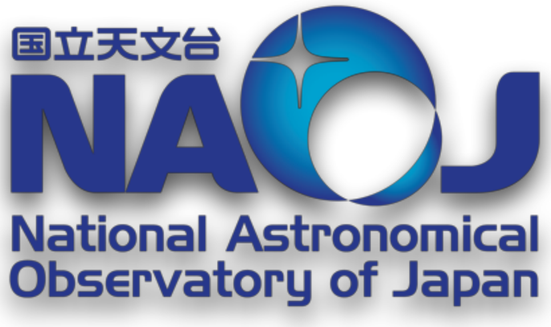
An introduction to FITSWebQL



C. Zapart, Y. Shirasaki, M. Ohishi, Y. Mizumoto, W. Kawasaki, T. Kobayashi,
G. Kosugi, E. Morita, A. Yoshino (NAOJ), S. Eguchi (Fukuoka Univ.)

FITS Web Quick Look

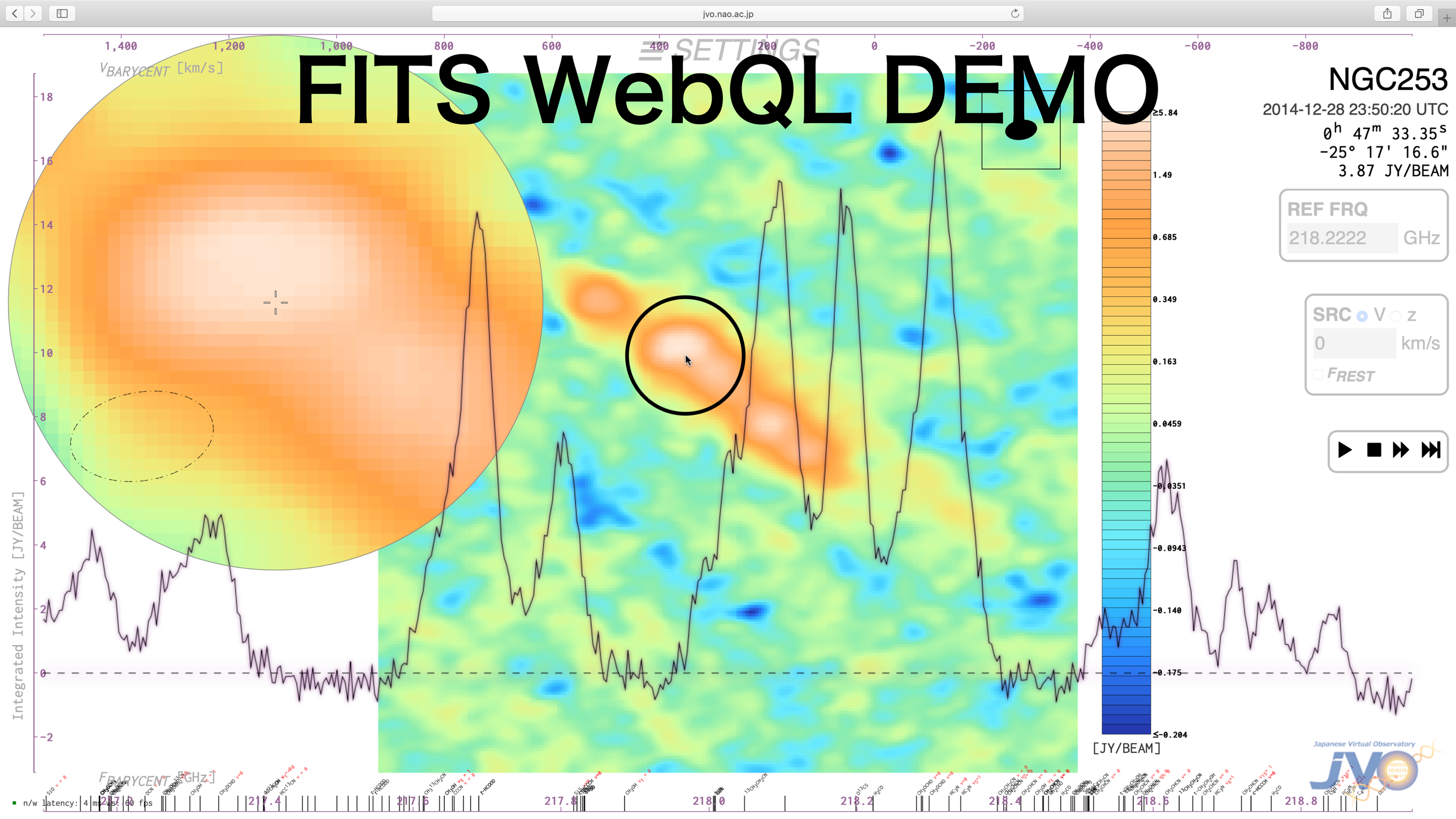
- preview over **100GB** large files in a web browser (no FITS file download)
- **exponential growth** in ALMA FITS file sizes
- **high-resolution** data cubes (10,000x10,000 pixels images, 4,000 frequency channels)
- FITS **cut-out**: download only a region of interest

