Fulfilling a Childhood Dream

Video Streaming for Remotely Controlling a Train

Simon Himmelbauer and Rainhard Driessler



About us

- Software Engineers @ A&R TECH
- Developing video solutions for railway
  - HMIs
  - SIL2 Camera Systems

RTO - Remote Train Operation



[1]

[1] Lesemaus

RTO - Remote Train Operation



[2]

[2]Driver's desk in the Remote Operation Centre. Copyright: Deutsche Bahn AG / Thomas Kiewning

RTO - Remote Train Operation

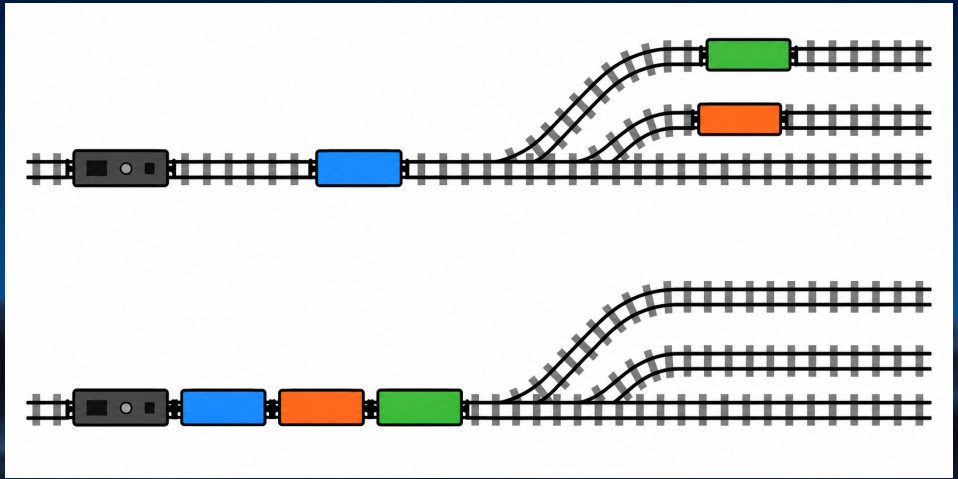
- Replica of train cabin in wayside office
- Control interfaces are relayed to actual train
- Communication over the air
- Perception via camera system

RTO - Remote Train Operation

- Replica of train cabin in wayside office
- Control interfaces are relayed to actual train
- Communication over the air
- Perception via **safe** camera systems

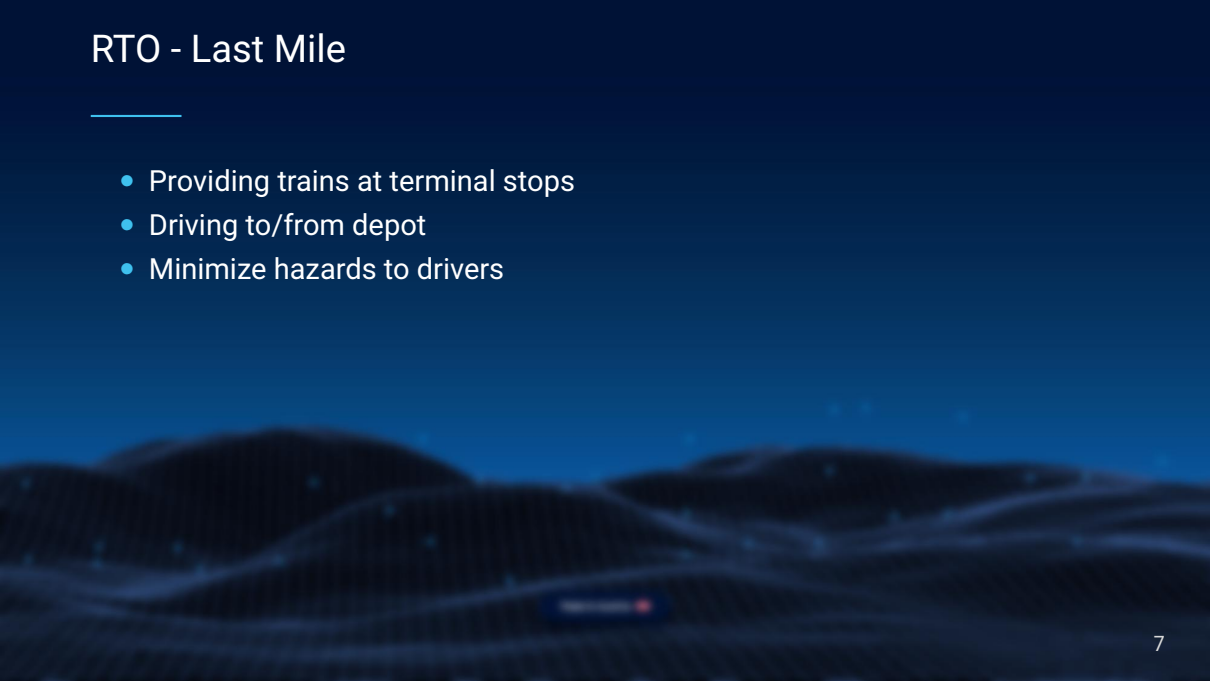


RTO - Shunting



RTO - Last Mile

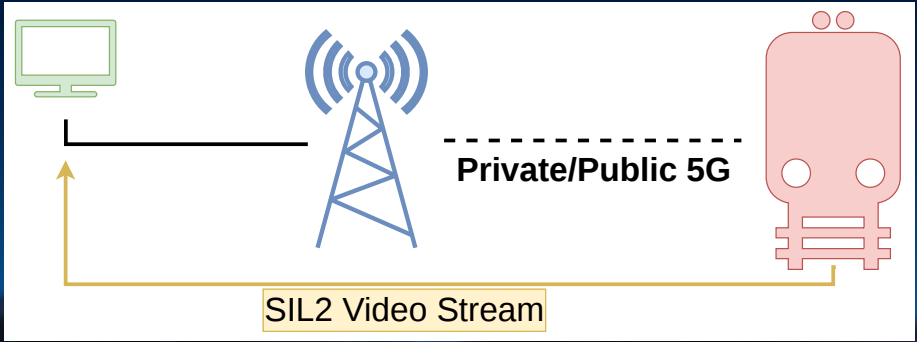
- Providing trains at terminal stops
- Driving to/from depot
- Minimize hazards to drivers



Setup



Setup



Setup





Challenges

- Lossy network (5G)

Challenges

- Lossy network (5G)
- Low Latency

Challenges

- Lossy network (5G)
- Low Latency
- Ideally no packet loss

Challenges

- Lossy network (5G)
- Low Latency
- Ideally no packet loss
- Unpredictable bandwidth
  - Conditions depend on weather and scene

Challenges

- Lossy network (5G)
- Low Latency
- Ideally no packet loss
- Unpredictable bandwidth
  - Conditions depend on weather and scene
- Latency spikes (antenna switch)

Challenges

- Just use RTP?

