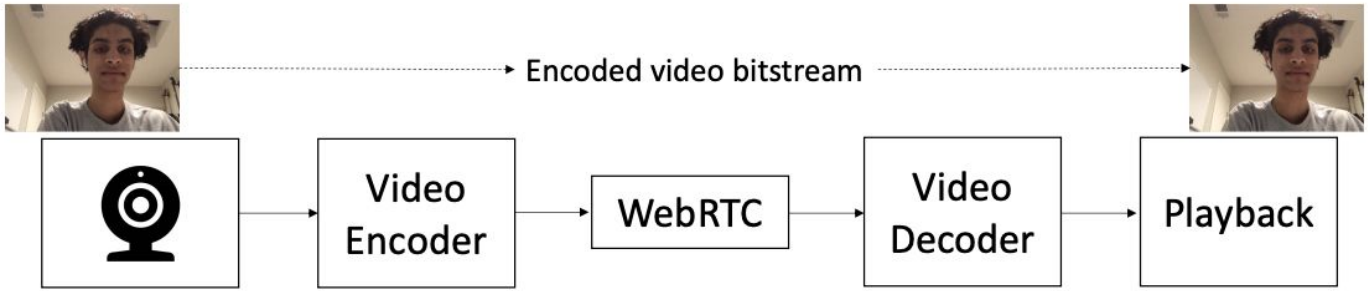
Reducing latency and bandwidth for video streaming using keypoint extraction and digital puppetry

Roshan Prabhakar
DCC 2021

Video Streaming



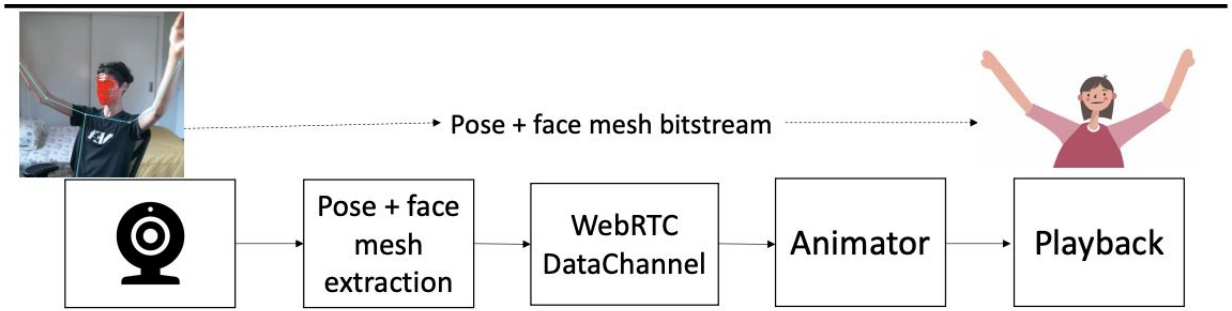
(a) Typical video streaming pipeline

Conventional vs. keypoint-centric codecs

- Conventional
  - Encode video as it is to enable visually similar reconstruction
  - No preservation bias to any one section of the video feed
  - Requires significant bandwidth to achieve high quality video
- Keypoint-centric
  - Focus on keypoints that carry most useful information
  - Encoding: extract keypoints
  - Transmission: transmit keypoints
  - Decoding: reconstruct video/animation based on keypoints
  - Much lower bandwidth consumption
  - Extraction and animation processes must be low latency

Project

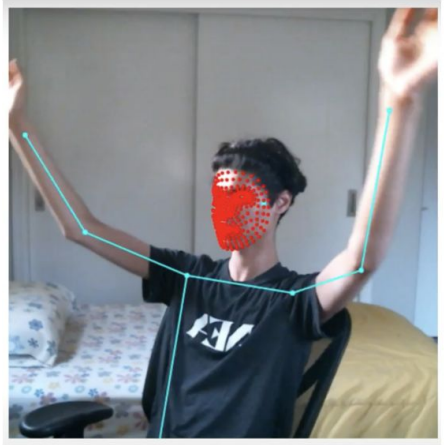
- Prototype
  - Implemented for video conference use case
  - Streaming of Key Points related to the human frame (a detailed face mesh + pose skeleton)
- Pose animator
  - Allows for the creation of a digital puppet based on human key points within a video feed
  - Extracts location keypoints, coder: extraction, decoder: animation reconstruction

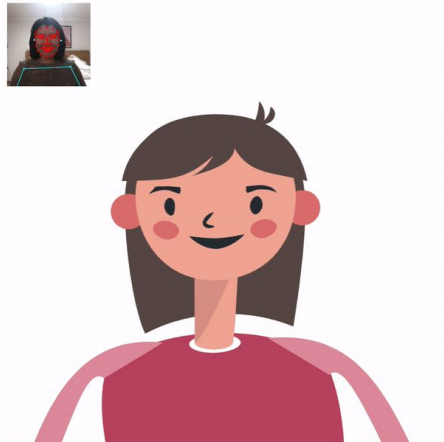


(b) Proposed streaming pipeline

Pose-animator

- https://github.com/yemount/pose-animator
- Efficient tensorflow neural networks for pose/face mesh extraction
- SVG skeleton-based animation
- Real-time
- No streaming aspect





