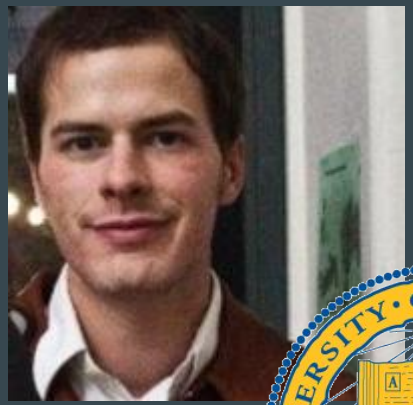
# QCT-Parse

...

Automated QCTools Analysis For a Better and Brighter Tomorrow

# QCT-Parse

Brendan Coates



UCSB Special Research Collections

Morgan Oscar Morel

George Blood Audio/Video/Film/Data

# QCT-Parse

...

Automated QCTools Analysis For a Better and Brighter Tomorrow

# Just a normal day in the QC office


# QCTools

NATIONAL ENDOWMENT FOR THE HUMANITIES

QC TOOLS

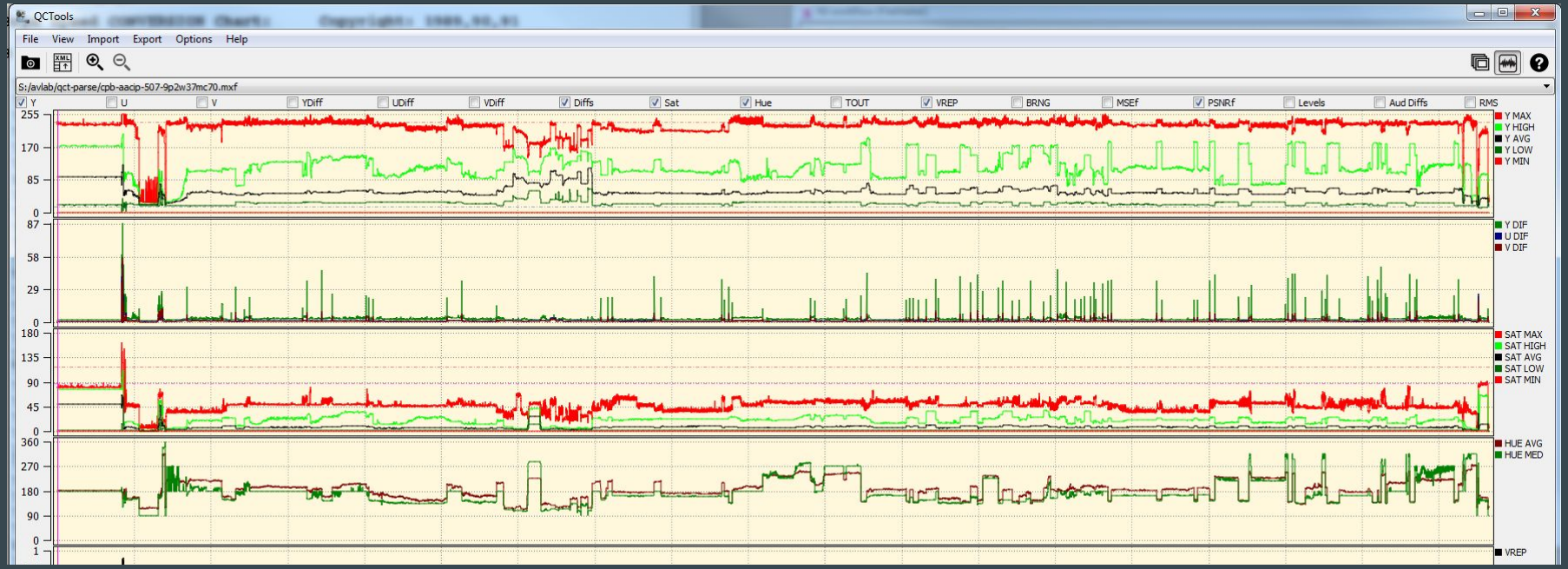
Knight Foundation

bavc

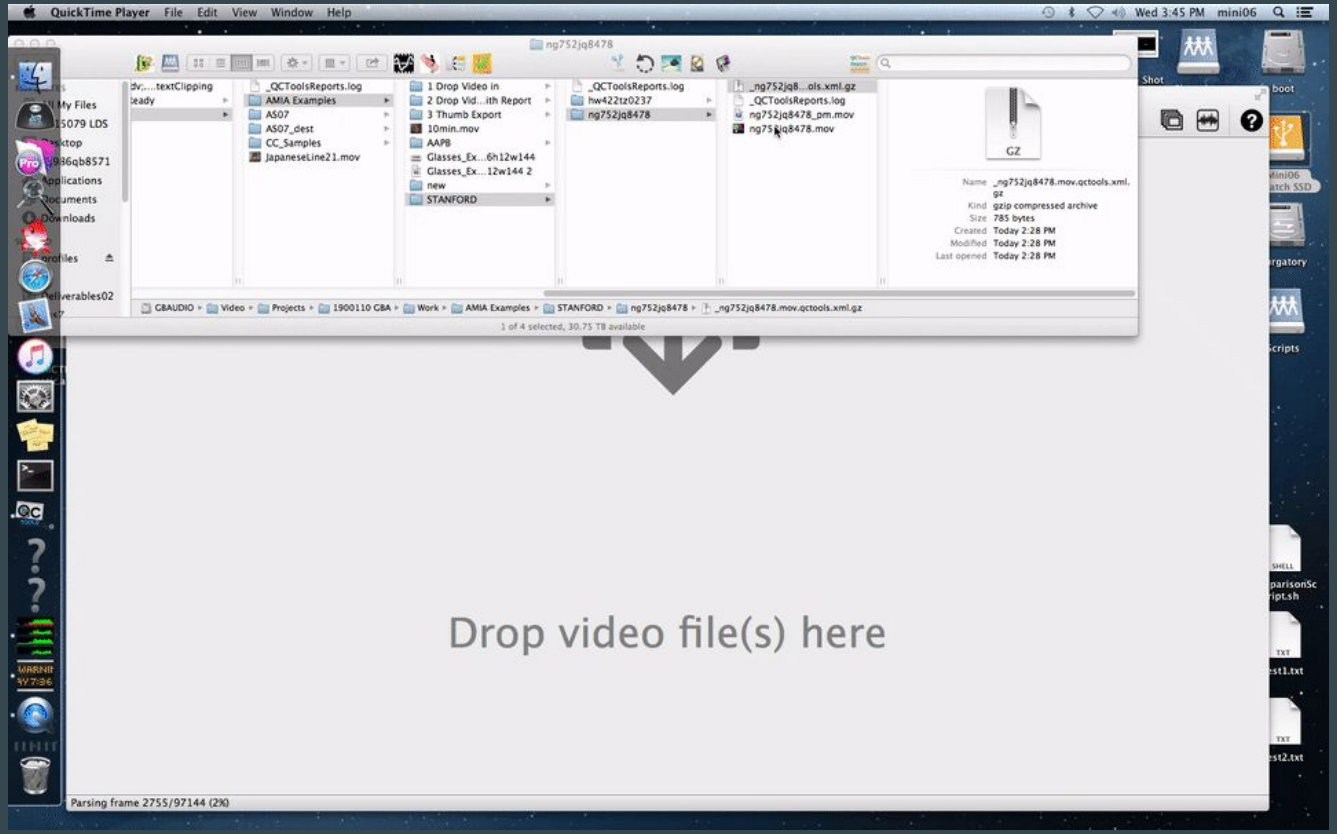
BAY AREA VIDEO COALITION

DANCE HERITAGE COALITION

# What QCTools looks like



# Drop a file in QCTools




x 100

# QCTools Report Generator

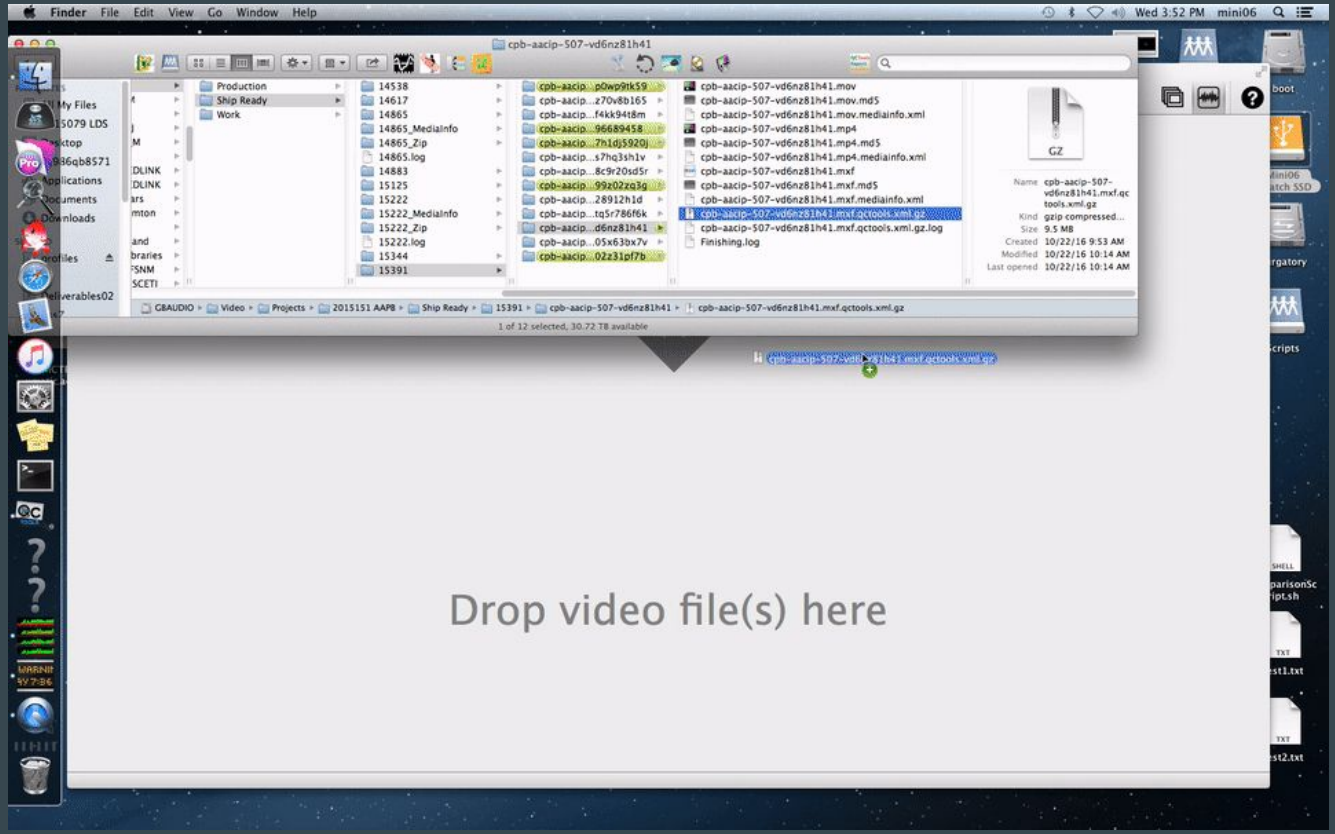
Q: What if we scripted making a QCTools Report?

A1: QCTools backend has a really fancy ffprobe call:

```
ffprobe -loglevel error -f lavfi -i movie='/path/to/video.mkv',signalstats=stat=tout+vrep+brng,cropdetect=reset=1,split[a][b];[a]field=top[a1];[b]field=
bottom[b1],[a1][b1]psnr -show_frames -show_versions -of xml=x=1:q=1 -noprivate /path/to/video.mkv.xml
```

A2: python makeqctoolsreport.py /path/to/video.mkv