Challenges

- Just use RTP?
- Need to remain within functionally safe bounds

Challenges

- Just use RTP?
- Need to remain within functionally safe bounds
- Limits to buffering

Challenges

- Just use RTP?
- Need to remain within functionally safe bounds
- Limits to buffering
- Limits to rate control

Challenges

- Just use RTP?
- Need to remain within functionally safe bounds
- Limits to buffering
- Limits to rate control
- Optimization is key

RTP - Retransmissions

- Requires AVPF profile
  - Not supported by all RTSP clients (most notably FFMPEG)

[3] Worth upstreaming?

RTP - Retransmissions

- Requires AVPF profile
  - Not supported by all RTSP clients (most notably FFMPEG)
- Configure retransmission time on `GstRTSPMediaFactory`

[3] Worth upstreaming?

RTP - Retransmissions

- Requires AVPF profile
  - Not supported by all RTSP clients (most notably FFMPEG)
- Configure retransmission time on `GstRTSPMediaFactory`
- **NOTE**: `rtprtxsend` by default stores only past 100 packets!
  - Property not exposed via `GstRTSPMediaFactory`[3]

[3] Worth upstreaming?

RTP - MTU

- `rtp*pay` have property to configure `MTU`
  - Does not include UDP/TCP+IP or any other header

[4] https://gitlab.freedesktop.org/gstreamer/gstreamer/-/work_items/3310
[5] https://gitlab.freedesktop.org/gstreamer/gstreamer/-/merge_requests/10364

RTP - MTU

- `rtp*pay` have property to configure `MTU`
  - Does not include UDP/TCP+IP or any other header
- Default value: 1400 (bytes)
  - Can be too big when using IPv6 and/or VPNs

[4] https://gitlab.freedesktop.org/gstreamer/gstreamer/-/work_items/3310
[5] https://gitlab.freedesktop.org/gstreamer/gstreamer/-/merge_requests/10364

RTP - MTU

- `rtp*pay` have property to configure `MTU`
  - Does not include UDP/TCP+IP or any other header
- Default value: 1400 (bytes)
  - Can be too big when using IPv6 and/or VPNs
- **Possible solution:** Path MTU Discovery
  - Retrievable via TCP (Max Segment Size (MSS))
  - Apparently not always reliable[4]

[4] https://gitlab.freedesktop.org/gstreamer/gstreamer/-/work_items/3310
[5] https://gitlab.freedesktop.org/gstreamer/gstreamer/-/merge_requests/10364

RTP - MTU

- `rtp*pay` have property to configure `MTU`
  - Does not include UDP/TCP+IP or any other header
- Default value: 1400 (bytes)
  - Can be too big when using IPv6 and/or VPNs
- **Possible solution:** Path MTU Discovery
  - Retrievable via TCP (Max Segment Size (MSS))
  - Apparently not always reliable[4]
- Added `mtu-probing` property to `rtspsrc`[5]
  - Retrieves MSS from RTSP connection
  - Requests 'optimal' MTU from server via 'Blocksize' header
  - `gstreamer-rtsp-server` respects header as long as pipelines are NOT shared

[4] `https://gitlab.freedesktop.org/gstreamer/gstreamer/-/work_items/3310`
[5] `https://gitlab.freedesktop.org/gstreamer/gstreamer/-/merge_requests/10364`

Optimising H265 encoding

- Using ranged QP and target bitrate
  - Encoder adjusts QP based on current bandwidth budget

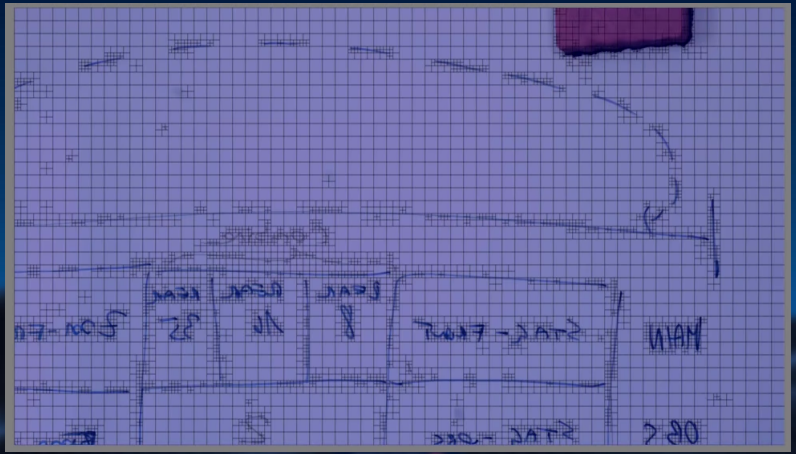
Optimising H265 encoding

- Using ranged QP and target bitrate
  - Encoder adjusts QP based on current bandwidth budget
- I-frames in H265 bitstream produce latency spikes
  - Even worse with multiple parallel streams

