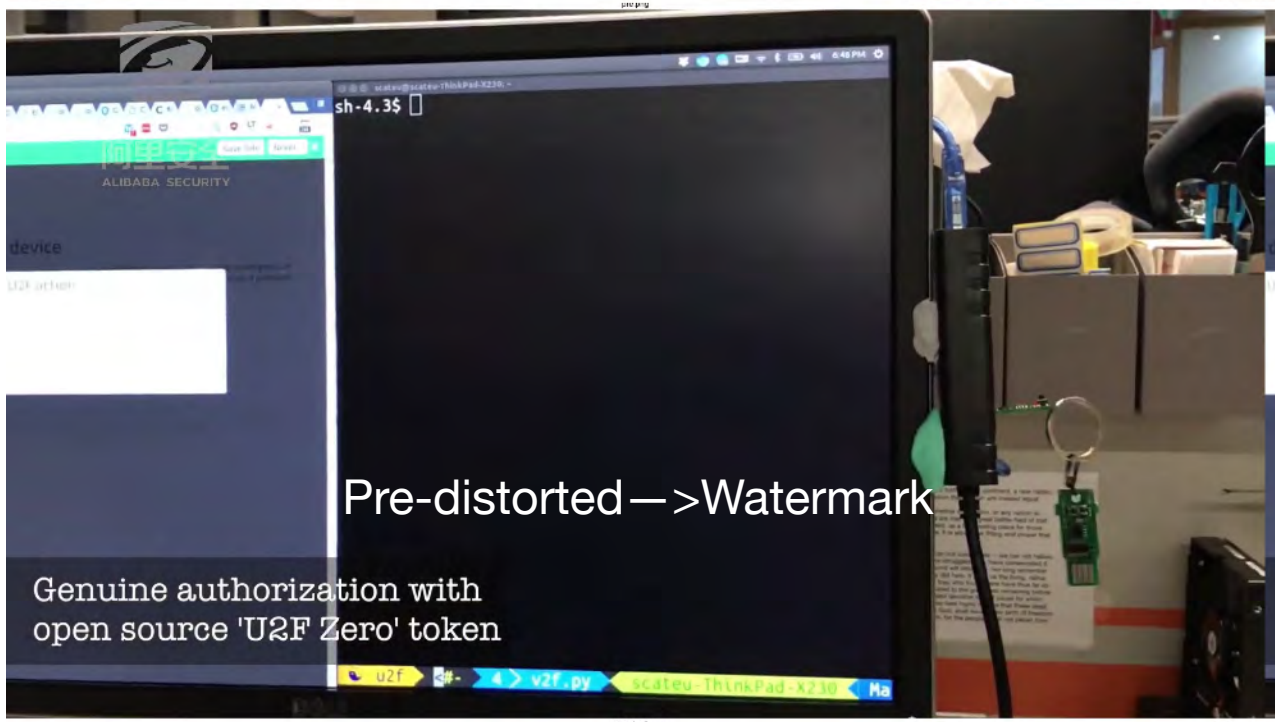
After a few days' coding and building…

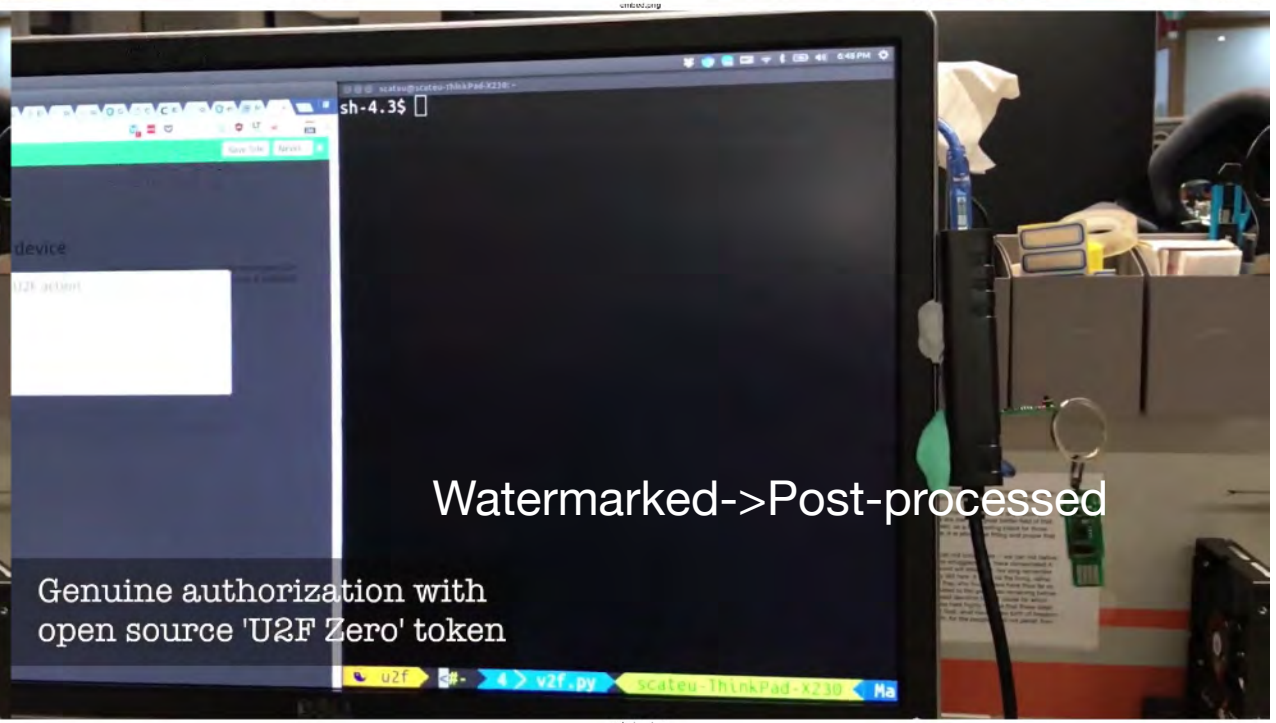
We released the project ™Block

```
# Build tmblock
$ cmake -DHALIDE_DISTRIB_DIR=path/to/halide -DHALIDE_DISTRIB_USE_STATIC_LIBRARY=TRUE ..


# Build ffmpeg with tmblock
$ ./configure --enable-tmblock --extra-cflags="-I$HOME/Develop/TMBlock/libtmblock/include/" -extra-ldflags="-L$HOME/Develop/TMBlock/libtmbloc
build/"
$ make -j16

# If you are using ArchLinux: https://aur.archlinux.org/packages/ffmpeg-tmblock-git/
# yaourt -S ffmpeg-tmblock-git

# Run..
$ LD_LIBRARY_PATH=$HOME/Develop/TMBlock/libtmblock/build/ ./ffmpeg_g -i input.mp4 -i logo.png -filter_complex "[0:v][1:v]
tmblock=x=100:y=100:func=pre [out]" -map "[out]" -t 10 -pix_fmt yuv420p -y output.mp4
                        #^^^ supported func: pre / post / embed
```


sh-4.3$
ALIBABA SECURITY
device
Pre-distorted (Anti-watermarked)
Genuine authorization with
open source 'U2F Zero' token


sh-4.3$
ALIBABA SECURITY
device
Watermarked
Genuine authorization with
open source 'U2F Zero' token


sh-4.3$
ALIBABA SECURITY
device
Pre-distorted—>Watermark
Genuine authorization with
open source 'U2F Zero' token


sh-4.3$
device
Watermarked->Post-processed
Genuine authorization with
open source 'U2F Zero' token