# Drop a report into QCTools



# QCTools Report Parser

Q: What if we scripted reading a QCTools Report?

A: github.com/FutureDays/qct-parse

Goals:

1. Speed up QC workflows by making robots find problems
2. Provide extensible i/o for others to build on
3. Uncover some notion of what constitutes “quality video”
    a. Much more on this later

# QCTools Report Parser: Feature Rundown

Find frames beyond the threshold for a single tag (e.g. SATMAX over 181.2)

-we call them “overs” or “unders”

Test just the bars,  or both content and bars

Export thumbnails of frames beyond threshold

Test against every tag ffprobe pops out, 36 in all

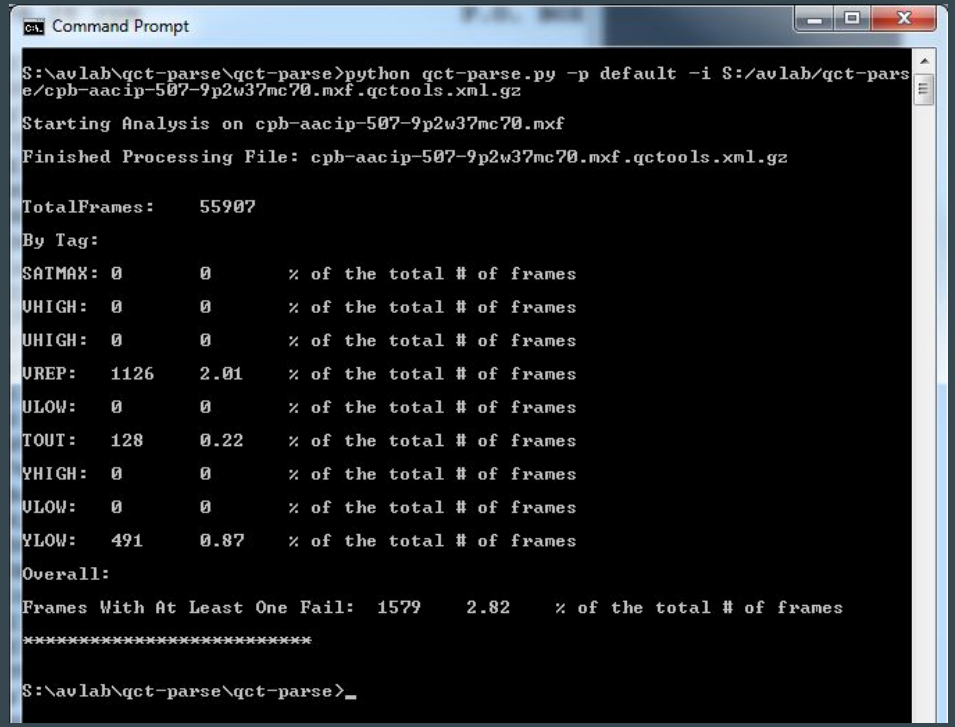
Make your own rules! (we call them “profiles”)

```
#based on qctools docs
[default]
YLOW: 16
YHIGH: 235
ULOW: 16
UHIGH: 235
VLOW: 0
VHIGH: 255
SATMAX: 181.02
TOUT: 0.009
VREP: 0.03

#Higher Tolerance for Peaking
[highTolerance]
YLOW: 10
YMAX: 250
UMIN: 16
UMAX: 250
VMIN: 0
VMAX: 255
SATMAX: 181.02
TOUT: 0.009
VREP: 0.03

#Medium Tolerance for Peaking
[midTolerance]
YLOW: 10
YMAX: 245
UMIN: 16
UMAX: 245
```