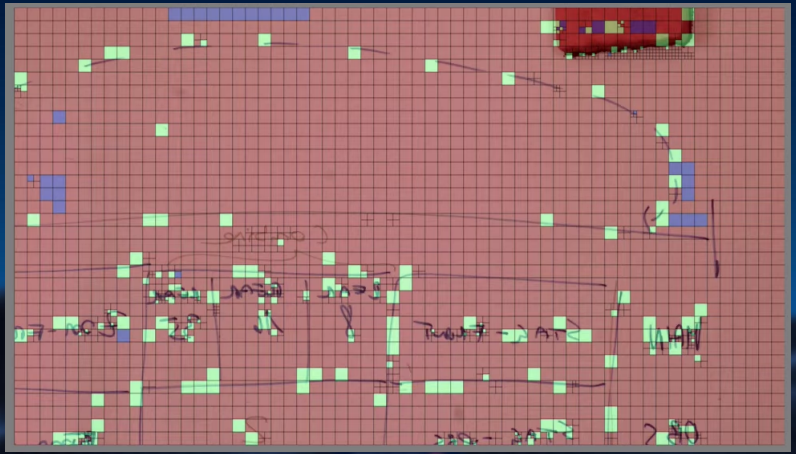
Optimising H265 encoding

- Using ranged QP and target bitrate
  - Encoder adjusts QP based on current bandwidth budget
- I-frames in H265 bitstream produce latency spikes
  - Even worse with multiple parallel streams
- **Solution:** Gradual Decoder Refresh (GDR)
  - Refresh stream over series of frames

