实时多媒体处理与传输

——Web下一代RTC技术实践

高纯
声网
Background
WebRTC

Capabilities:
- Media capture
- Adaptive encoding (VP8, VP9, H264)
- Media Processing (AEC AGC ANC)
- Capability Negotiation (SDP)
- NAT traverse (TURN/STUN)
- Network QoS (NACK, FEC, TCC, GCC)
- Encryption (DTLS)

Specification:
- WebRTC 1.0: Real-Time Communication Between Browsers
- Media Capture and Streams
WebRTC based System

Capabilities Required:

- Customized transporting protocols
- Customized codecs
- Data channel which is exactly synchronized with media data to support online gaming and metaverse
- Enhanced media processing algorithms (AI based 3A, image enhancement, etc.)
- Alpha channel transporting
- ...

Cloud

Transport Network

Edge Server

Edge Server

Edge Server

WebRTC Gateway

Native Client

Native Client

Native Client

Web Client

Web Client

Web Client
WebRTC NV based Solutions
Raw Data to Web Worker

- CPU usage of JS main thread

Insertable Media Processing + Web Worker

Agora Virtual Background online Demo

Processing in main thread
In general, media data should be processed in Web Worker to enhance performance of JS main thread.

- `postMessage()` with `ArrayBuffer` performance

**PostMessage performance data from Surma**

![Graph showing performance data for Chrome and Firefox](image-url)
Raw Data to Web Worker

Passing raw data to worker with transferable objects.

- MediaStreamTrack Insertable Media Processing using Streams

```javascript
const transformer = new TransformStream({
  async transform(videoFrame, controller) {
    let blurredFrame = blurBackground(videoFrame);
    videoFrame.close();
    controller.enqueue(blurredFrame);
  }
});
readableStream.pipeThrough(transformer)
  .pipeTo(writableStream);
```

- mediacapture-extensions (Mozilla Draft)

```javascript
[Exposed=(Window,Worker), Transferable]
partial interface MediaStreamTrack {
}
```
Known Issues

- MediaStreamTrackGenerator h264 HW encoding compatibility issue

A same h264 HW encoding issue as MediaStreamTrack from HTMLMediaElement.captureStream() or from remote. Related chromium issues:

- Issue 1156408: If the stream obtained via HTMLVideoElement.captureStream() is sent on peer connection, it does not send any actual data.

- Issue 1132965: MediaStream when forwarded through WebRTC stops sending frames, tainted?
WebCodecs Issues

- 48bit image is not supported by WebCodecs

IDL

```javascript
interface VideoFrame {
constructor(CanvasImageSource image, optional VideoFrameInit init = {});
...}
```

type (HTMLOrSVGImageElement or HTMLVideoElement or HTMLCanvasElement or ImageBitmap or OffscreenCanvas or VideoFrame) CanvasImageSource;

Fail Case

```javascript
const image = new Image();
image.src = './48bit.png';
image.onload = () => {
  // Uncaught (in promise) DOMException: Failed to construct
  // 'VideoFrame': Failed to create video frame
  const videoFrame = new VideoFrame(image, 0);
  videoFrame.copyTo(destination);
}
```
WebCodecs Issues

- Incorrect color space in VideoFrame local renderer
WebCodecs Issues

- Incorrect color space in VideoFrame local renderer

```javascript
let frameBuffer = new Uint8Array(width * height * 4);

let transformer = new TransformStream({
    async transform(videoFrame, controller) {
        gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, width, height,
        0, gl.RGBA, gl.UNSIGNED_BYTE, videoFrame);
        gl.bindFramebuffer(gl.FRAMEBUFFER, null);
        gl.drawArrays(gl.TRIANGLES, 0, 6);
        gl.readPixels(0, 0, width, height,
        gl.RGBA, gl.UNSIGNED_BYTE, frameBuffer);

        const newFrame = new VideoFrame(frameBuffer, {
            format: 'RGBA',
            codedWidth: width,
            codedHeight: height,
            timestamp: videoFrame.timestamp
        });
        videoFrame.close();
        controller.enqueue(newFrame);
    }
});

readableStream.pipeThrough(transformer).pipeTo(trackGenerator.writableStream);

playerElem.srcObject = trackGenerator;
playerElem.player();
```
WebTransport

WebTransport over HTTP/3 support for multiple streams, unidirectional streams, out-of-order delivery, and reliable as well as unreliable transport.

Session resumption and 0-RTT data are TLS features which can be used with both TCP and QUIC. To send an HTTP request during the handshake.
WebRTC-wasm

- Port WebRTC to WebAssembly
WebRTC-wasm

- VCM initialization
- Data receiving in network thread
Conclusion

- WebRTC is an excellent tool to build effective RTC apps on web.
- WebRTC can not meet all needs of build leading edge apps.
- WebRTC NV introduces more powerful standalone APIs and covers more RTC/media processing use cases.
- Agora has delivered many interesting features (AI noise cancelation, Virtual Background, Spatial Audio, Beauty effect, etc.) with WebRTC NV API, some of them has been integrated to “Agora Video Call”.
- WebRTC-wasm is being developed by Agora WebRTC team, it combines the WebRTC pipeline/algorithms and WebRTC NV APIs to provide better user experience.
THANKS