# QCTools Report Parser: Output



# QCTools Report Parser: Data Model

```
<frames>
    <frame media_type="audio" stream_index="1" key_frame="1" pkt_pts="0" pkt_pts
        <tag key="lavfi.r128.M" value="-120.691"/>
        <tag key="lavfi.r128.S" value="-120.691"/>
        <tag key="lavfi.r128.I" value="-70.000"/>
        <tag key="lavfi.r128.LRA" value="0.000"/>
        <tag key="lavfi.r128.LRA.low" value="0.000"/>
        <tag key="lavfi.r128.LRA.high" value="0.000"/>
    </frame>
    <frame media_type="video" stream_index="0" key_frame="1" pkt_pts="0" pkt_pts
        <tag key="lavfi.signalstats.YMIN" value="0"/>
        <tag key="lavfi.signalstats.YLOW" value="16"/>
        <tag key="lavfi.signalstats.YAVG" value="88.8546"/>
        <tag key="lavfi.signalstats.YHIGH" value="178"/>
        <tag key="lavfi.signalstats.YMAX" value="250"/>
        <tag key="lavfi.signalstats.UMIN" value="32"/>
        <tag key="lavfi.signalstats.ULOW" value="62"/>
        <tag key="lavfi.signalstats.UAVG" value="129.38"/>
```

# QCTools Report Parser: Data Model

```
<frame media_type="video" stream_index="0" key_frame="1" pkt_pts="0" pkt_pts
    <tag key="lavfi.signalstats.YMIN" value="0"/>
    <tag key="lavfi.signalstats.YLOW" value="16"/>
    <tag key="lavfi.signalstats.YAVG" value="88.8546"/>
    <tag key="lavfi.signalstats.YHIGH" value="178"/>
    <tag key="lavfi.signalstats.YMAX" value="250"/>
```

| YMIN | YLOW | YAVG | YHIGH | YMAX |
|---|---|---|---|---|
| 0 | 16 | 89 | 160 | 233 |

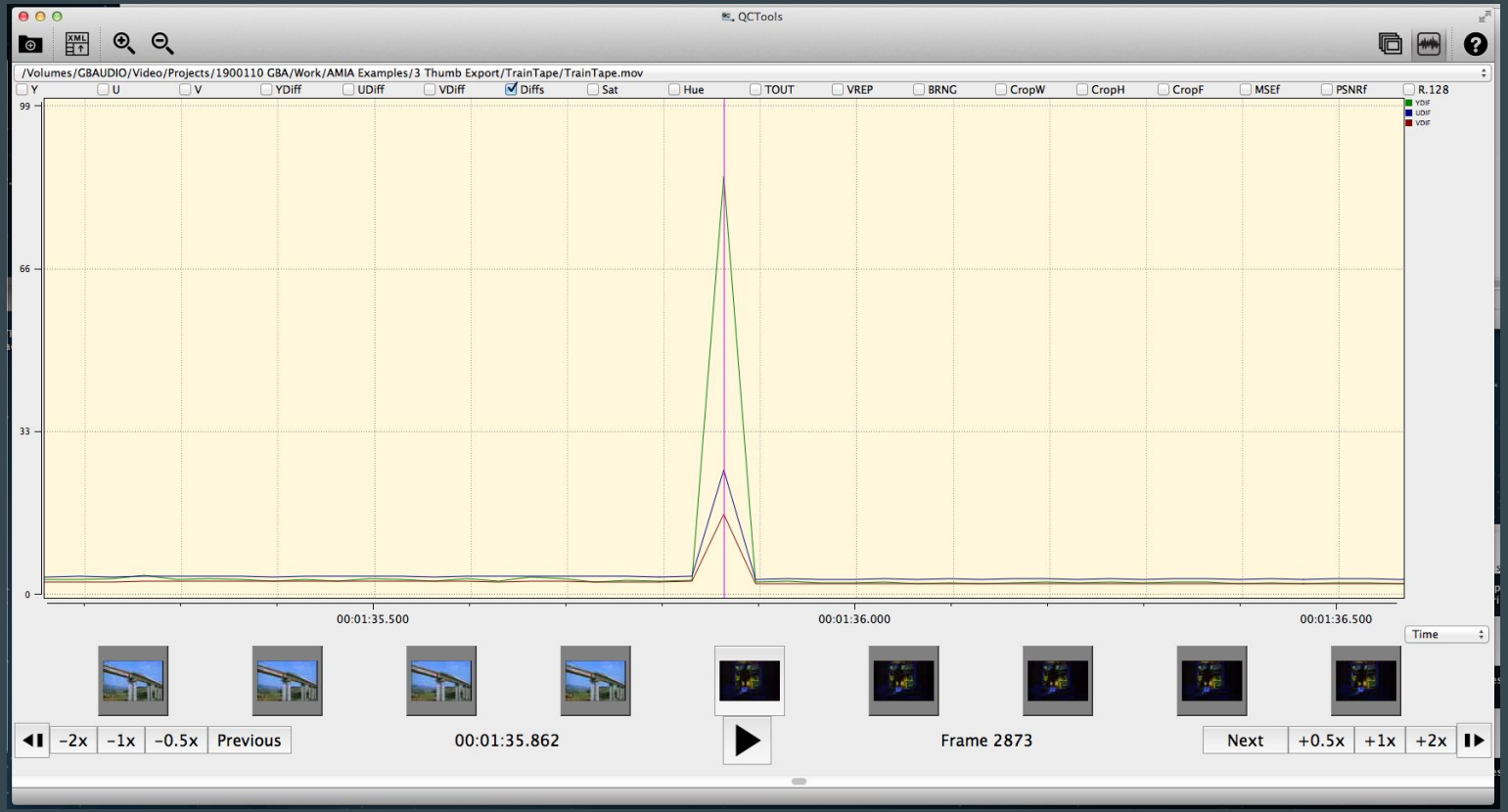
etc. . .