Regular I-Frames

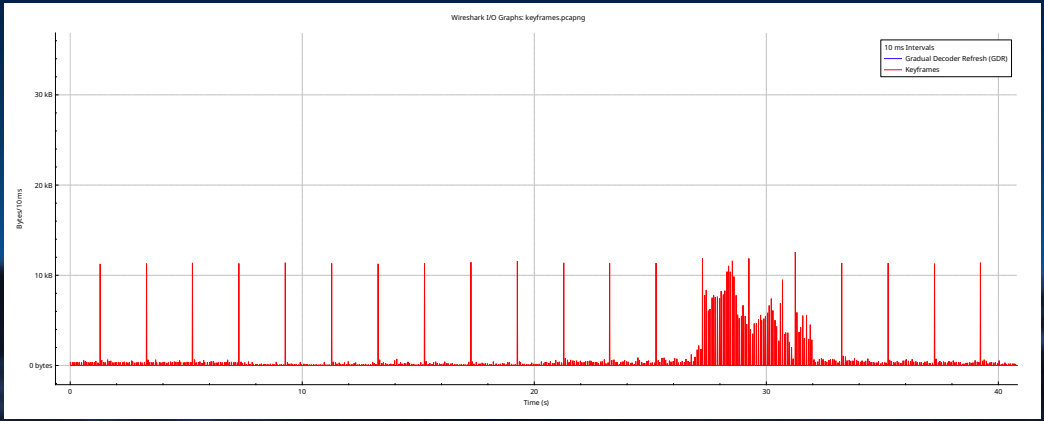


Regular I-Frames



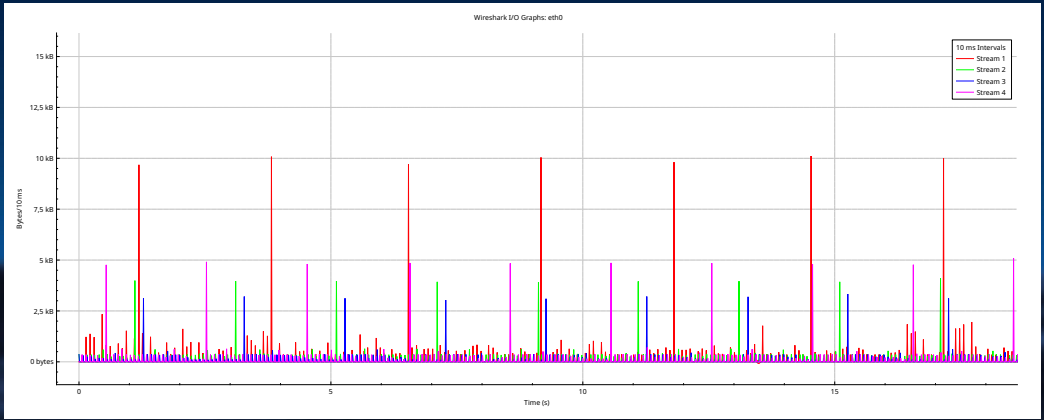
Regular I-Frames

Single Stream

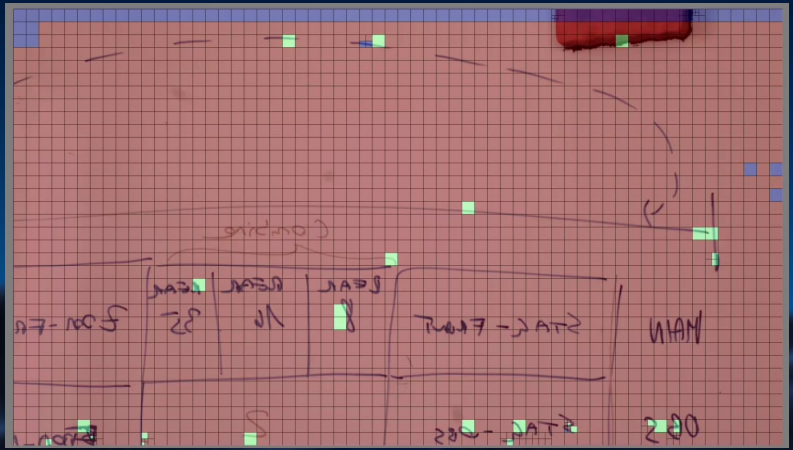


Regular I-Frames

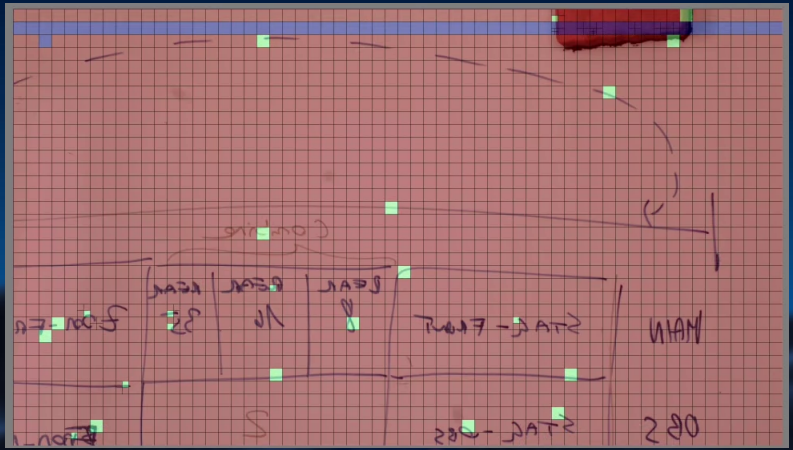
Four Streams



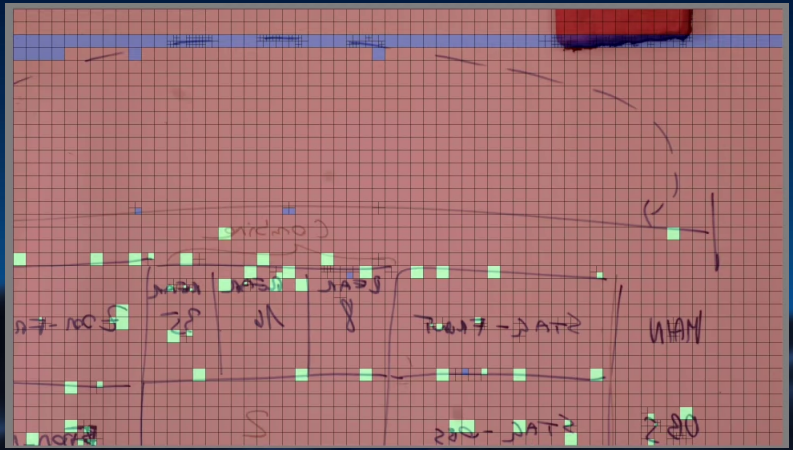
Gradual Decoder Refresh (GDR)



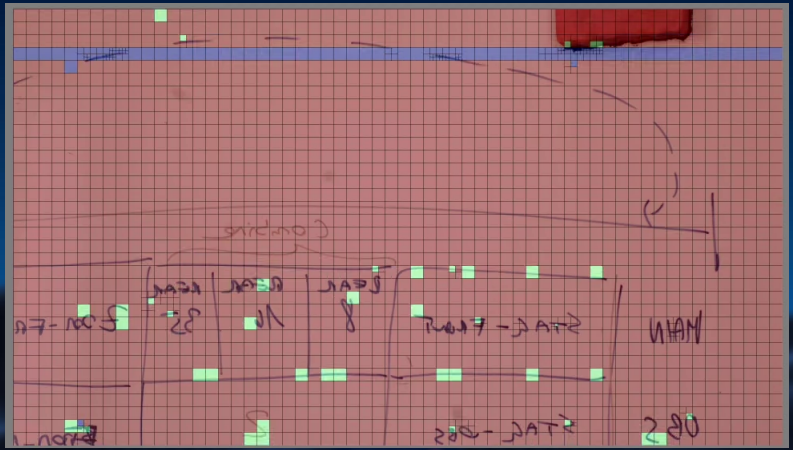
Gradual Decoder Refresh (GDR)



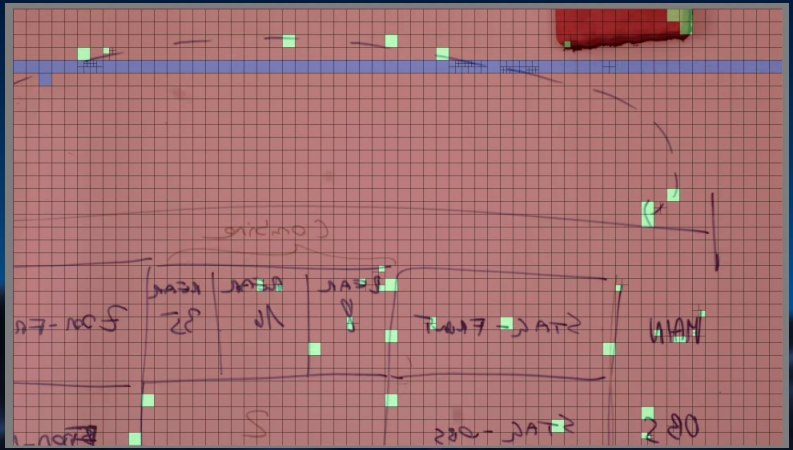
Gradual Decoder Refresh (GDR)



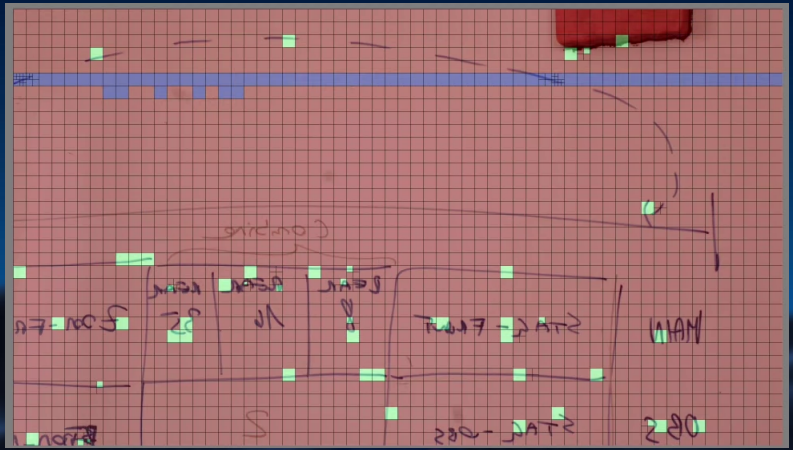
Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



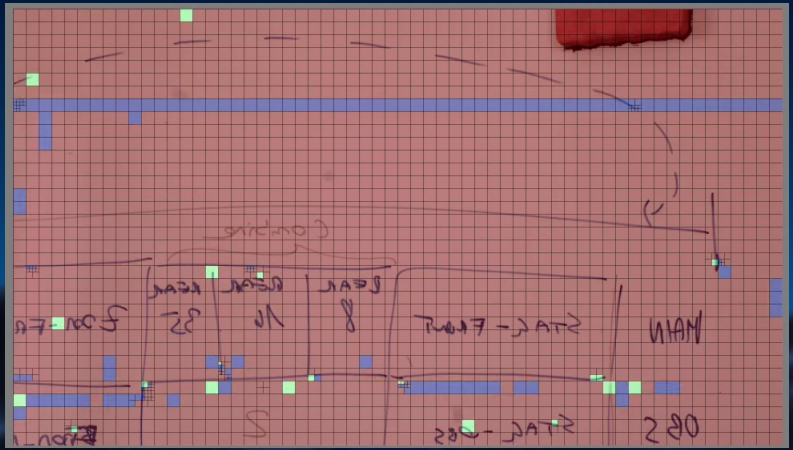
Gradual Decoder Refresh (GDR)