FITS Web Quick Look

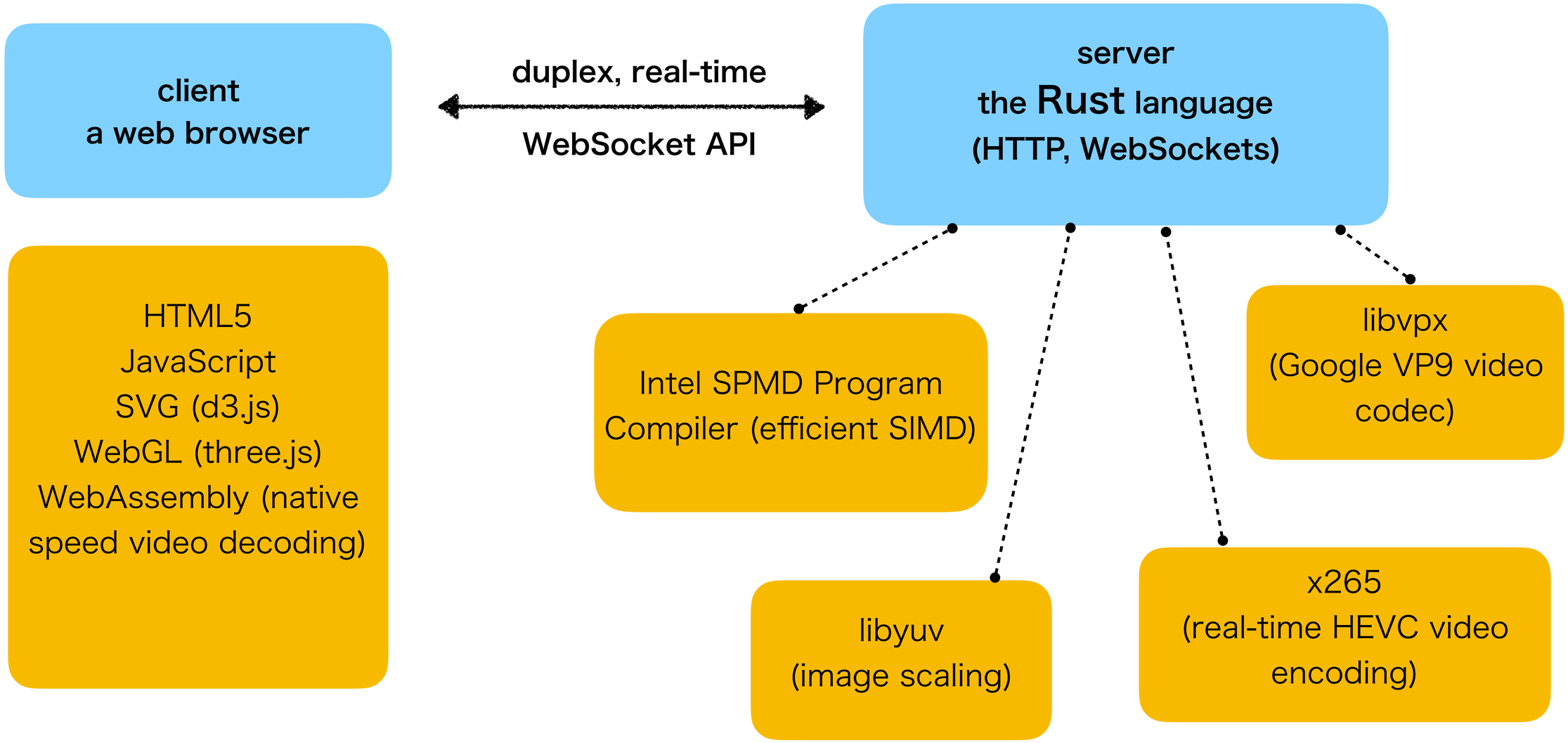
- the original JVO ALMA WebQL service demo at ADASS 2012
- ALMA WebQL v2 demo at ADASS 2016
- 2017: FITS WebQL v3 (3D view)
- 2018: FITS WebQL v4 (re-written in Rust, real-time streaming videos of FITS data cubes)

standalone desktop edition:

`https://github.com/jvo203/fits_web_ql`


jvo.nao.ac.jp
FITS WebQL DEMO
SETTINGS
NGC253
2014-12-28 23:50:20 UTC
0h 47m 33.35s
-25° 17' 16.6"
3.87 JY/BEAM
REF FRQ
218.2222 GHz
SRC V z
0 km/s
FREST
VBARYCENT [km/s]
FBARYCENT [GHz]
Integrated Intensity [JY/BEAM]
[JY/BEAM]
Japanese Virtual Observatory
n/w latency: 4 ms; 7.60 fps

technical architecture



why Rust?

Rust is a systems programming language that runs blazingly fast, prevents segmentation faults, and guarantees thread safety.

in a 24-hour continuous operation:

- no crashes 
- no memory leaks 
- fearless concurrency 

beware of a steep learning curve

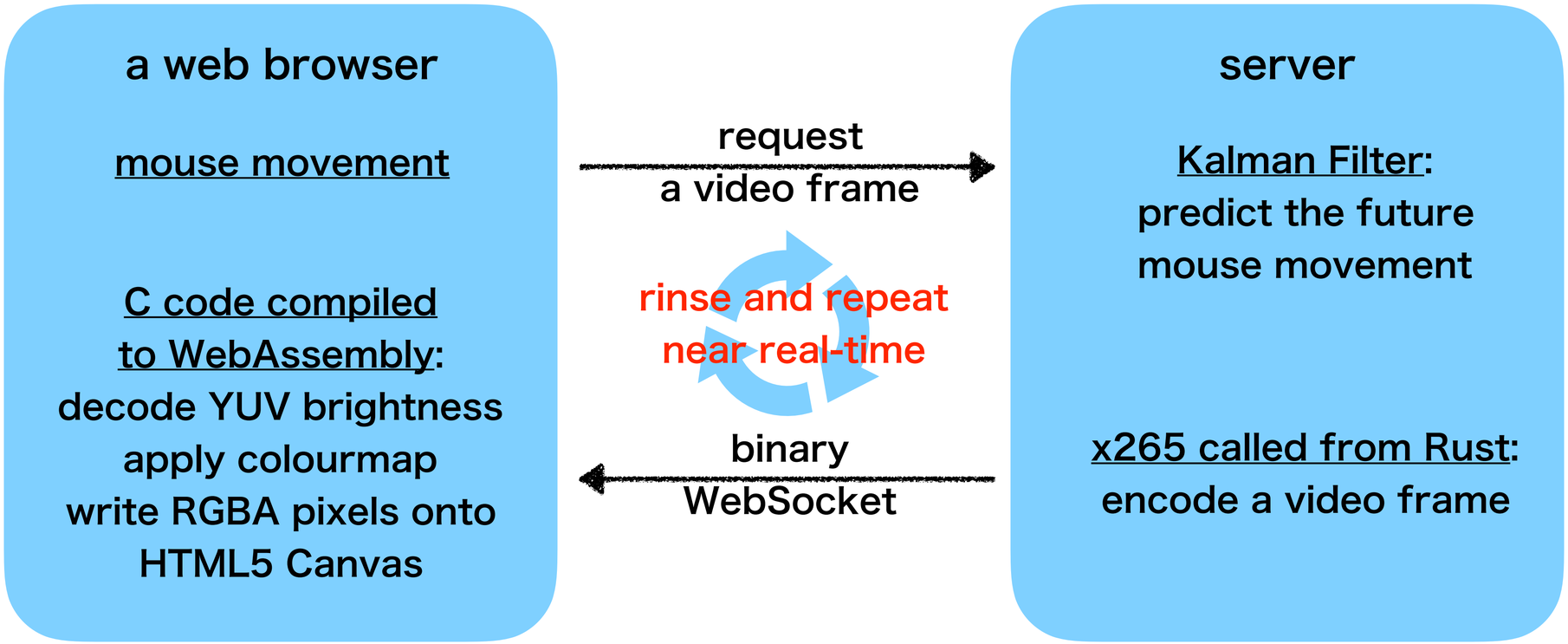
Rust: pros and cons

Rust is a systems programming language that runs blazingly fast, prevents segmentation faults, and guarantees thread safety.