pre.png

embed.png

pre-embed.png

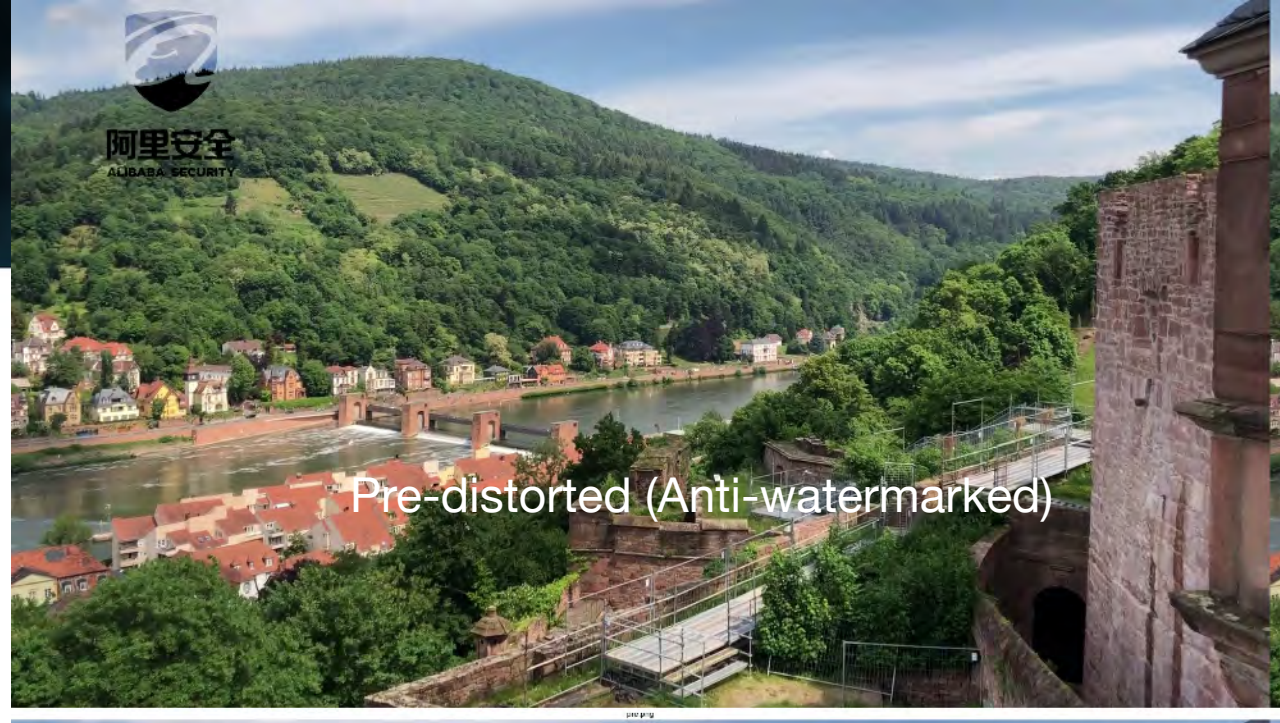
embed-post.png

pre.png

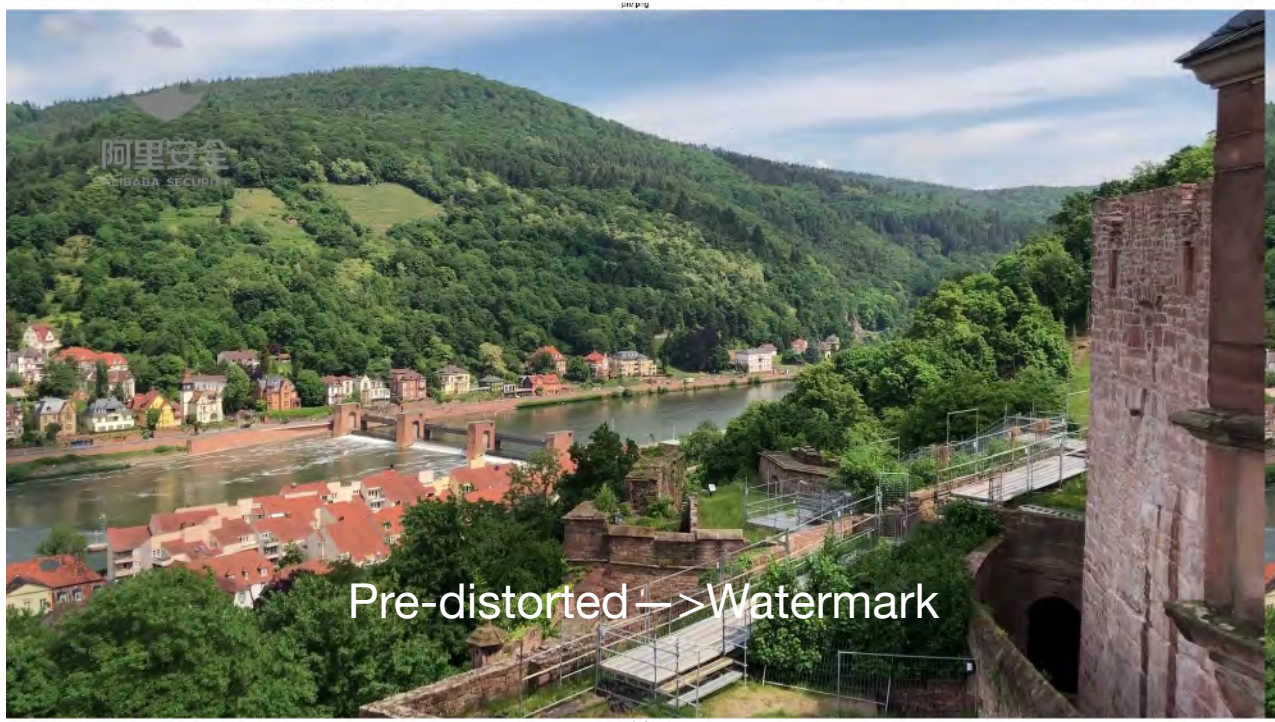
embed.png

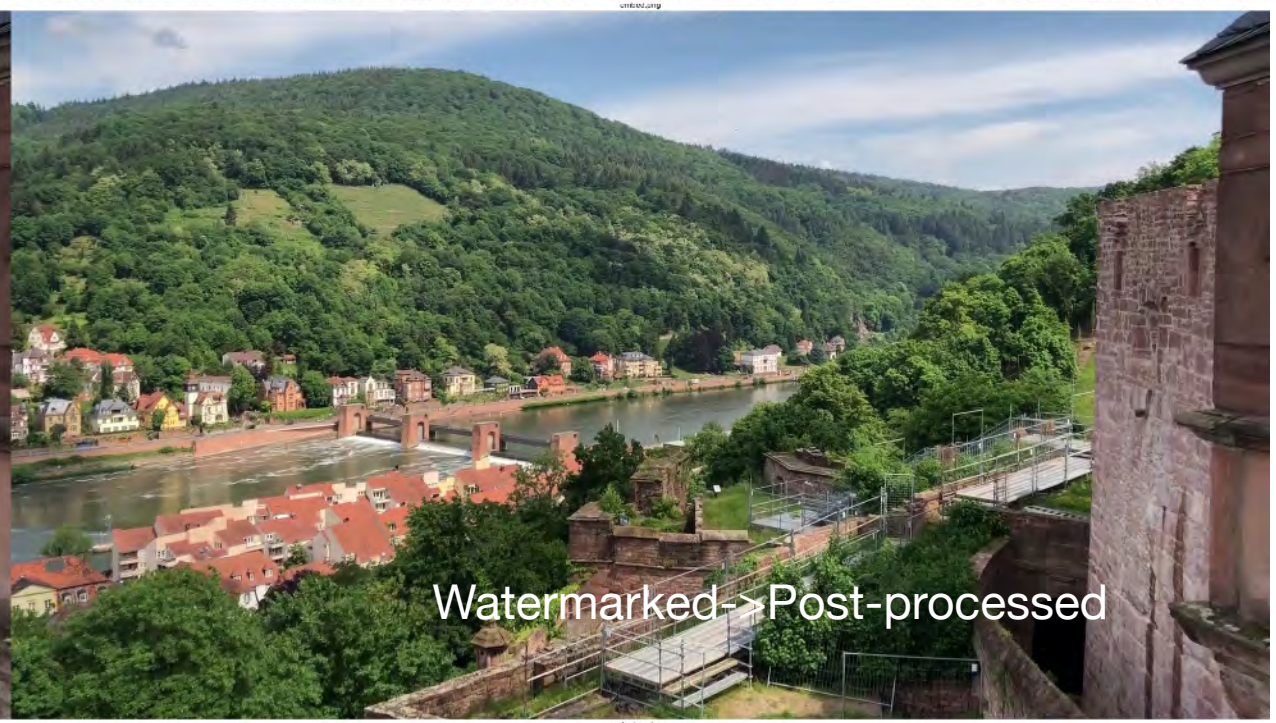
pre-embed.png

embed-post.png


阿里安全
ALIBABA SECURITY
Pre-distorted (Anti-watermarked)