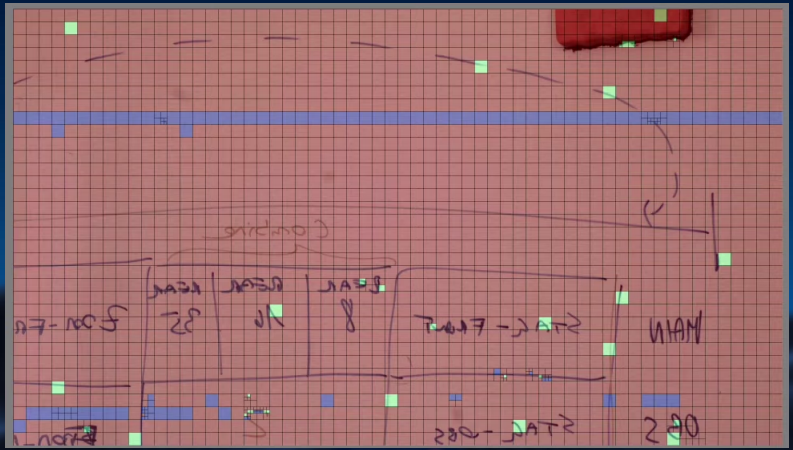
Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



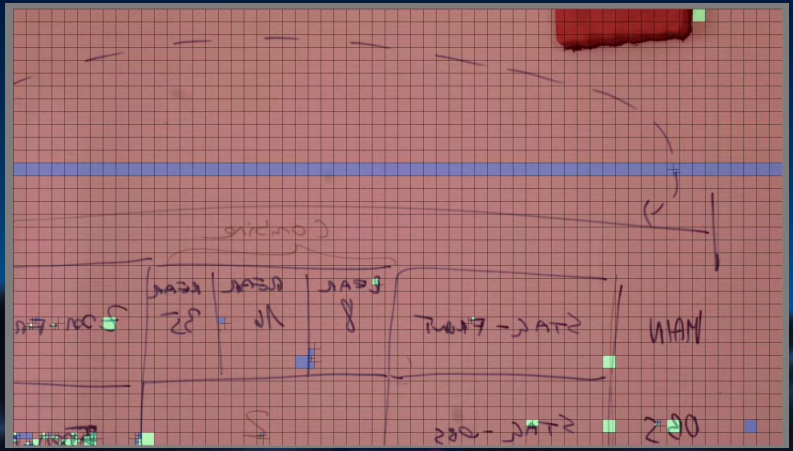
Gradual Decoder Refresh (GDR)



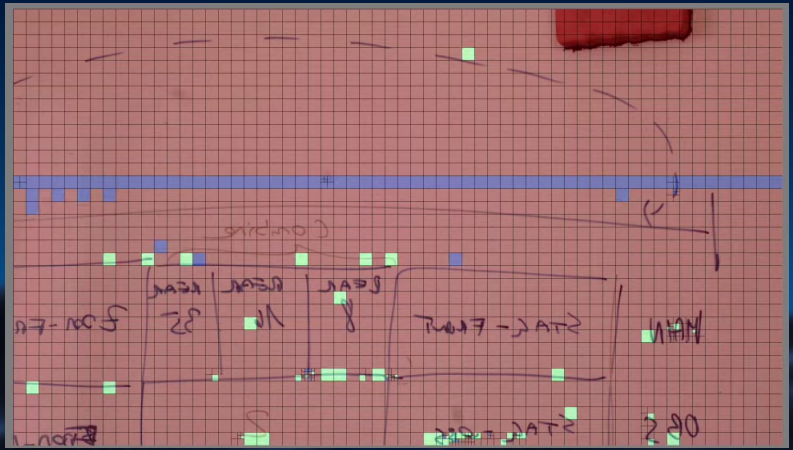
Gradual Decoder Refresh (GDR)



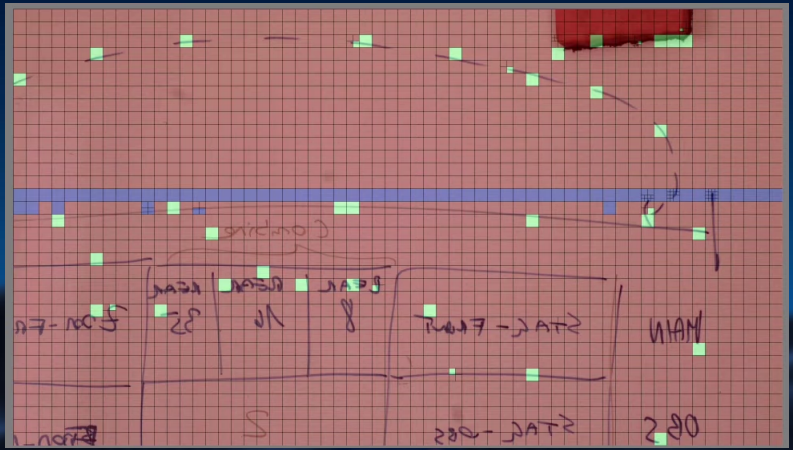
Gradual Decoder Refresh (GDR)



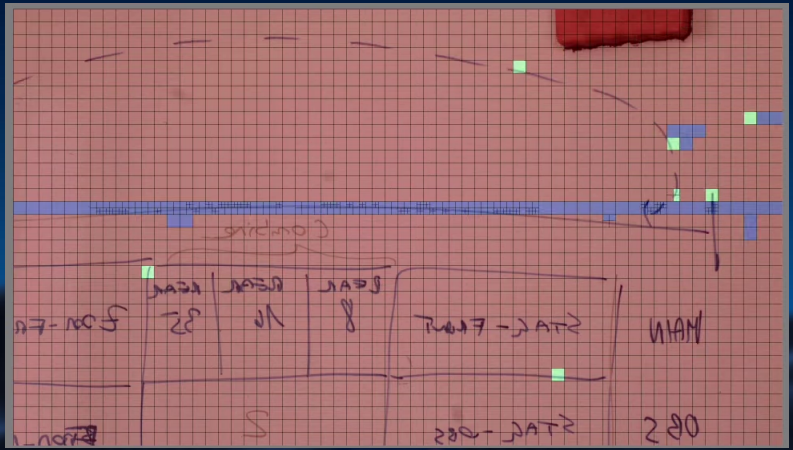
Gradual Decoder Refresh (GDR)