1 Frame

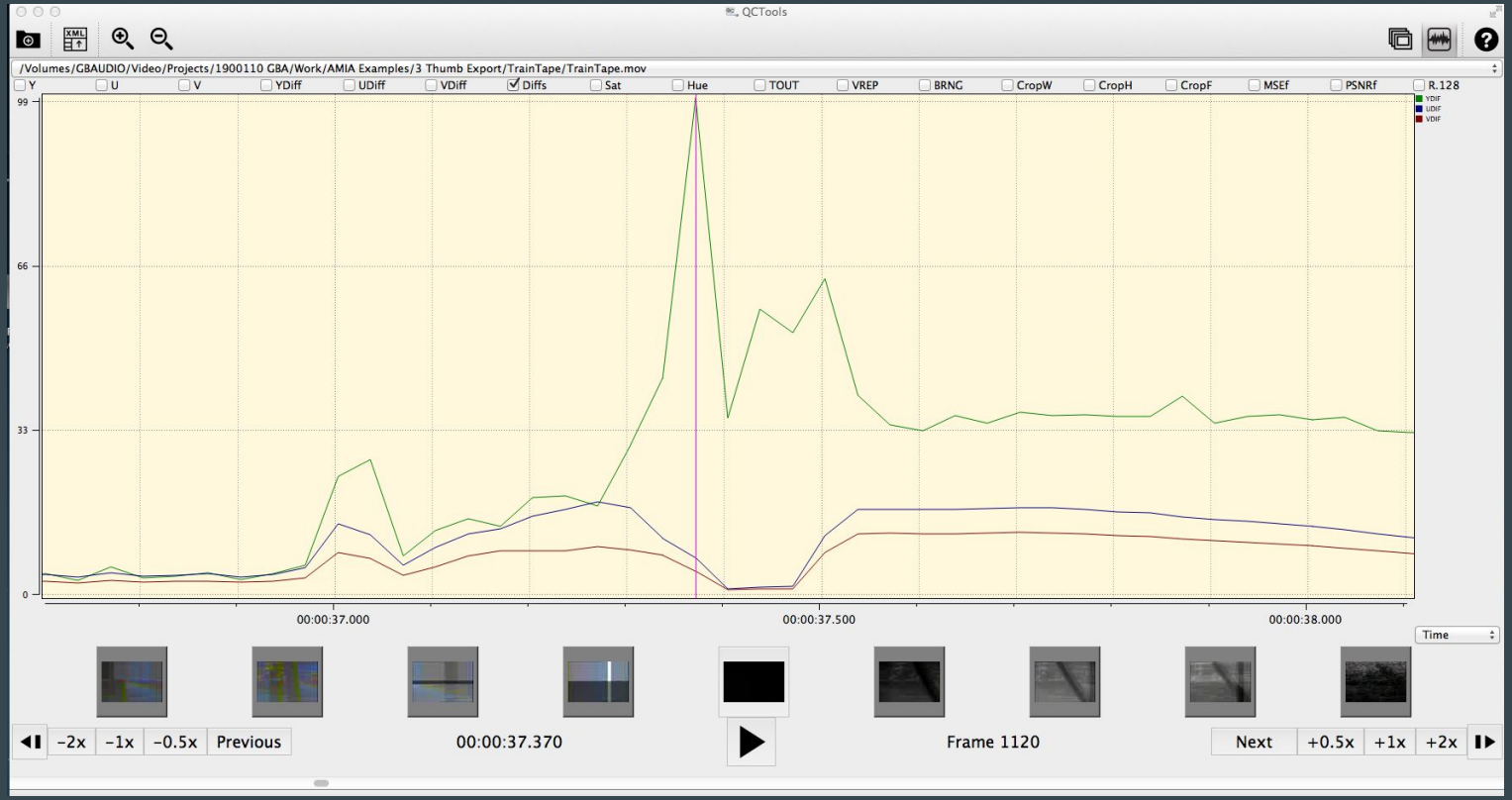
| | N - 2 | N- 1 | N | N + 1 | N + 2 |
|---|---|---|---|---|---|
| YMIN | 0 | 0 | 0 | 0 | 0 |
| YLOW | 11 | 12 | 16 | 14 | 16 |
| YAVG | 82 | 83 | 89 | 90 | 86 |
| YHIGH | 160 | 161 | 160 | 160 | 159 |
| YMAX | 230 | 235 | 233 | 240 | 230 |

Array of Frames

# QCTools Report Parser: Why The Buffer



# QCTools Report Parser: Why The Buffer

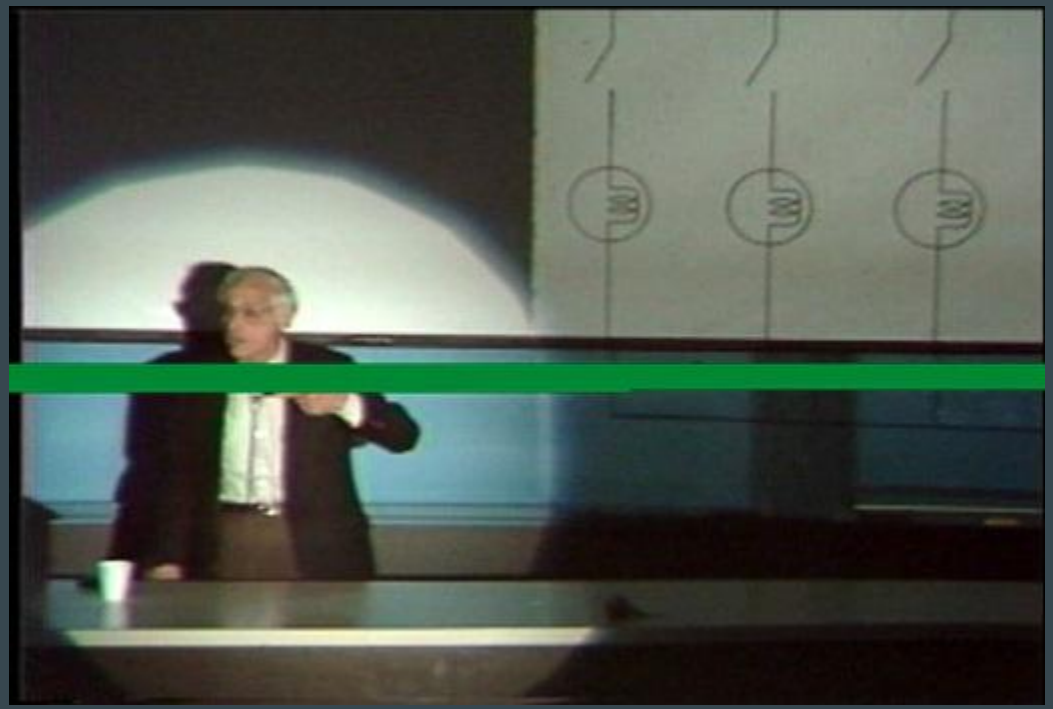


# Using QCTools Report Parser

# QCTools Report Parser: Batch Processing

# QCTools Report Parser: Batch Processing

# QCTools Report Parser: Batch Processing



# QCTools Report Parser: Batch Processing

# QCTools Report Parser: Batch Processing

```
find /Volumes/MII_Tapes -name “*xml.gz”
-exec python qct-parse.py -i {} -t
SATMAX -o 130 -te -ted 100 -tep
/Volumes/MII_Tapes/BadFrames -q \;
```

# QCTools Report Parser: Batch Processing

```
find /Volumes/MII_Tapes -name “*xml.gz”
-exec python qct-parse.py -i {} -t
SATMAX -o 130 -te -ted 100 -tep
/Volumes/MII_Tapes/BadFrames -q \;
```

# QCTools Report Parser: Batch Processing

```
find /Volumes/MII_Tapes -name “*xml.gz”
-exec python qct-parse.py -i {} -t
SATMAX -o 130 -te -ted 100 -tep
/Volumes/MII_Tapes/BadFrames -q \;
```

# QCTools Report Parser: Batch Processing

```
find /Volumes/MII_Tapes -name “*xml.gz”
-exec python qct-parse.py -i {} -t
SATMAX -o 130 -te -ted 100 -tep
/Volumes/MII_Tapes/BadFrames -q \;
```