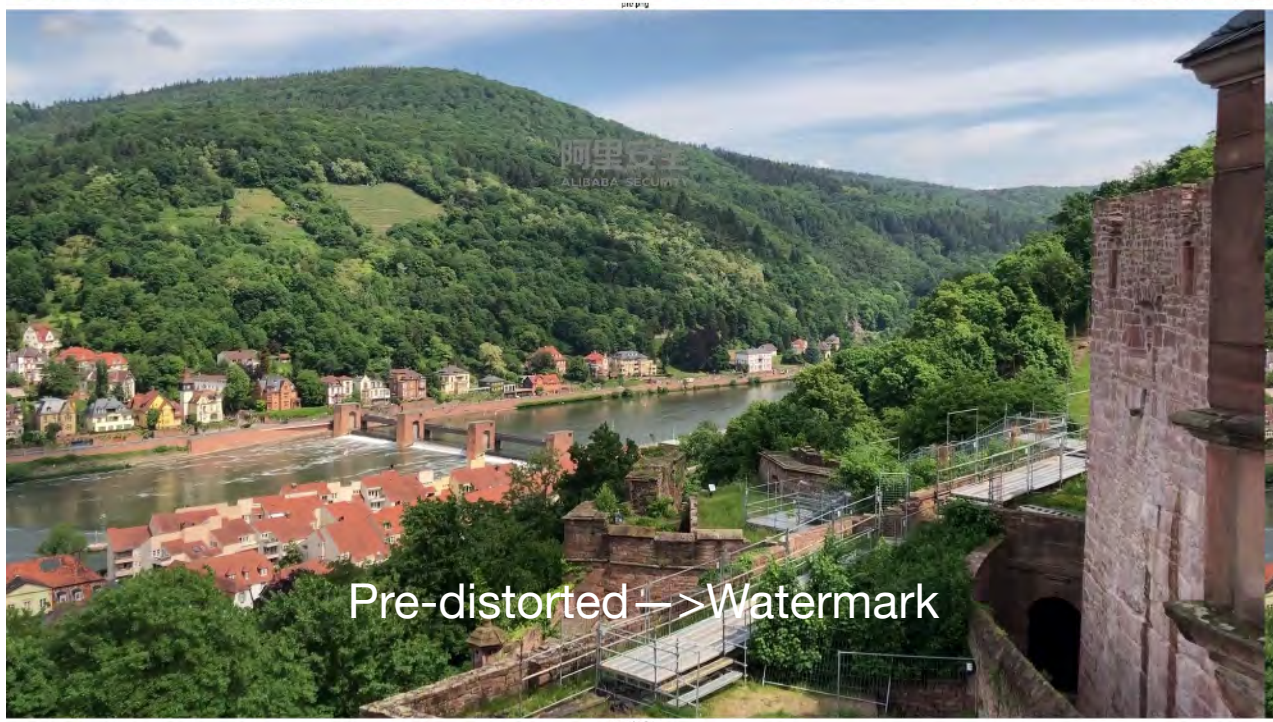

阿里安全
ALIBABA SECURITY
Watermarked

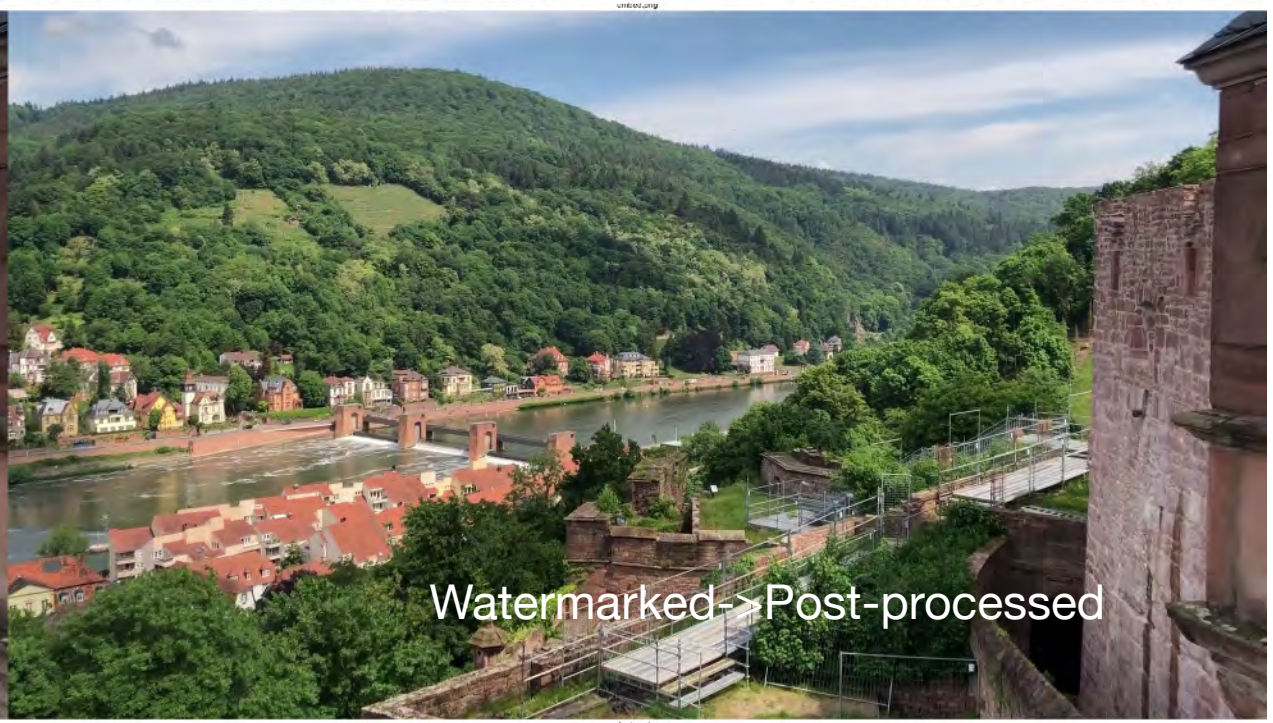

阿里安全
Pre-distorted—>Watermark


Watermarked->Post-processed


阿里安全
ALIBABA SECURITY
Pre-distorted (Anti-watermarked)