- speed on par with **C/C++**, faster than **Java**, no garbage collection freezes
- compiler detects thread data races, a small runtime keeps an eye on array bounds
- **C/C++**: smooth compilation, headaches during execution
- **Rust**: frustration/headaches at compilation, plain sailing at runtime

WebAssembly (Wasm)

Compile and run high-level languages like C/C++/Rust in a web browser at native speed

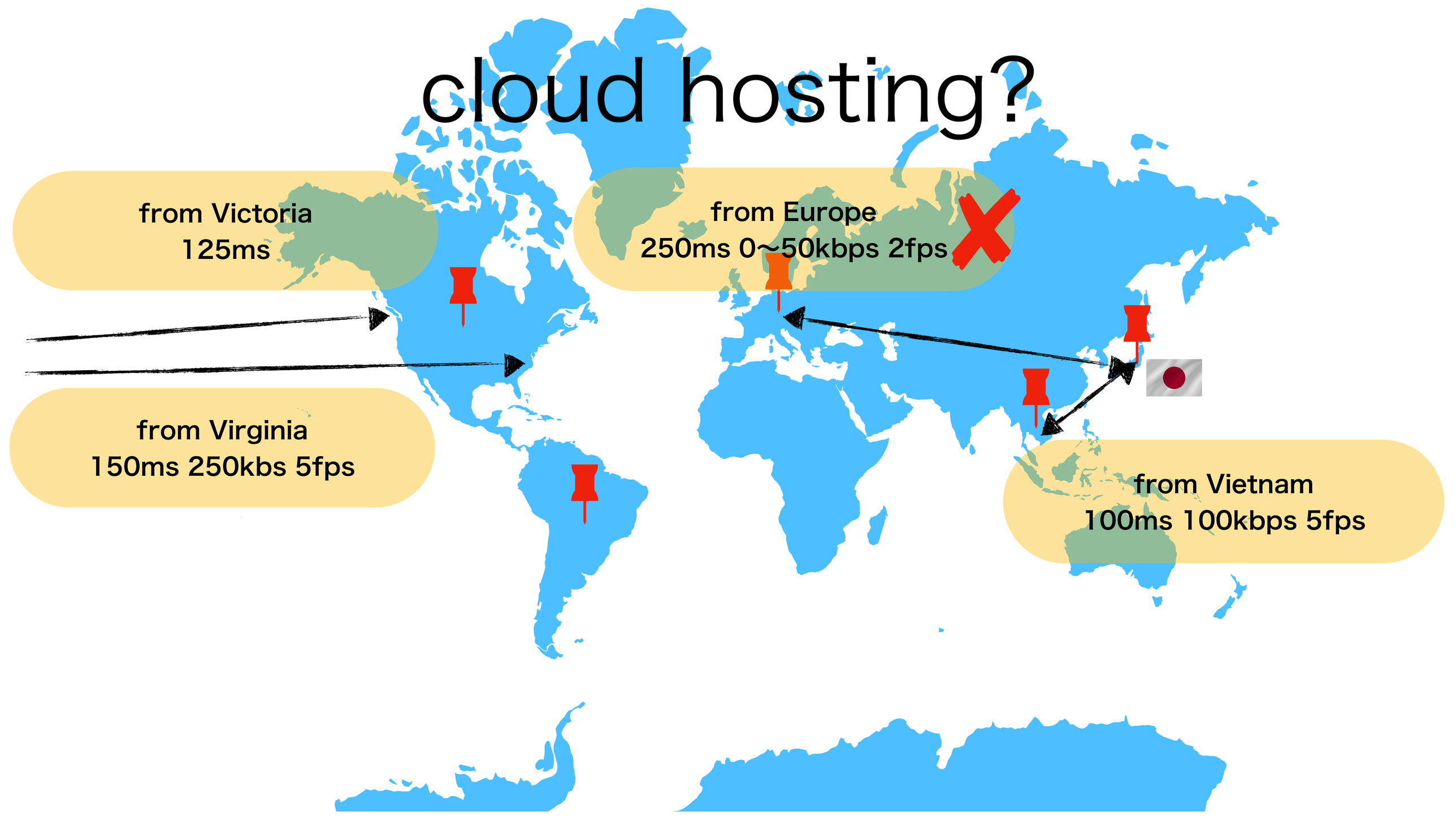


supported by all major browsers

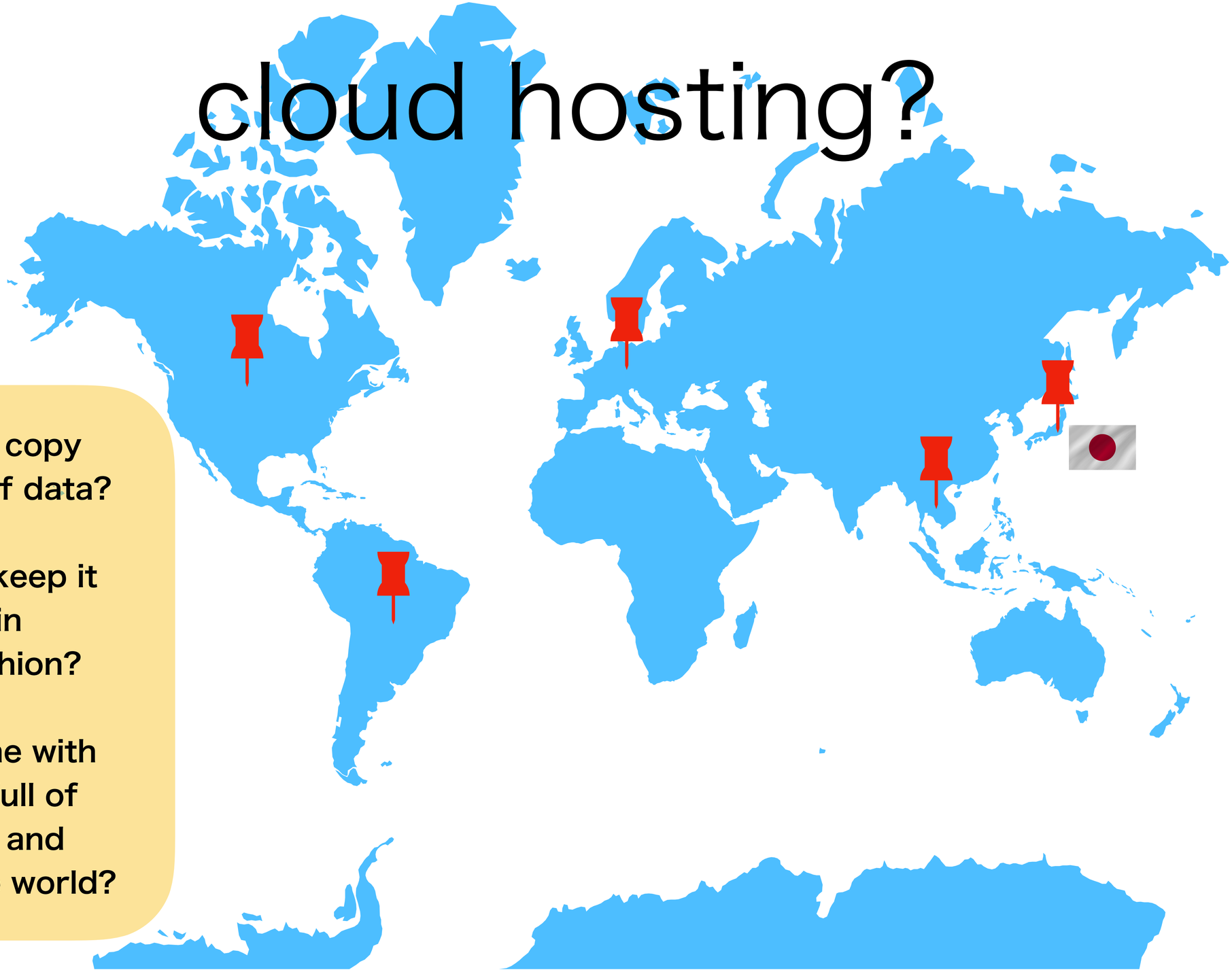
VP9 vs. HEVC

Google's VP9 (libvpx) **FITS cube images (a still keyframe)**	**HEVC (x265)** **real-time video encoding**
libvpx library: both an encoder and decoder	x265 library: only an encoder (search the Internet for a decoder to suit your task)
slower, less efficient encoding, inferior multithreading	faster than libvpx, more efficient (bandwidth-friendly), scales across all CPU cores
no greyscale (an overhead of handling redundant RGB/YUV channels)	YUV 4:0:0 support (server-encode as greyscale, add colour in the client)
an easy API, trivial to compile the decoder into WebAssembly	extreme difficulty finding a suitable JavaScript decoder (DIY: FFmpeg C API compiled to WebAssembly)

cloud hosting?



cloud hosting?