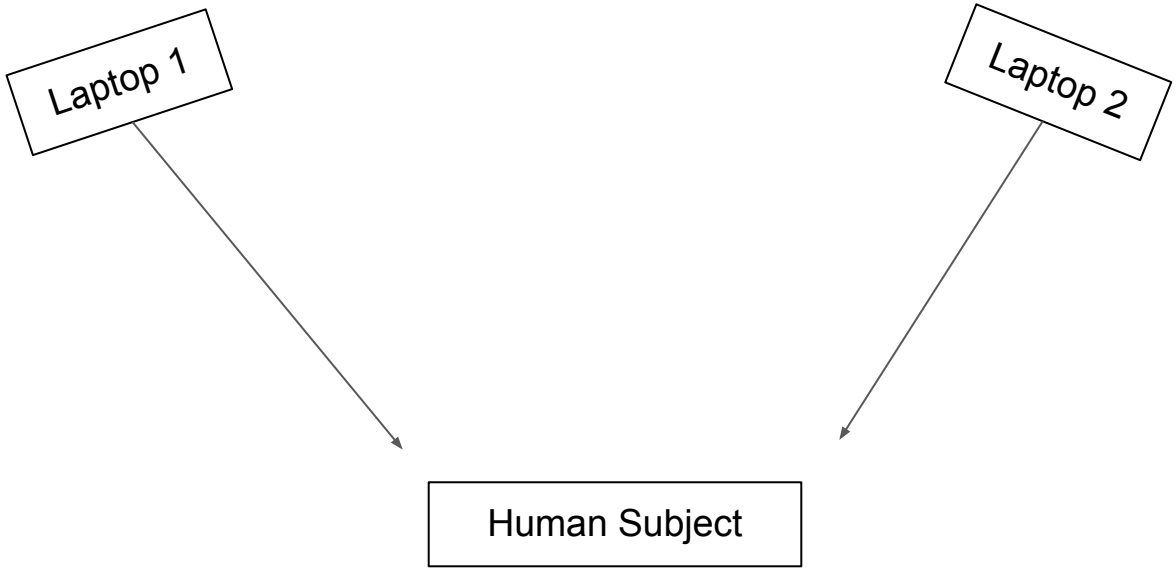
WebRTC

- Open web standard for real-time communication for the web
- Connection involves two steps:
  - Exchange of “peer addresses” through signaling server
  - Direct peer-to-peer streaming of information
- Specialized connection modes for video/audio transmission
- We use WebRTC data channel
  - Allows exchange of arbitrary messages
  - Transmit mesh points and their confidence values
  - Only transmit the mesh points that are used for animation

Experimental Setup

- Both conventional and proposed frameworks tested on computers on same LAN network
- Experiments run on typical laptop without specialized GPUs
- Measurement:
  - Bandwidth
  - Net latency
  - Latency due to keypoint extraction
  - Latency due to animation
- No audio
- https://github.com/shubhamchandak94/digital-puppetry

Experimental Setup



Keypoint-centric Demo

Measurement type	Typical range
Bitrate	25-35 kbps
Net latency	140-190 ms
Extraction latency	60-100 ms
Transmission latency	40-60 ms
Render latency	10-15 ms

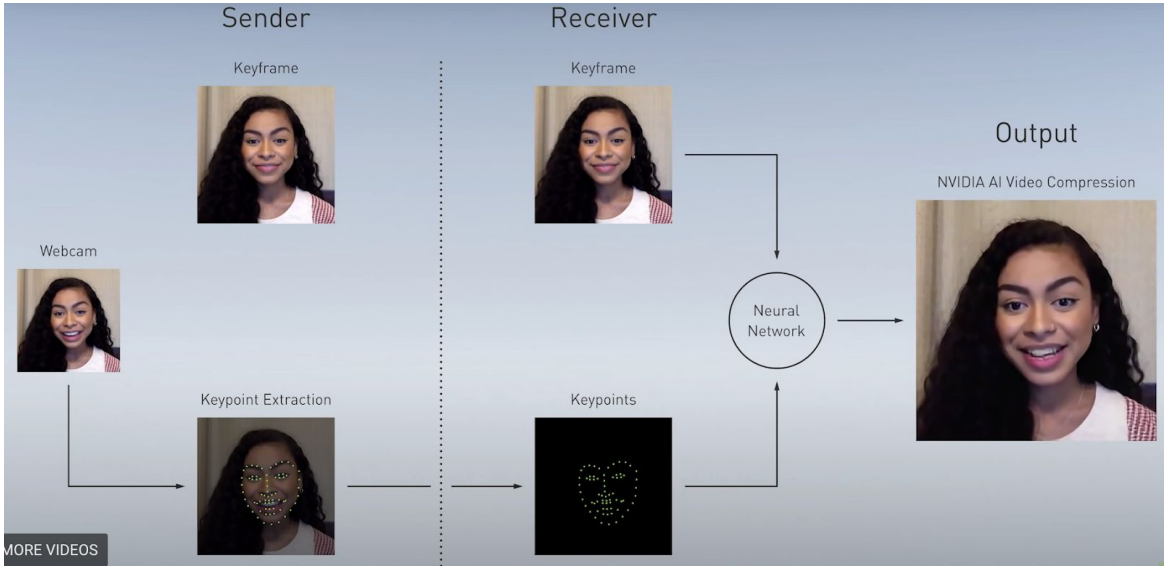
Conventional Pipeline Demo

Conclusions and future work

- Clearly better performance at much lower bandwidth requirements
- Future work:
  - Improve reliability as well as quality of detection/reconstruction
  - Reduce bandwidth using compression
  - Identify most impactful application areas: e.g., theater and digital puppetry, privacy
- Key points do not have to be geometric points
- Likely to be highly applicable given increase in publicly accessible computing resources (better smartphones, laptops, etc)

Recent update: NVIDIA Maxine

- Announced early October*
- Similar framework - use GPU and GANs for more realistic reconstruction



* Note: Our first blog post published in August

Thank You!

Check out the code at
https://github.com/shubhamchandak94/digital-puppetry