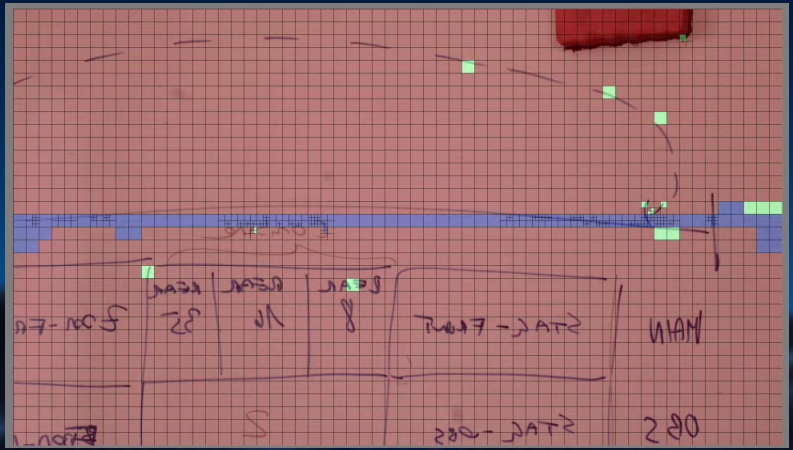
Gradual Decoder Refresh (GDR)



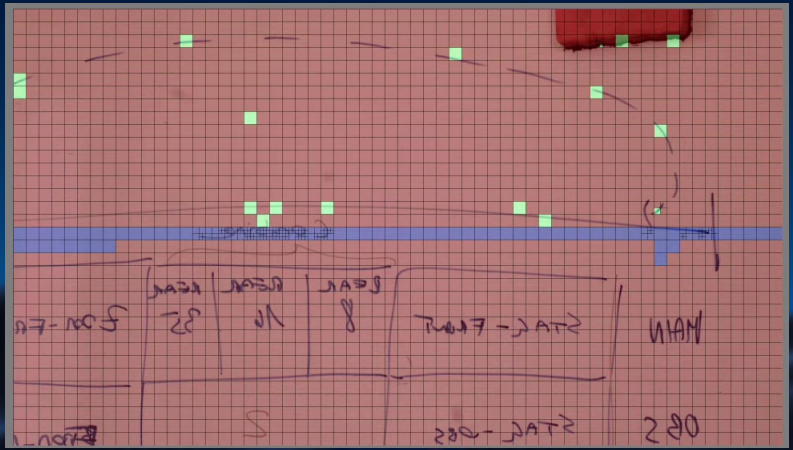
Gradual Decoder Refresh (GDR)



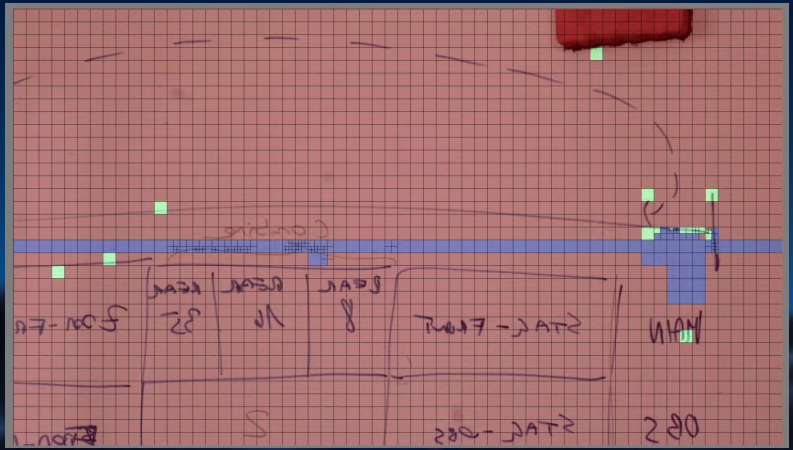
Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



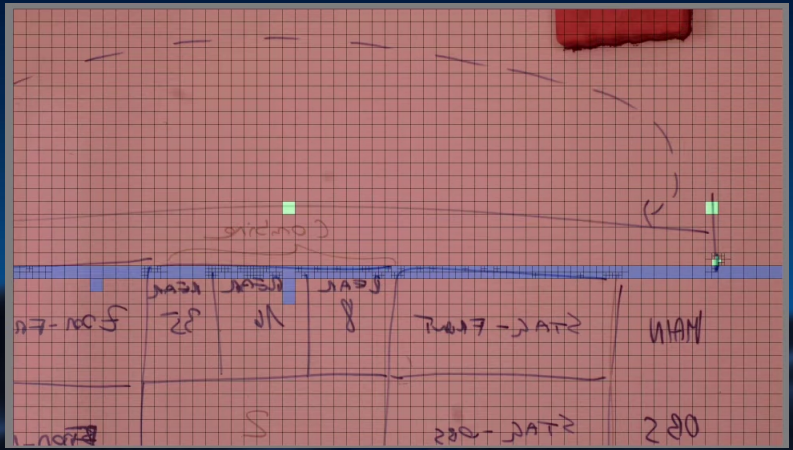
Gradual Decoder Refresh (GDR)



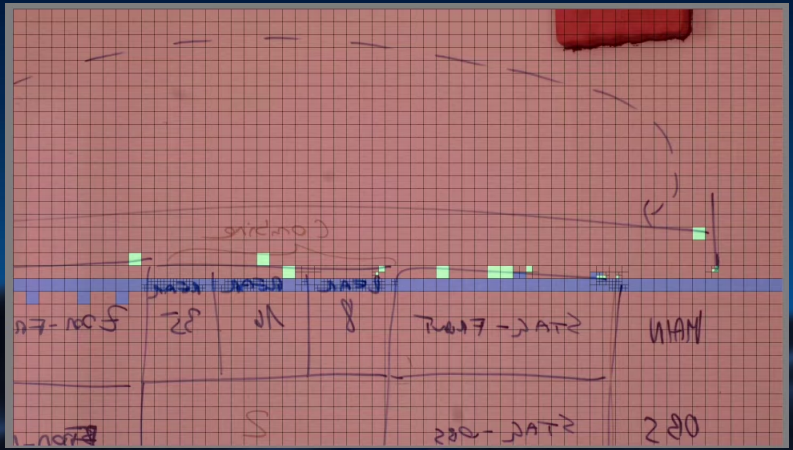
Gradual Decoder Refresh (GDR)