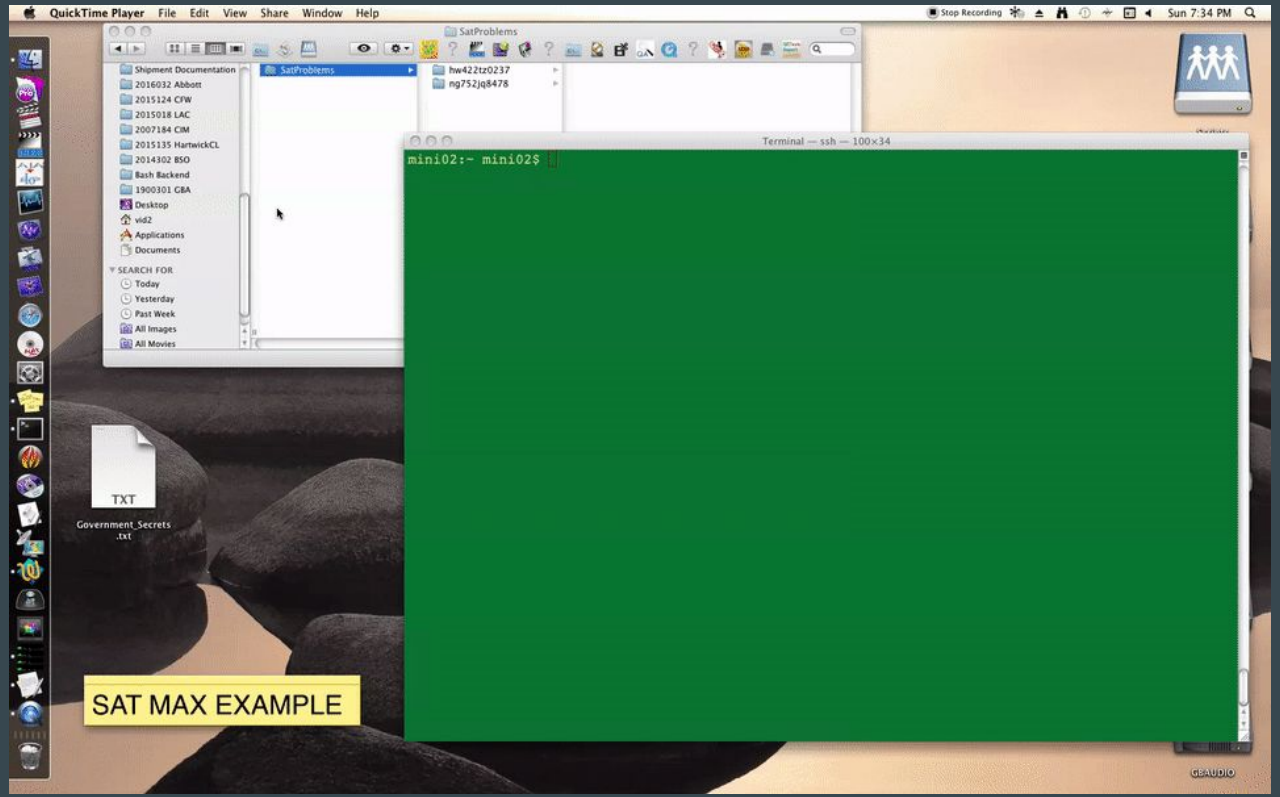
# QCTools Report Parser: Batch Processing

```
find /Volumes/MII_Tapes -name “*xml.gz”
-exec python qct-parse.py -i {} -t
SATMAX -o 130 -te -ted 100 -tep
/Volumes/MII_Tapes/BadFrames -q \;
```

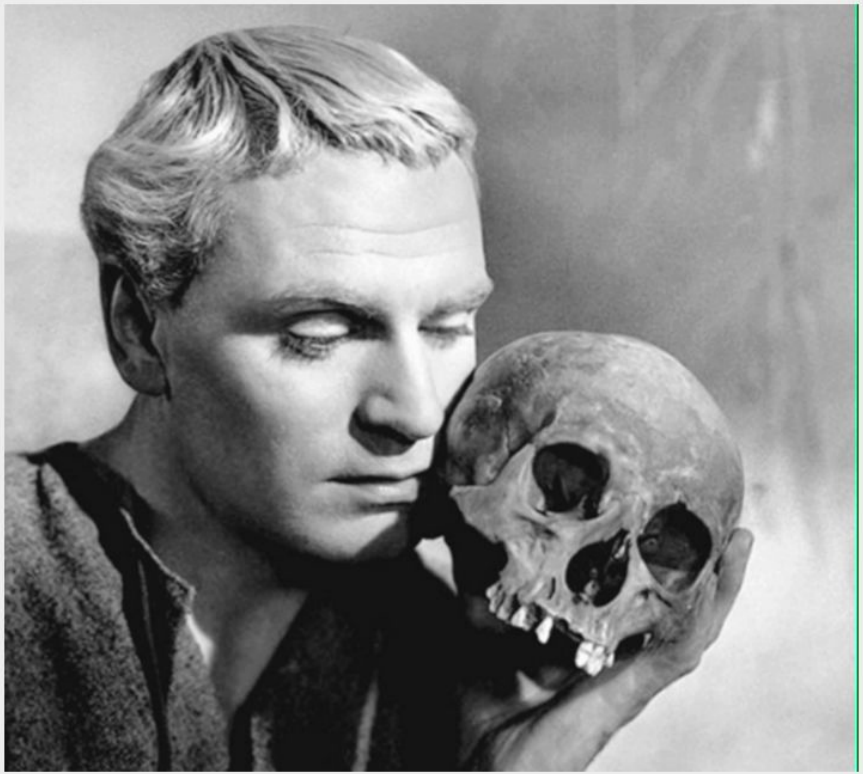
# QCTools Report Parser: Batch Processing

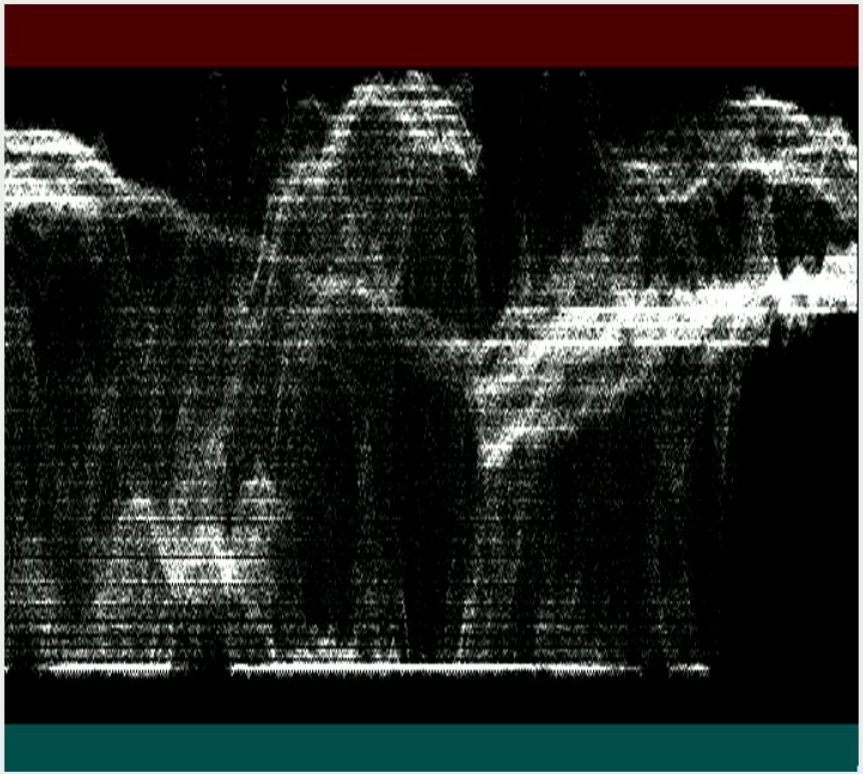
```
find /Volumes/MII_Tapes -name “*xml.gz”
-exec python qct-parse.py -i {} -t
SATMAX -o 130 -te -ted 100 -tep
/Volumes/MII_Tapes/BadFrames -q \;
```