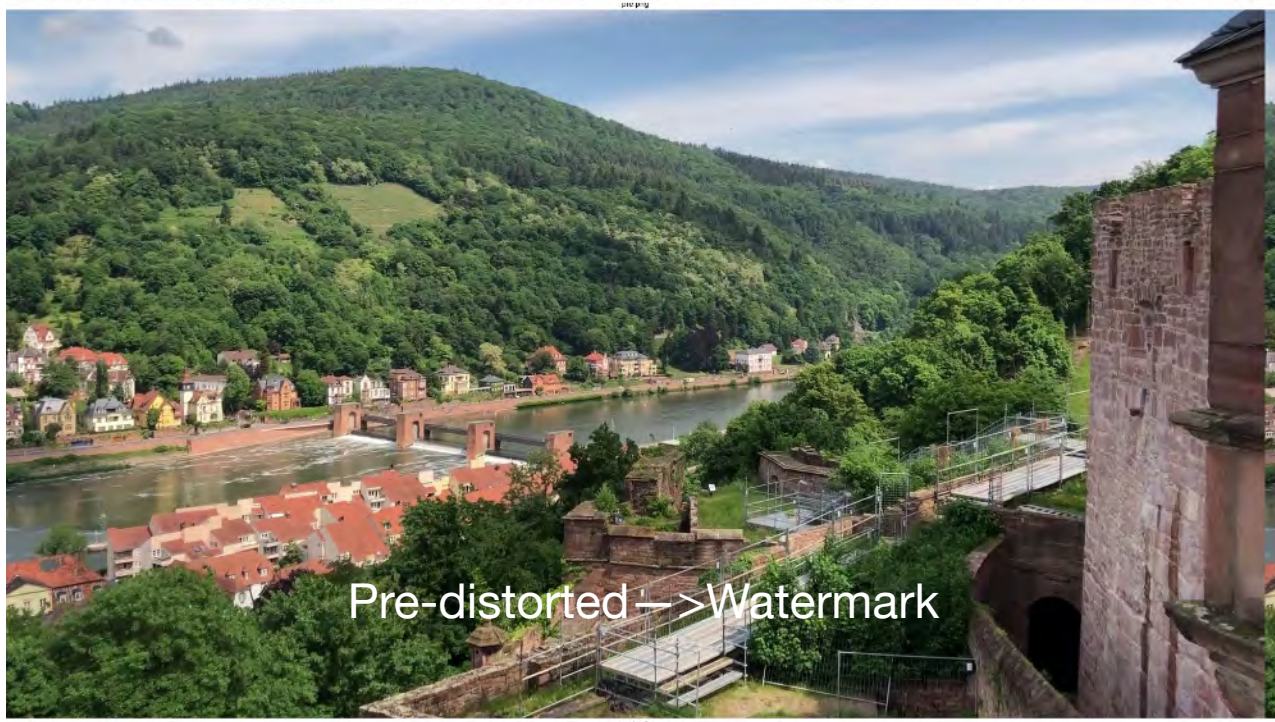

阿里安全
ALIBABA SECURITY
Watermarked

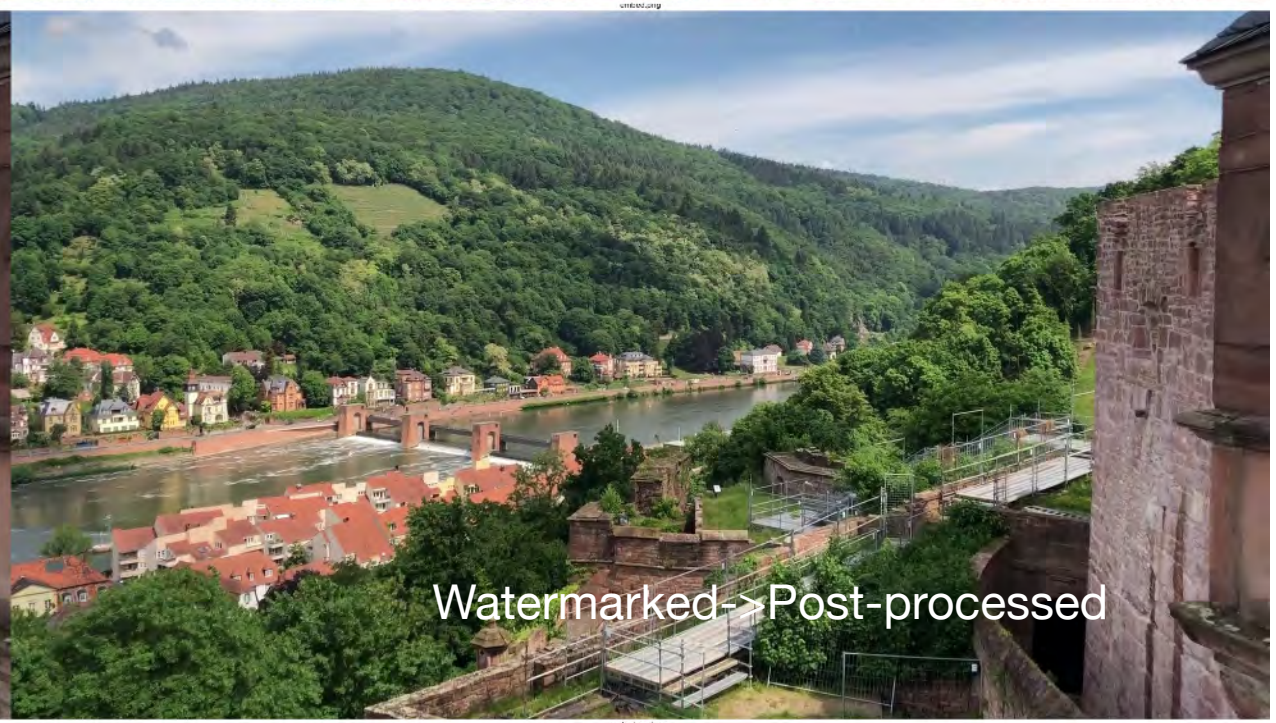

Pre-distorted—>Watermark


Watermarked->Post-processed


Pre-distorted (Anti-watermarked)


Watermarked


Pre-distorted—>Watermark


Watermarked->Post-processed


Pre-distorted (Anti-watermarked)


Watermarked


Pre-distorted—>Watermark


Watermarked->Post-processed

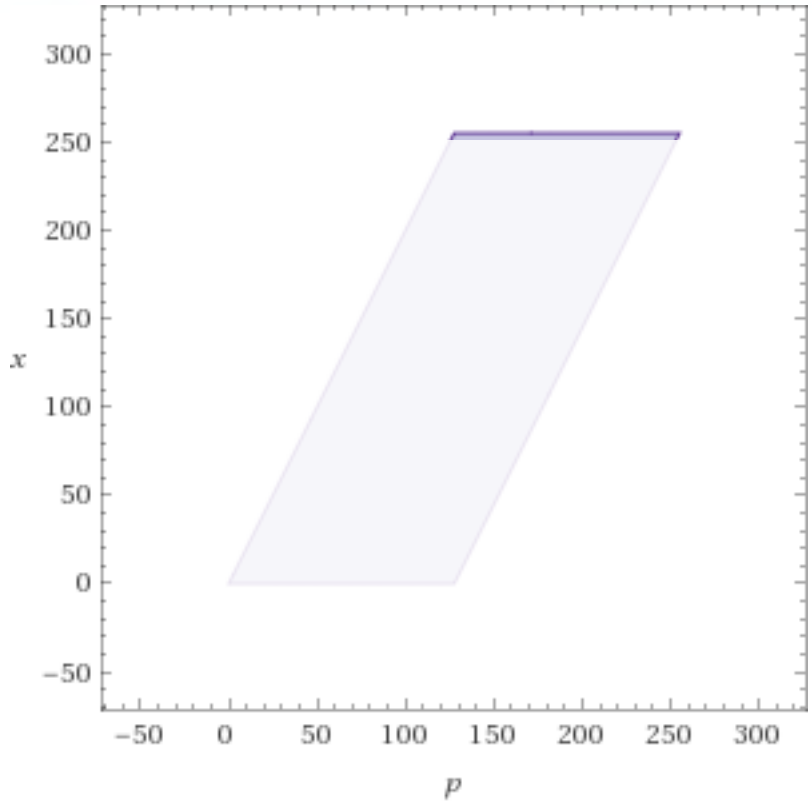
Why?