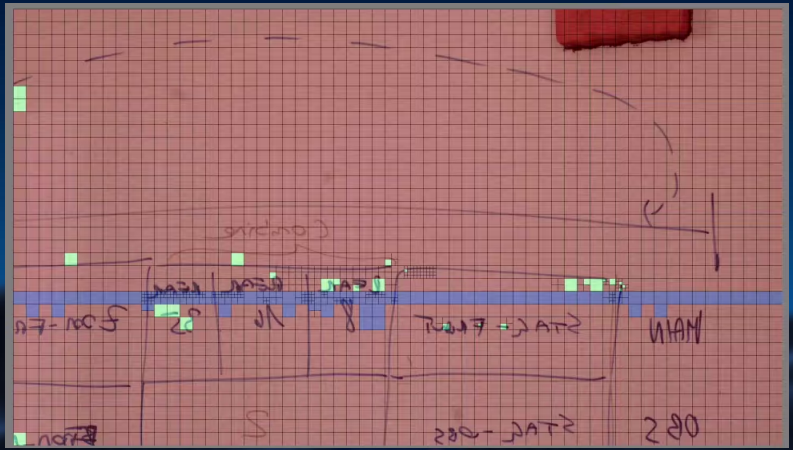
Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



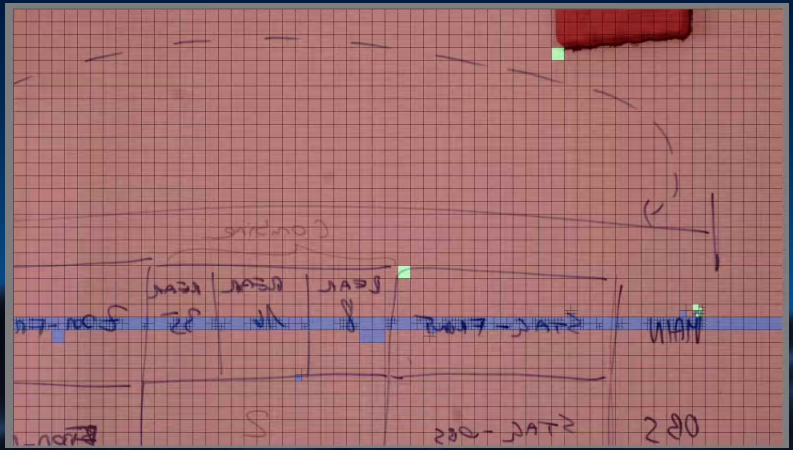
Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



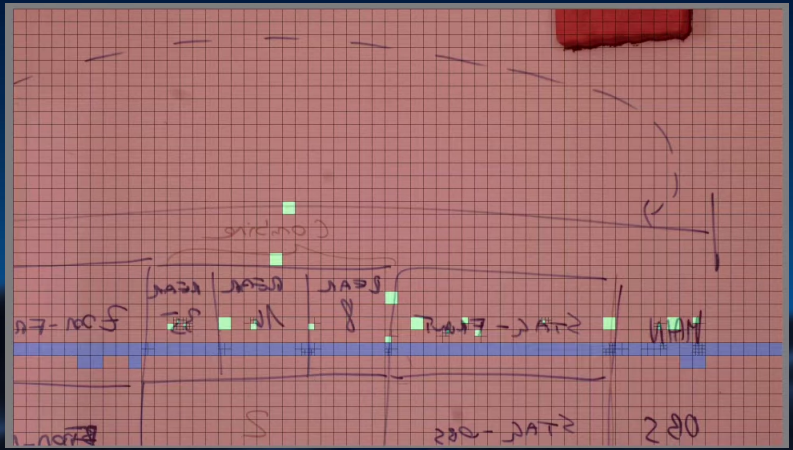
Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



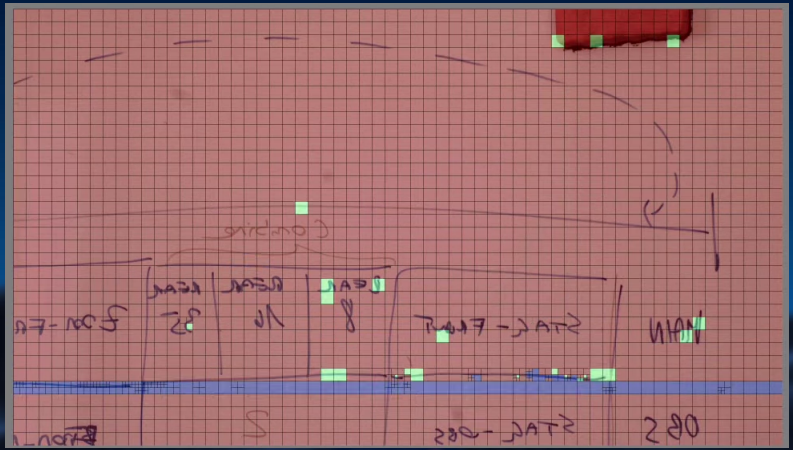
Gradual Decoder Refresh (GDR)



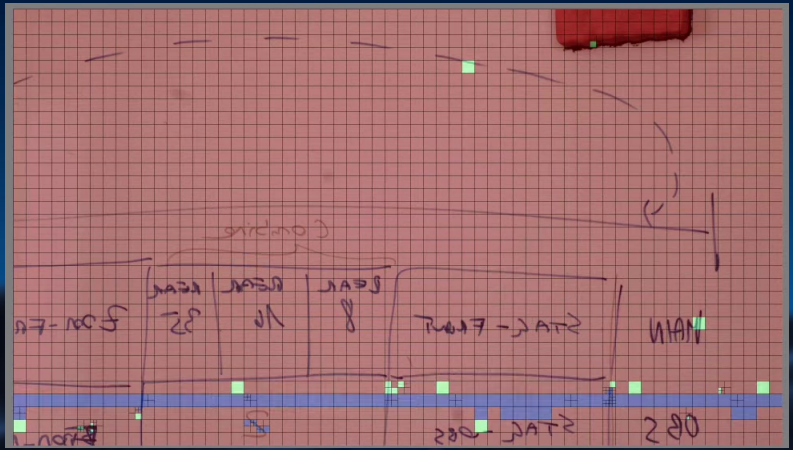
Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