# QCTools Report Parser: Batch Processing



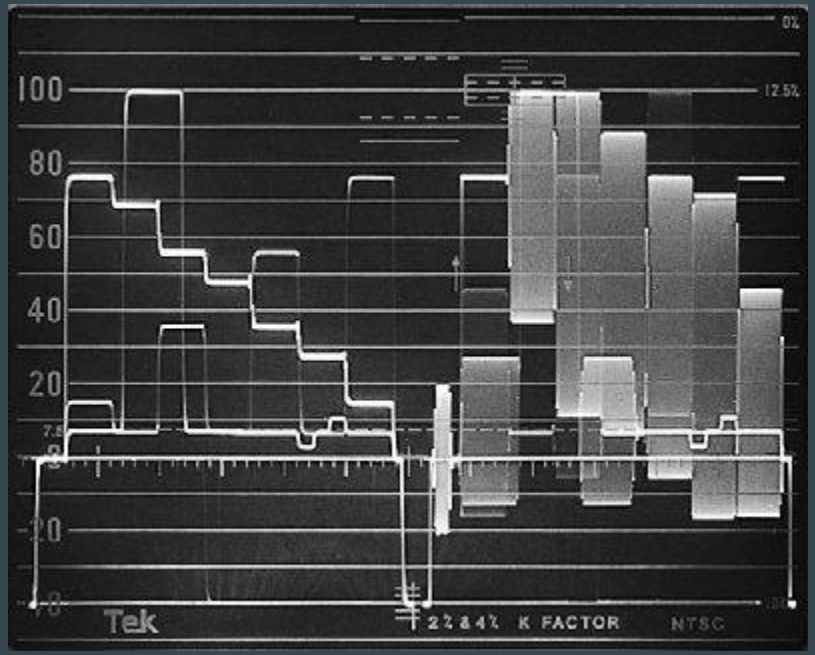
# Overs: To be or not to be (over)





# Overs: To be or not to be (over)

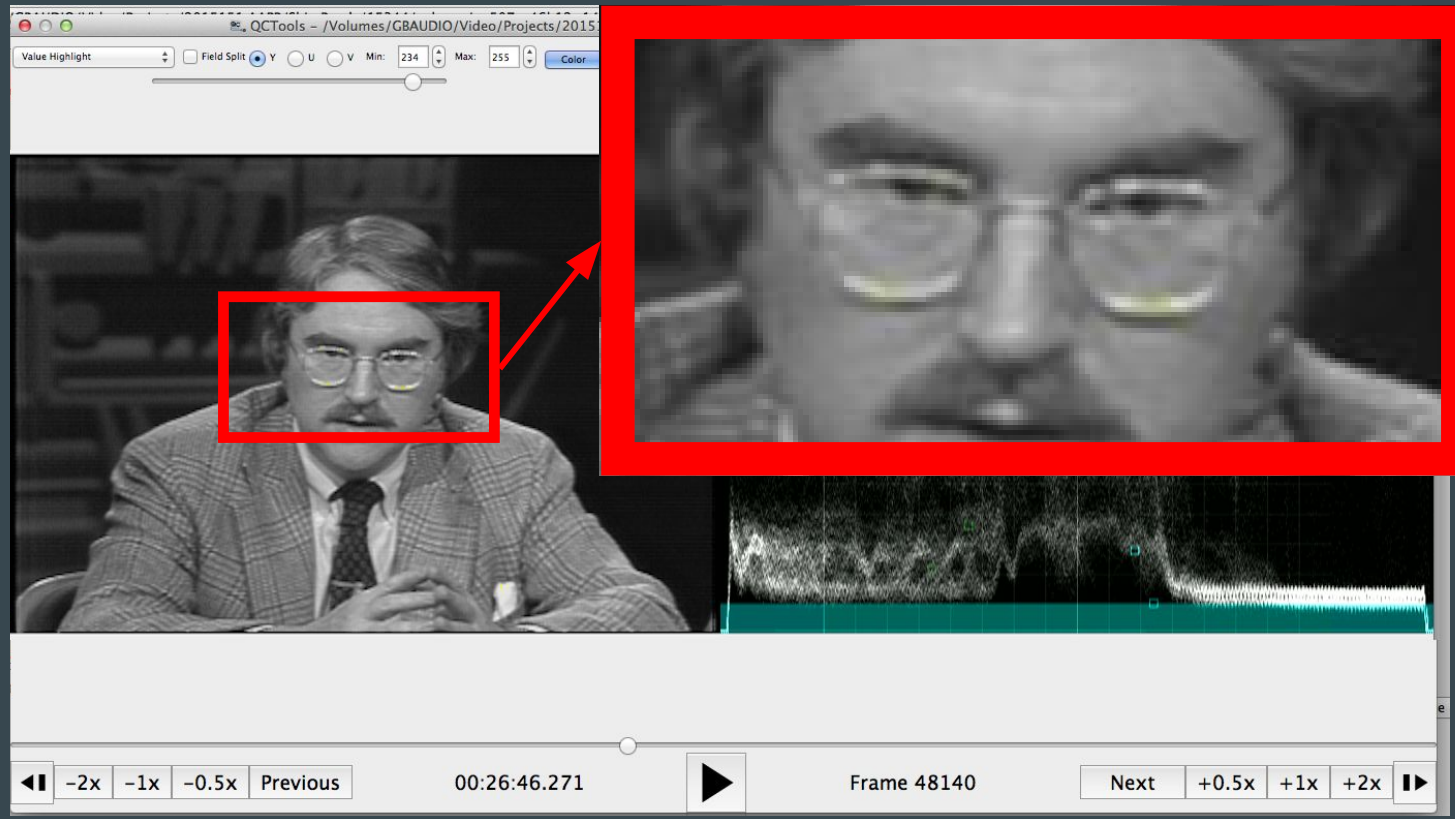
# Overs: To be or not to be (over)