Recoverable Range

- White: (255,255,255) α=0.5
- Lossless Recoverable Range:
  - (127-255, 127-255, 127-255)

Range of 100% Recovery

90% Recovery

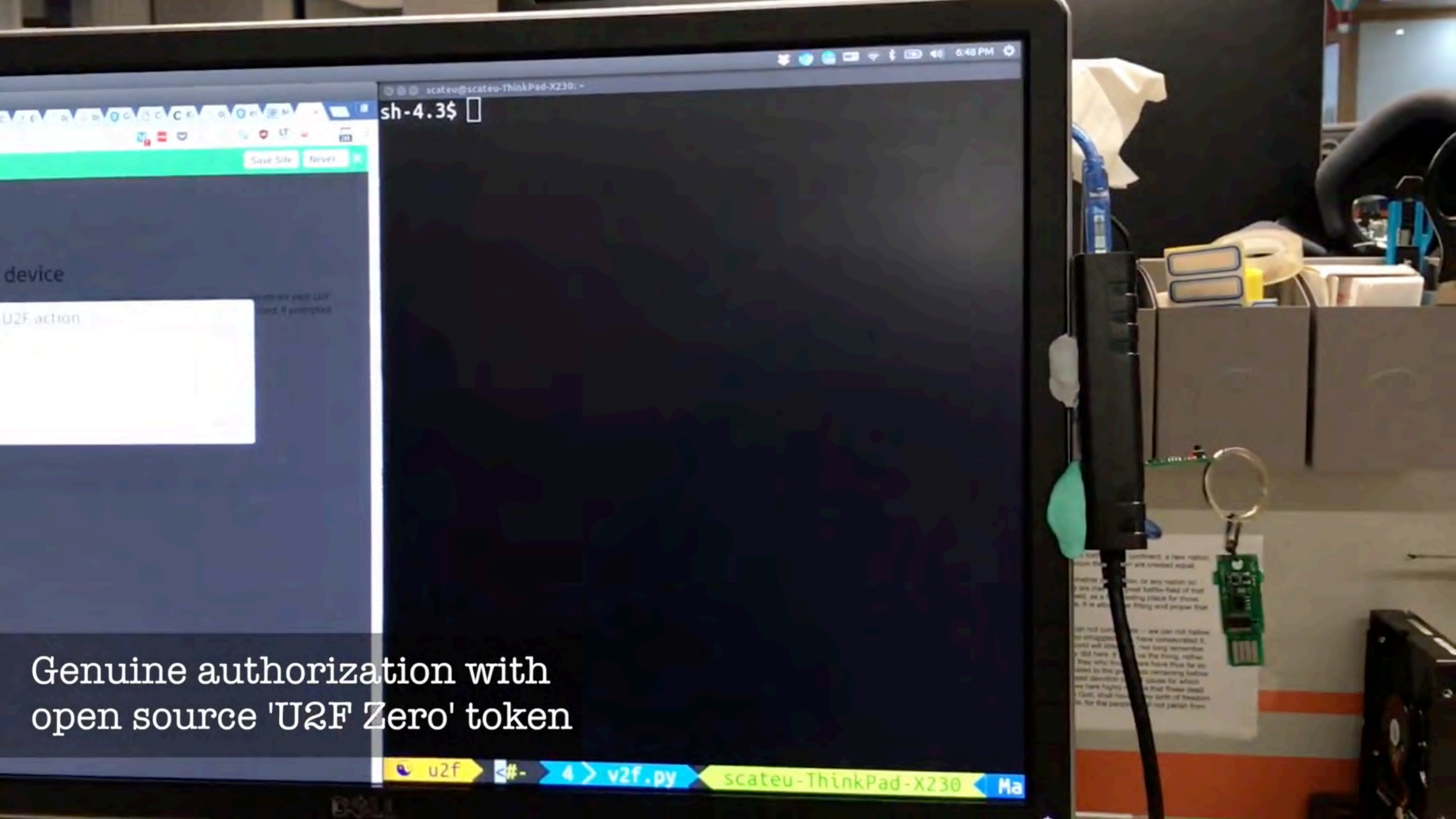


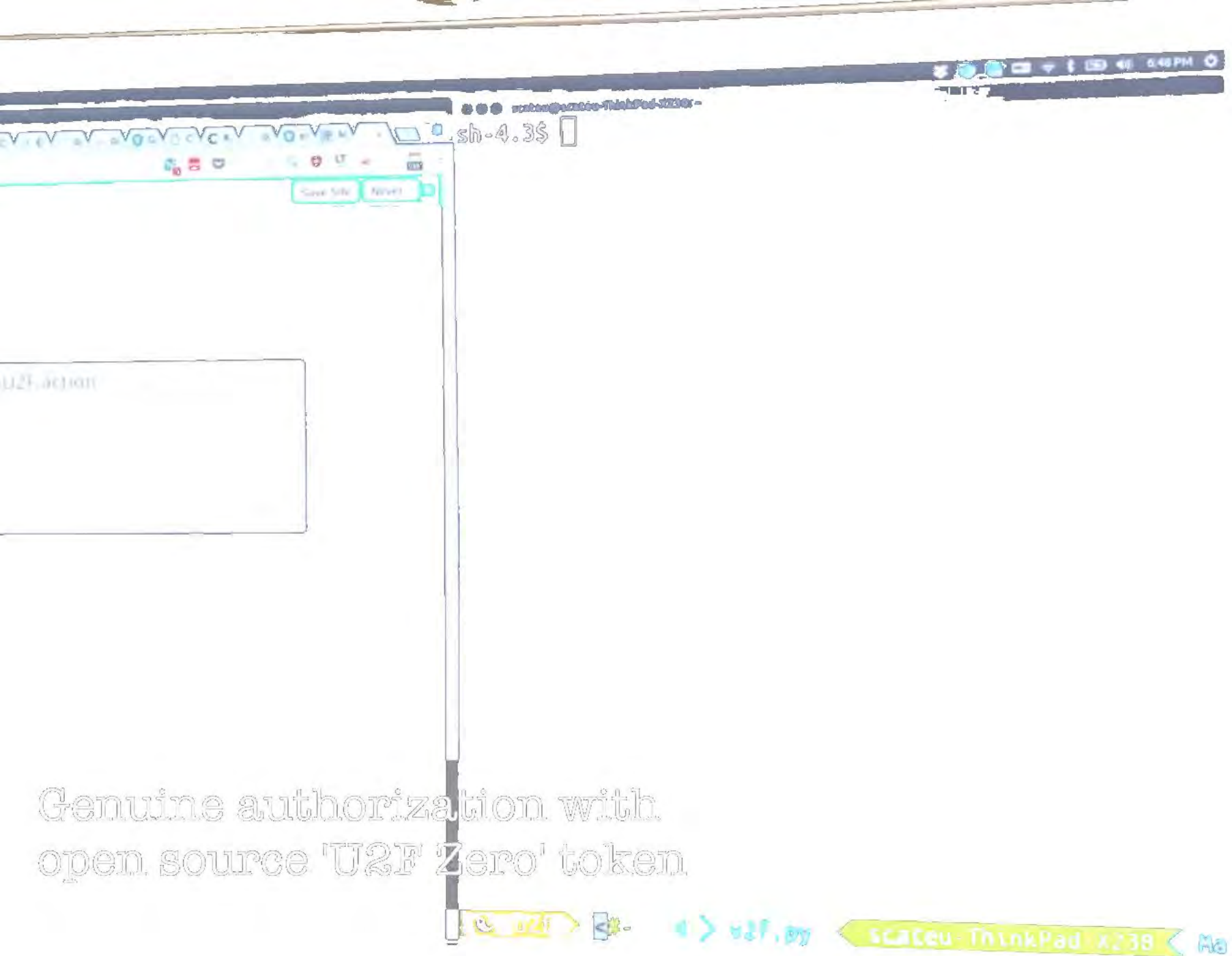
*Human Eyes

Recoverable Area


《苏堤漫步》
知音雅集民乐团
2017年12月24日
清华大学


《苏堤漫步》
知音雅集民乐团
2017年12月24日
清华大学


6:48 PM
scateu@scateu-ThinkPad-X230: ~
sh-4.3$
Save Site
Never
device
U2F action
Genuine authorization with
open source 'U2F Zero' token
u2f
4
v2f.py
scateu-ThinkPad-X230


sh-4.3$
Genuine authorization with
open source 'U2F Zero' token

Record under sky.™

Demo: Bilibili.tv

- bilibili.tv(NYSE: BILI):
- One of biggest video providers in China
- Still picture watermark on top-left
  - A little trouble: Blurred edge
- Solving PNG Codec Issue
- Watermark cancelled completely with method #1: pre-distortion