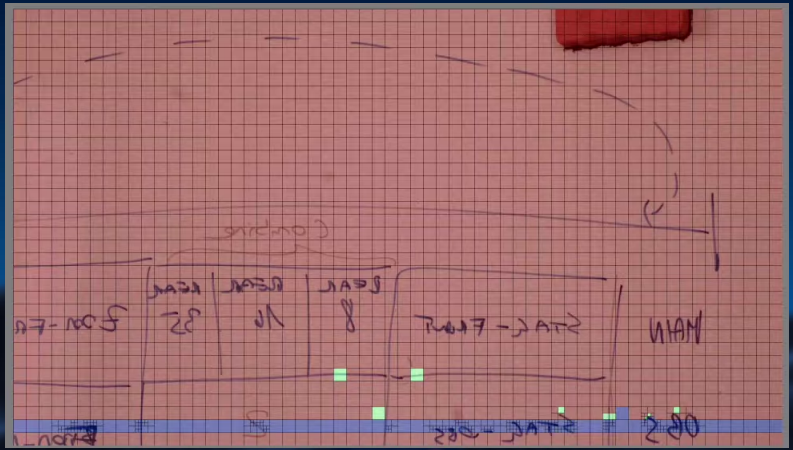
Gradual Decoder Refresh (GDR)



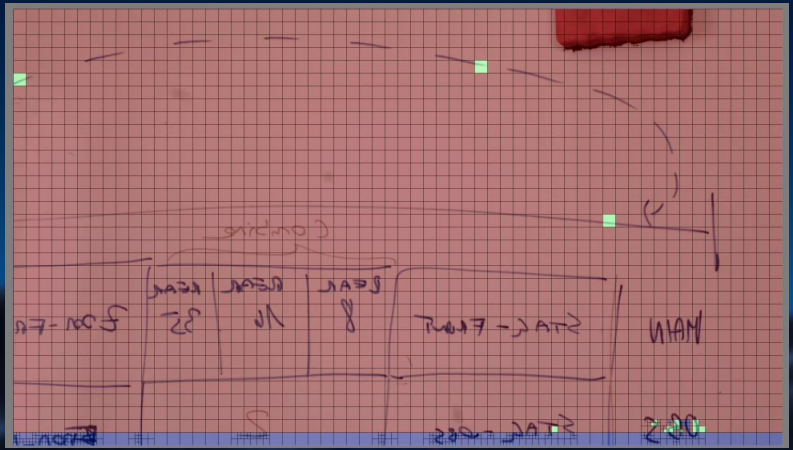
Gradual Decoder Refresh (GDR)



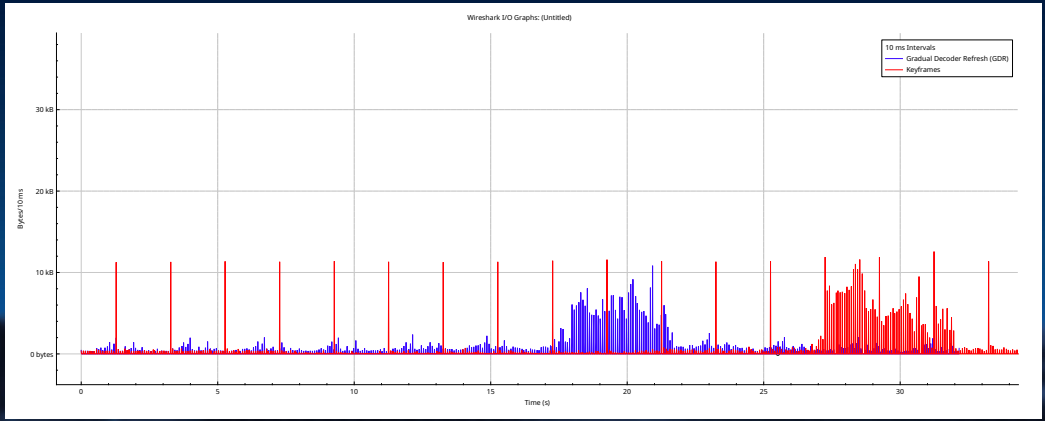
Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



Gradual Decoder Refresh (GDR)



Network Optimisations

- Packet loss seen even on `local` network (queues)

Network Optimisations

- Packet loss seen even on `local` network (queues)
- Traffic control (queueing disciplines)
- Hardware traffic shaping to smoothen network traffic

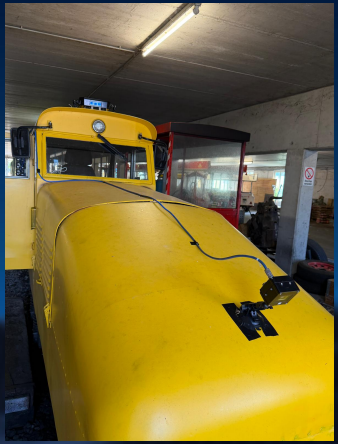
Network Optimisations

- Packet loss seen even on `local` network (queues)
- Traffic control (queueing disciplines)
- Hardware traffic shaping to smoothen network traffic
- Tuning retransmission requests

Network Optimisations

- Packet loss seen even on `local` network (queues)
- Traffic control (queueing disciplines)
- Hardware traffic shaping to smoothen network traffic
- Tuning retransmission requests
- Prioritise retransmission packets over regular stream

Liliputbahn





RTO

Currently, the RTO system has been developed for 30 km/h and for depots. Alstom expects around 25 percent time savings, as the train is changed at the push of a button and one no longer has to move between tracks. [6]

[6] https://www.heise.de/en/news/Remote-Train-Operation-The-hurdles-for-remote-control-of-large-trains-11468360.html

Lookout

- Dynamic resolution changes
- Variable framerate
- ROI based encoding
- ... all while remaining within safety bounds!

Questions?