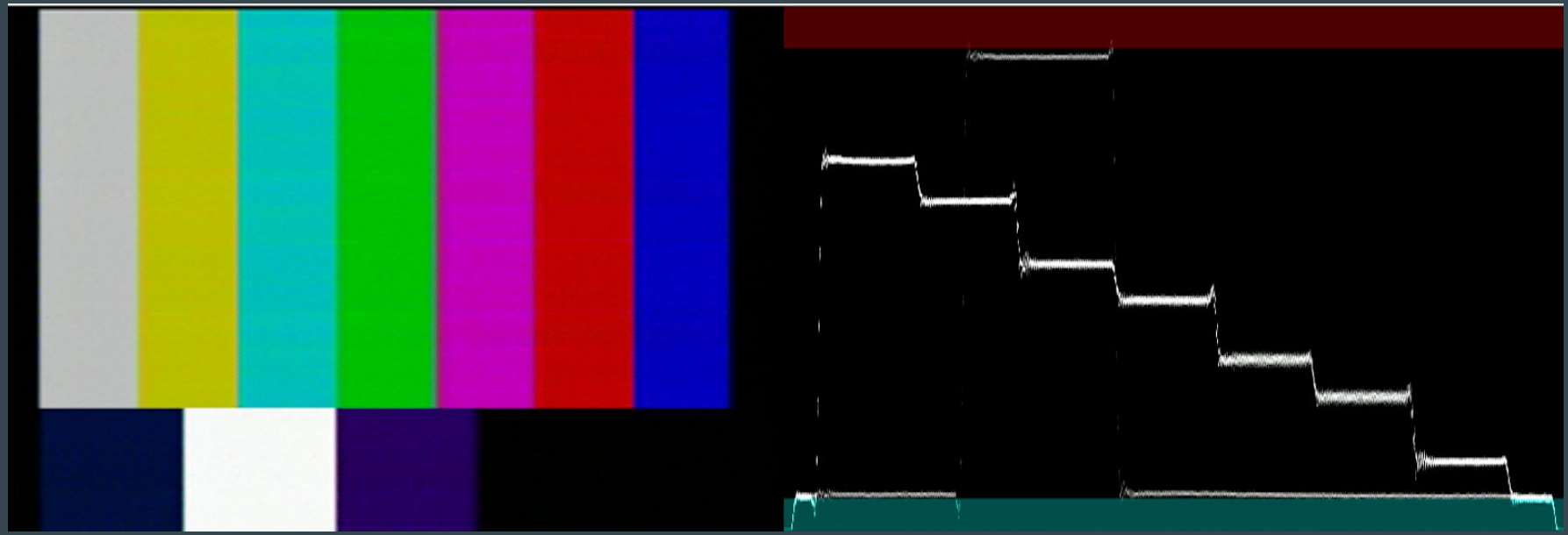
# Overs: To be or not to be (over)