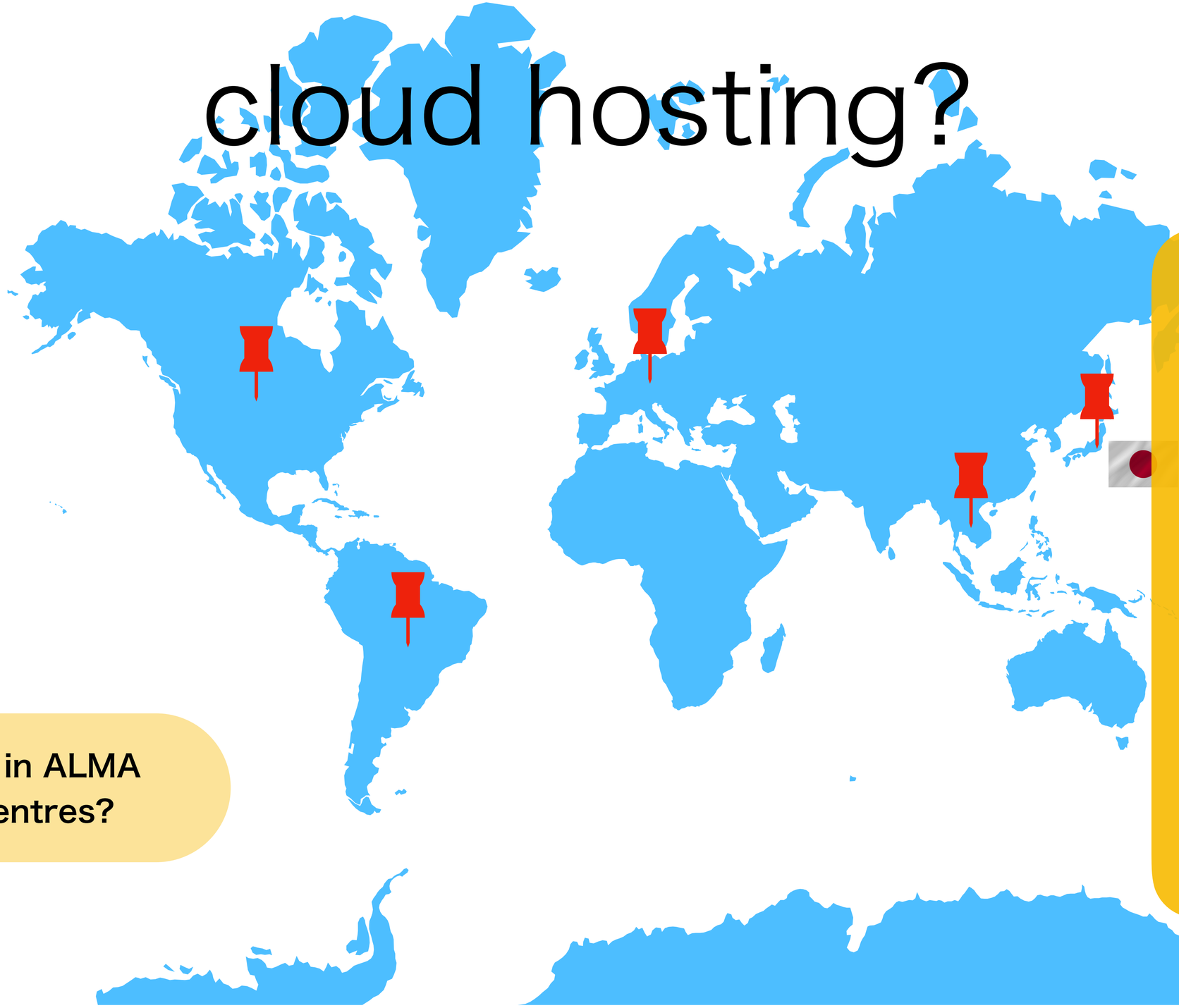


how do you copy
over 100TB of data?

how do you keep it
in sync in
a timely fashion?

get on a plane with
a suitcase full of
hard disks and
fly around the world?

cloud hosting?



host servers in ALMA Regional Centres?

256GB RAM
dual CPU socket
(32 threads)

2xPCI Express
NVME SSDs in
RAID0

SATA III SSDs

NFS HDD RAID

thank you Rust

- superior **stability**, improved **performance**
- better **memory management**

Google “JVO Portal”:
https://jvo.nao.ac.jp/portal/top-page.do

Google “fitswebql”:
https://github.com/jvo203/fits_web_ql