PRE-01-origin.png

PRE-02-pre-anti-watermarked.png

PRE-03-normal-watermarked.png

PRE-04-final.png

POST-03-output.png


PRE-01-origin.png

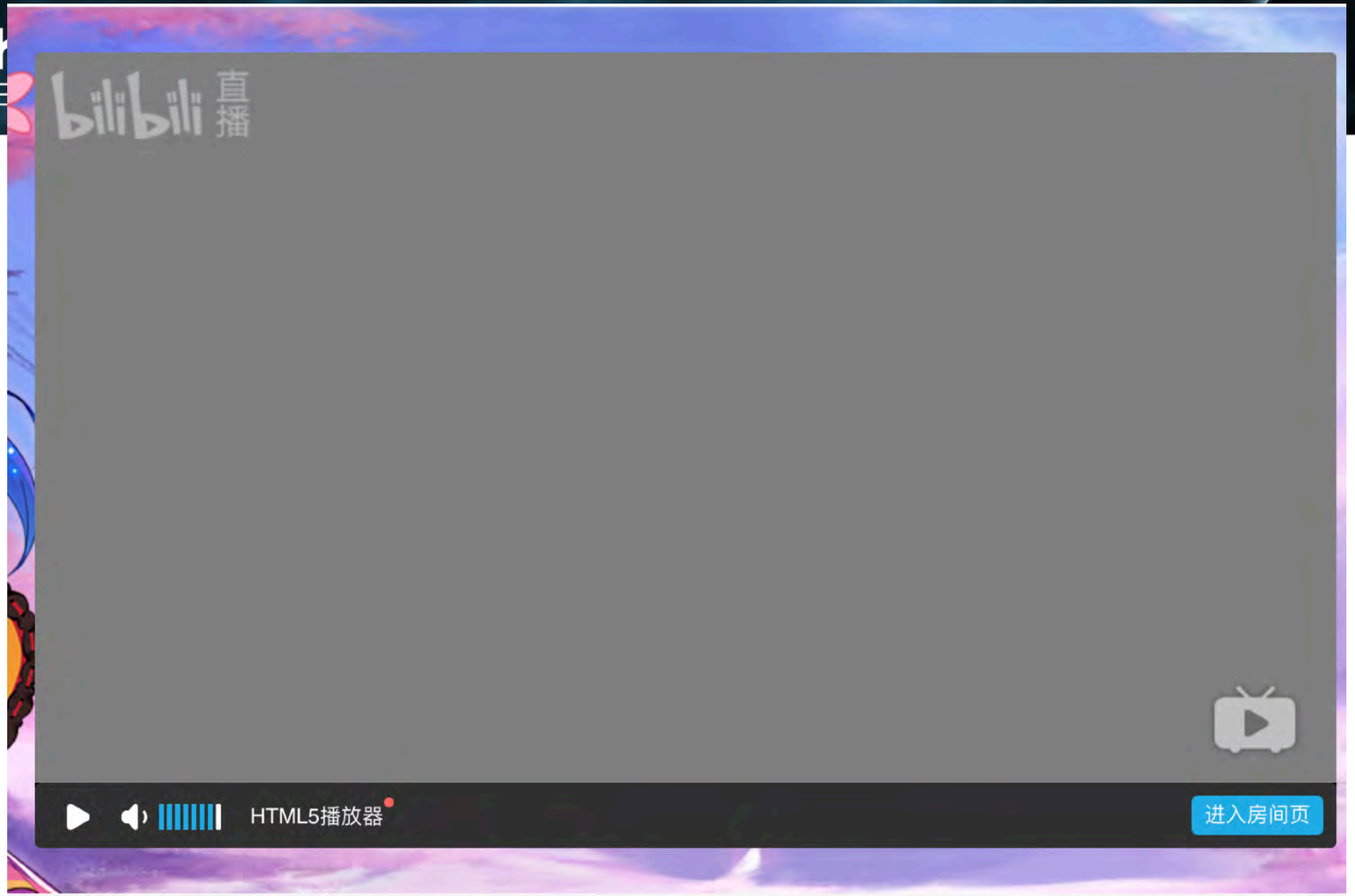

PRE-02-pre-anti-watermarked.png


PRE-03-normal-watermarked.png

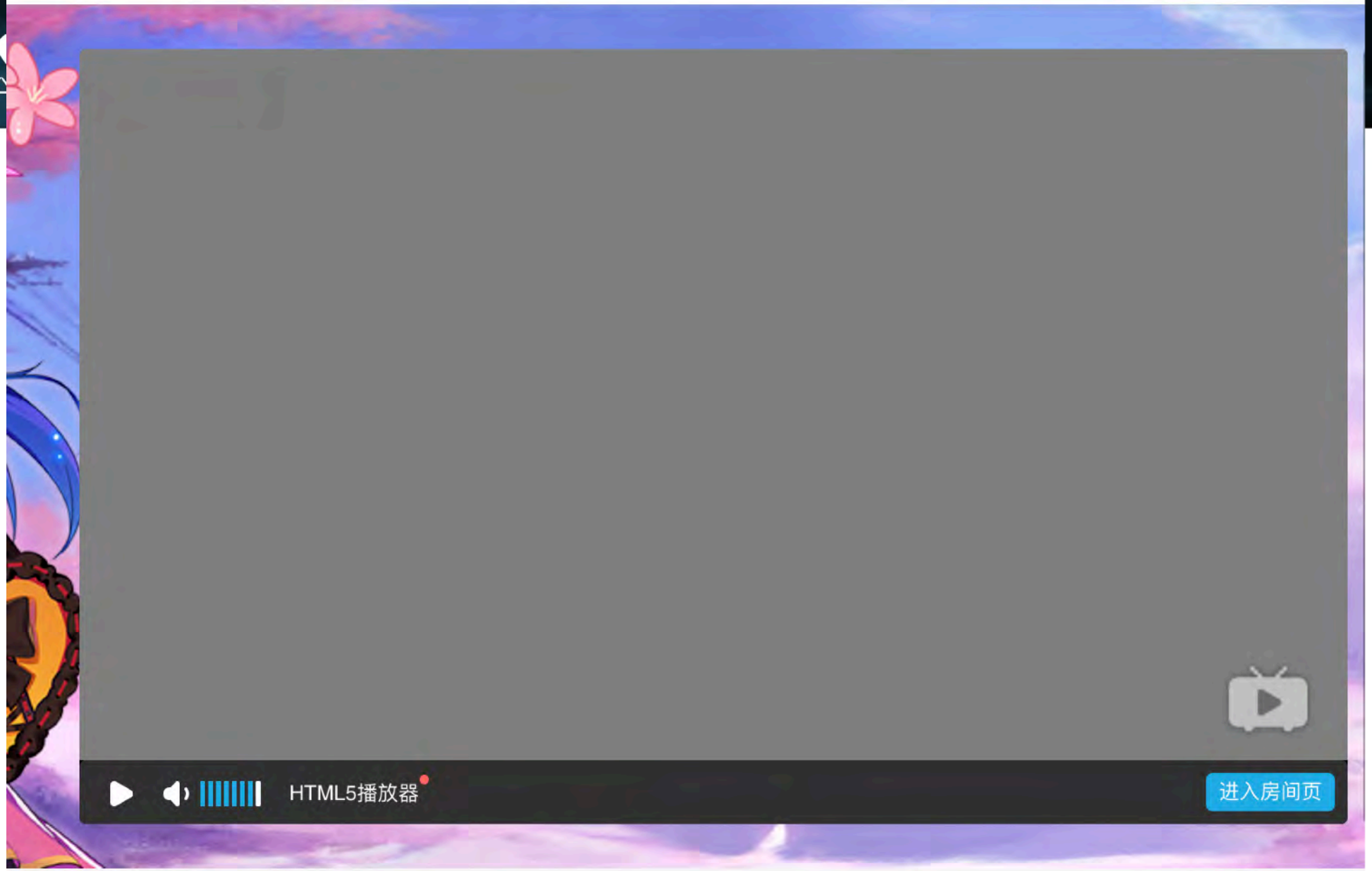

PRE-04-final.png


POST-03-output.png

lucky zone.