# Overs: Metal Reflections

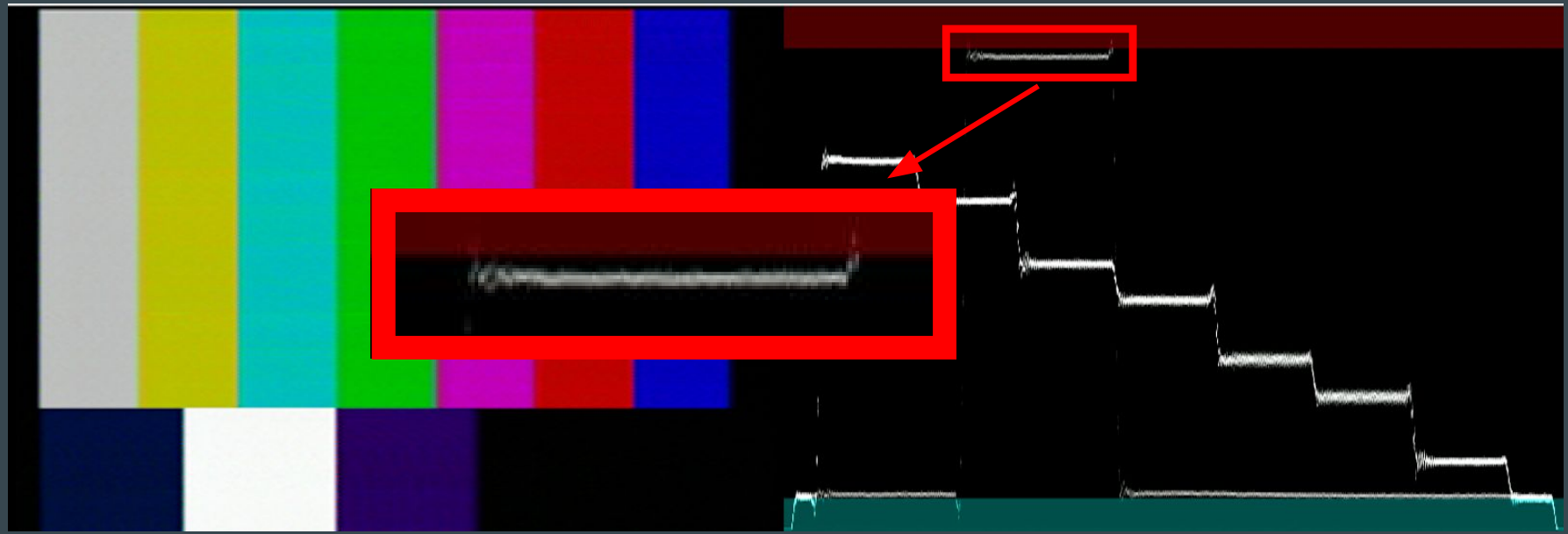
# Overs: Metal Reflections



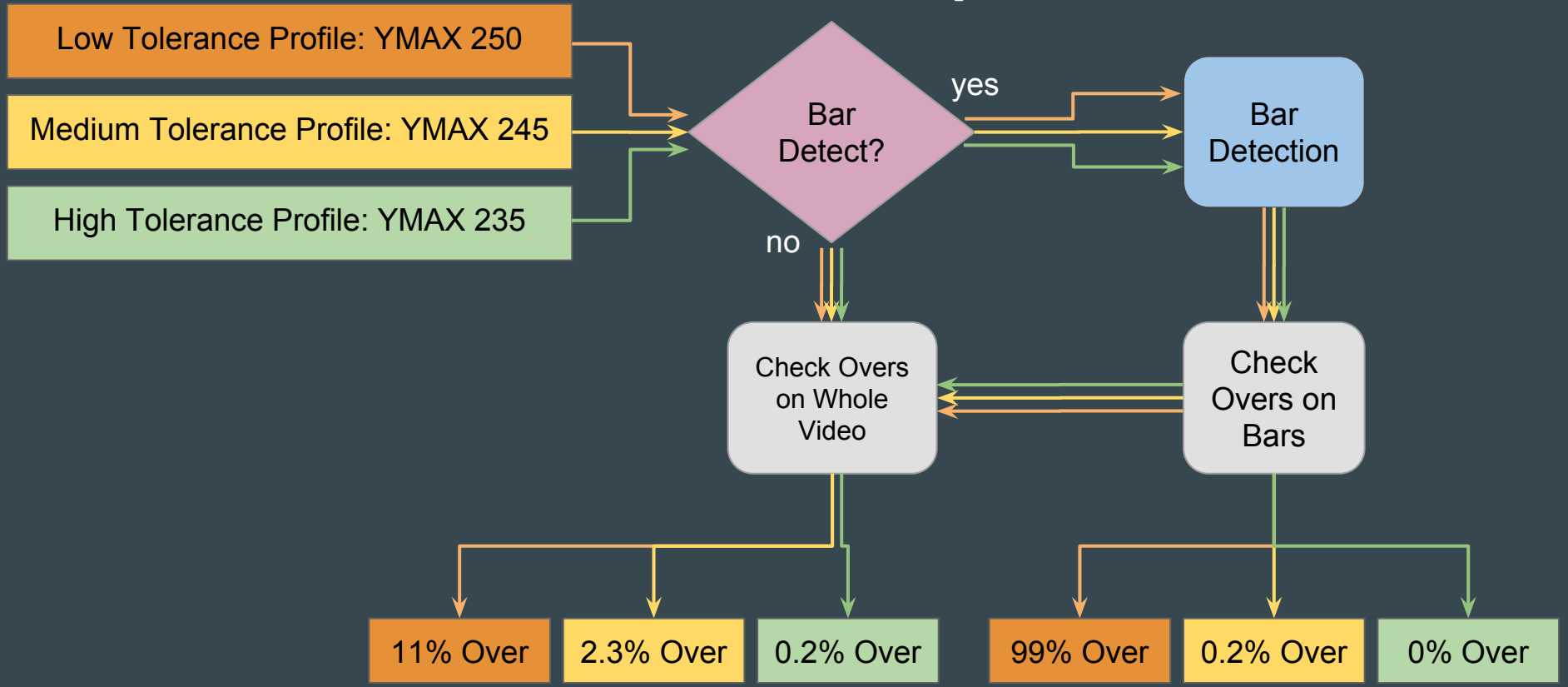
# Overs: Ringing



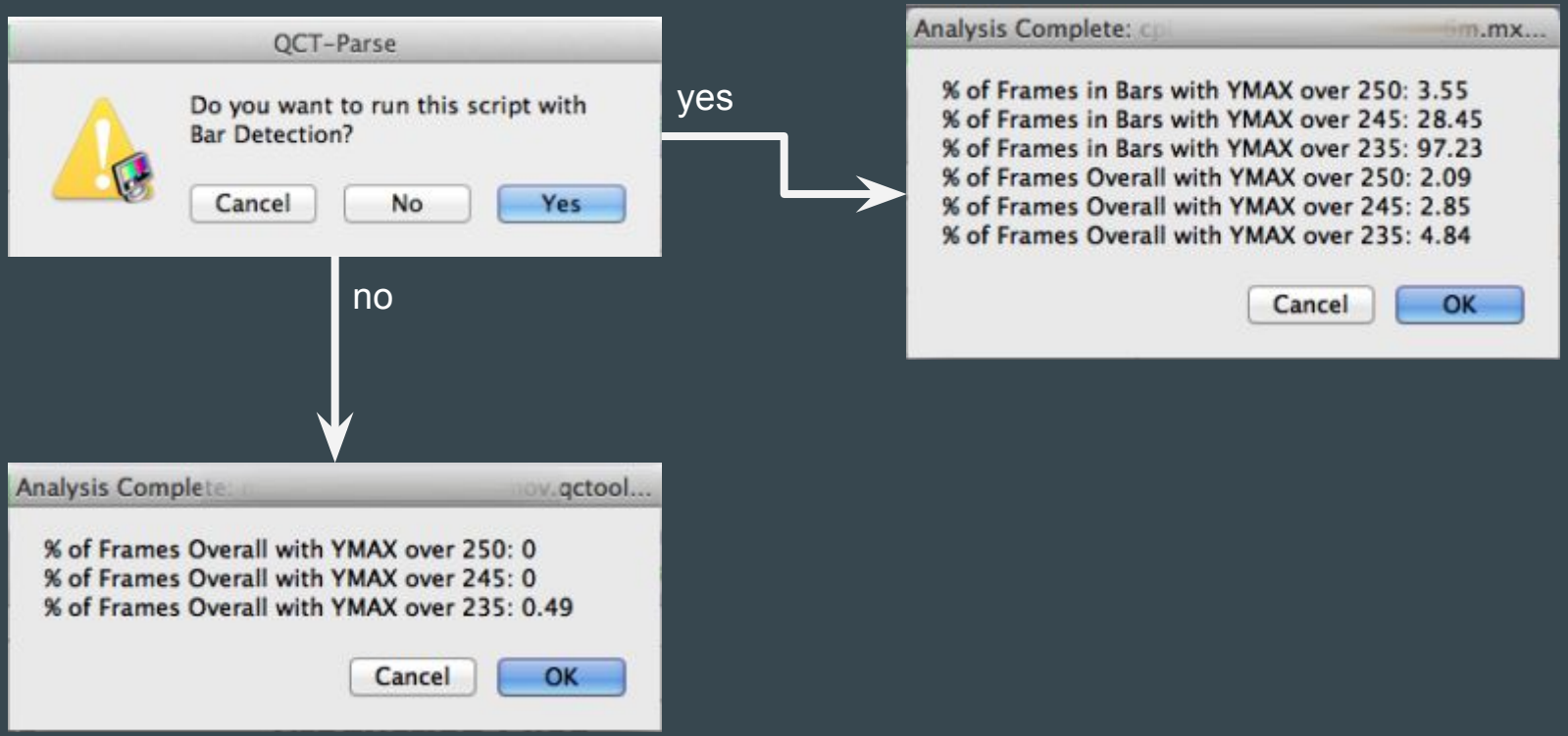
# Overs: Ringing



# OverCatch Script

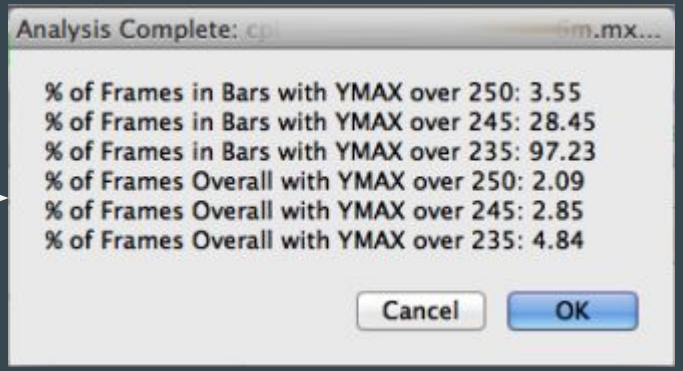


# OverCatch Script

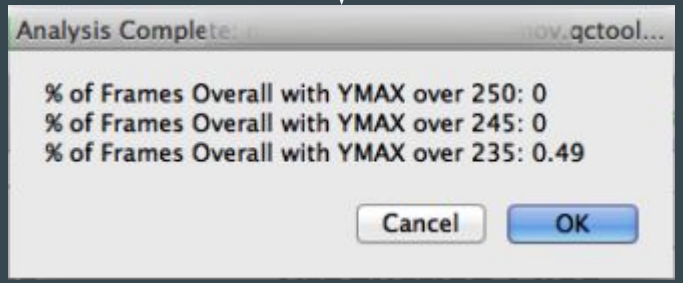


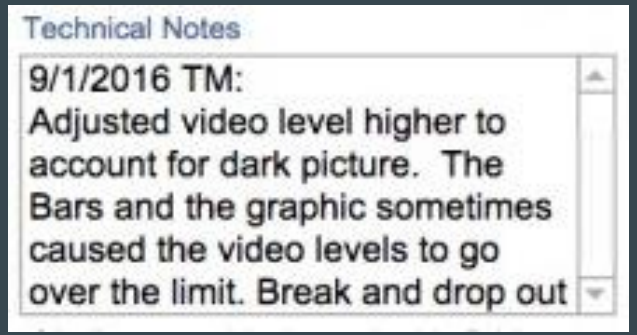
# OverCatch Script


yes




no



# Into The Future

## Future Development:

Headclog Detection

Dropout Detection

Duplicate Detection

# Into The Future

## Future Development:

Headclog Detection

Dropout Detection

Duplicate Detection

Community Practice

Better Metrics

Improved Vendor Relations

# A note on Free and Open-Source Software

Please make your methods public:

1. Helps others learn
2. Occasionally, neat stuff like this develops (which you don't necessarily anticipate)

ALSO:

Please send us your QCTools Reports

# Thank You

WGBH

NewsHour

Dave Rice

BAVC

Stanford University Libraries

Google Image Search

AMIA Conference Committee

AV Staff

Omni Hotel Staff