bilibili 直播
HTML5播放器
进入房间页

Pre-distorted Media

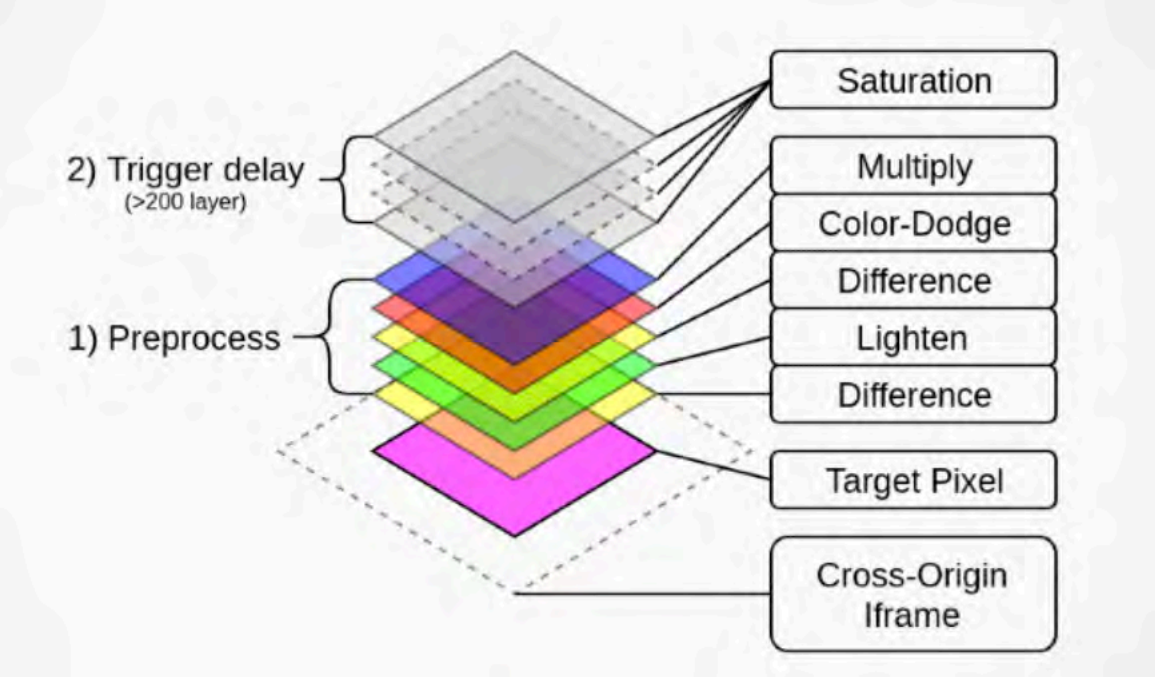


Problem: Difference between PNG parsing libraries for edge blurring

Other ways…

- Transform
  - to transform one watermark into another provider's.
- Code Rate Jitter
  - filling with high-frequency components
  - force video codec to reduce the code rate near watermark.
  - eg: World Cup
- Frame Squeezing
  - sacrificing some resolution by squeezing screen
  - restoring with a user-defined javascript or iFrame to bypass watermark
  - on your blog

- CSS is So Overpowered It Can Deanonymize Facebook Users

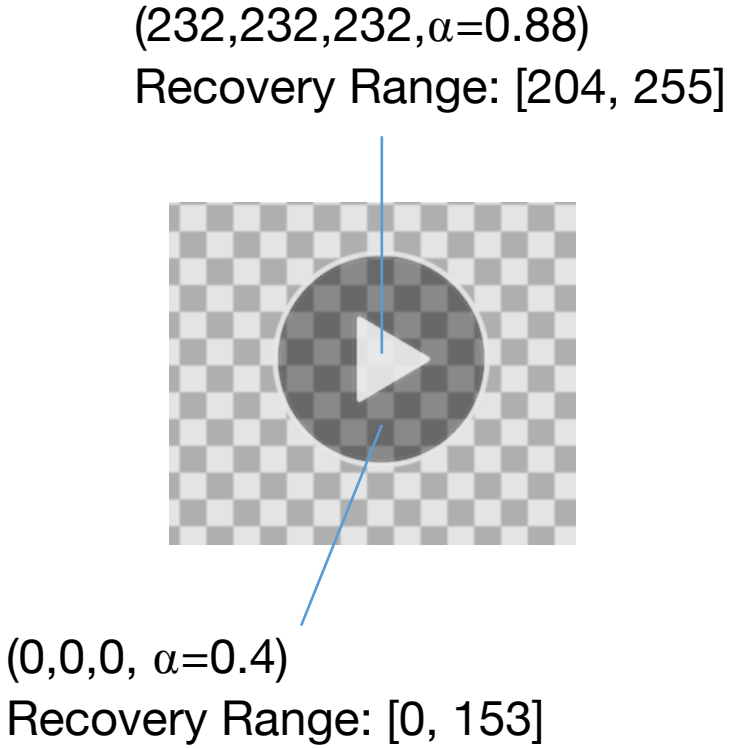


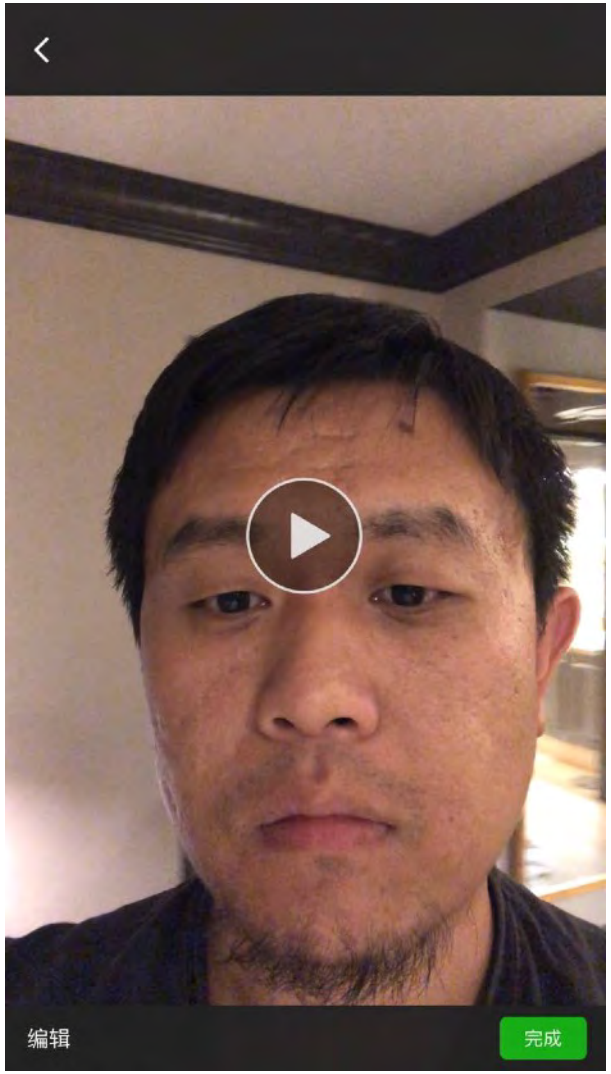
Some of the recent additions to the Cascading Style Sheets (CSS) web standard are so powerful that a security researcher has abused them to deanonymize visitors to a demo site and reveal their Facebook usernames, avatars, and if they liked a particular web page of Facebook. Information leaked via this attack could aid some advertisers link IP addresses or advertising profiles to real-life persons, posing a serious threat to a user's online privacy.
The leak isn't specific to Facebook but affects all sites which allow their content to be embedded on other web pages via iframes.

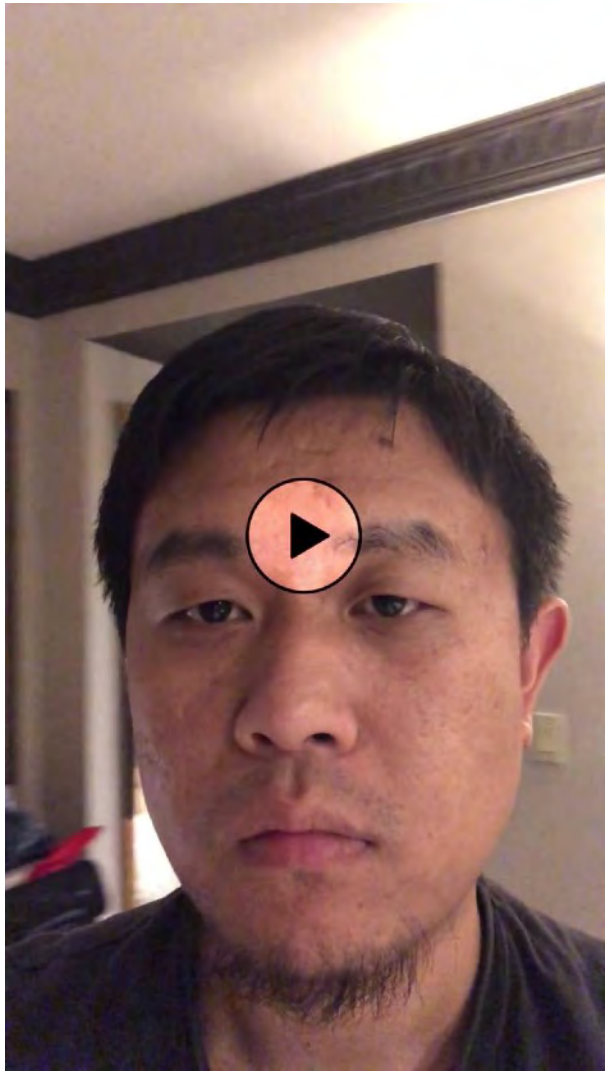
Interesting new research surface.

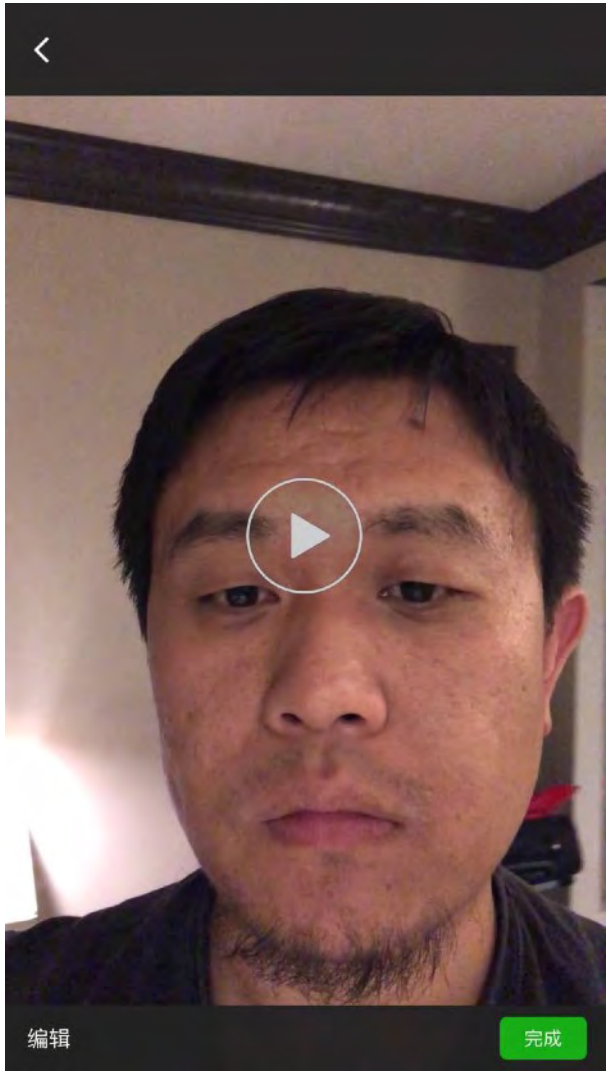
Countermeasure (for the evil side)

- Edge Blurring
- Combine transparent part with non-transparent part
- Combine multiple α to make recovery range unavailable
  - to provide a better user experience
- Position Randomize
  - Breathing Style
  - Dynamic Watermark
- Anti-Anti-Watermark




编辑
完成




编辑
完成

Conclusion & Future Work

- What we have done so far:
  - Theoretical estimation and experiment results of
    - pre-process method
    - post-process method
  - Realtime processing framework
    - Debug testbed
  - Running code can be found at https://github.com/tmblock/
- Future Work
  - Auto detect and extract from arbitrary video sequence
  - With AI:
    - Enhanced image recovery with GAN
    - Recovery of loss caused by opaque watermarks
  - Integrated the algorithm into USB Camera with FPGA
    - On the fly

Hopefully..

- This work will be able to push video content providers to stop using ugly and annoying visible watermark.
  - User experience matters.
- To protect copyright,
  - please deploy DRM or invisible watermark
  - or be a man
    - CC or Open Source/Open Access

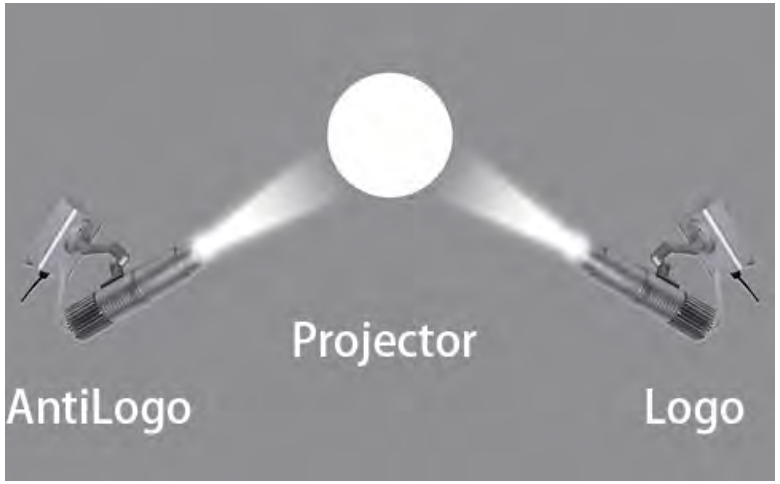
This is a hack

of the people, by the people, and for the people.

The watermark shall perish from the earth.

Reference

- https://blog.huiyiqun.me/2016/10/28/livestream-your-desktop-with-nginx-rtmp-module.html
- https://github.com/tmblock
- https://halide-lang.org/
- https://www.nmm-hd.org/newbbs/viewtopic.php?t=649



Thank you.

Q&A