

CS3530: CN Project

Progress Report - Group 2

Resources collected

We found some repositories which are linked below:

1. Head tracker using saliency map:

<https://github.com/phananh1010/PanoSalNet>

- ## 2. QUIC implementation in python

<https://github.com/aiortc/aioquic>

Experimentation with QUIC

We experimented with the code and established connection between server-client and tested the verification using wireshark.

As mentioned above we used aioquic open source library to establish the QUIC connection between client and server for which we wrote small python codes with help of examples provided by them which required understanding of aioquic and asyncio. Now the next step is to modify these codes and stream a video using DASH.

Here is the output of packet capture:

Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
133	10.11175741	127.0.0.1	127.0.0.1	TCP	137	43028 -> 44133 [PSH, ACK] Seq=332 Ack=210 Win=2514 Len=0 Tsvail=119984028 TSecr=119983859
134	10.11287780	127.0.0.1	127.0.0.1	TCP	106	44133 -> 42028 [PSH, ACK] Seq=210 Ack=401 Win=26571 Len=38 Tsvail=119984030 TSecr=119984028
135	10.11898041	127.0.0.1	127.0.0.1	TCP	83	42028 -> 44133 [ACK] Seq=401 Ack=257 Win=2214 Len=0 Tsvail=119984030 TSecr=119984030
136	10.134457869	127.0.0.1	127.0.0.1	TCP	83	44133 -> 42028 [PSH, ACK] Seq=2182 Ack=814 Win=112 Len=5 Tsvail=119984051 TSecr=119983879
137	10.13447587	127.0.0.1	127.0.0.1	TCP	68	42028 -> 44133 [ACK] Seq=814 Ack=1897 Win=24571 Len=0 Tsvail=119984051 TSecr=119984011
138	10.134810978	127.0.0.1	127.0.0.1	TCP	87	42028 -> 44133 [PSH, ACK] Seq=401 Ack=257 Win=2214 Len=38 Tsvail=119984133 TSecr=119984030
139	10.260821123	127.0.0.1	127.0.0.1	TCP	68	44133 -> 42028 [ACK] Seq=257 Ack=420 Win=24571 Len=0 Tsvail=119984178 TSecr=119984133
140	10.260395066	127.0.0.1	127.0.0.1	QUIC	1244	Initial, CID=1154cde000b76756, SCID=cac4e717f80bfab7, PKB=8, CNPDT, PADDNG
141	10.260395333	127.0.0.1	127.0.0.1	QUIC	1244	Handshake, CID=cac4e717f80bfab7, SCID=aad11255954637c
142	10.27259809	127.0.0.1	127.0.0.1	QUIC	197	Protected Payload (KDP), DCID=cac4e717f80bfab7
143	10.27259342	127.0.0.1	127.0.0.1	QUIC	143	Handshake, DCID=aad11255954637c, SCID=aad11255954637c
144	10.29353716	127.0.0.1	127.0.0.1	QUIC	401	Protected Payload (KDP), DCID=aad11255954637c
145	10.29454112	127.0.0.1	127.0.0.1	QUIC	79	Protected Payload (KDP), DCID=aad11255954637c
146	10.29573535	127.0.0.1	127.0.0.1	QUIC	184	Protected Payload (KDP), DCID=aad11255954637c
148	10.294831456	127.0.0.1	127.0.0.1	QUIC	184	Protected Payload (KDP), DCID=aad11255954637c
149	10.29145128	127.0.0.1	127.0.0.1	QUIC	202	Protected Payload (KDP), DCID=cac4e717f80bfab7
150	10.295492972	127.0.0.1	127.0.0.1	QUIC	138	Protected Payload (KDP), DCID=cac4e717f80bfab7
151	10.29623024	127.0.0.1	127.0.0.1	QUIC	99	Protected Payload (KDP), DCID=cac4e717f80bfab7
152	10.29635235	127.0.0.1	127.0.0.1	QUIC	76	Protected Payload (KDP), DCID=cac4e717f80bfab7
153	10.296440199	127.0.0.1	127.0.0.1	QUIC	76	Protected Payload (KDP), DCID=aad11255954637c
154	10.29665208	127.0.0.1	127.0.0.1	QUIC	184	Protected Payload (KDP), DCID=aad11255954637c
155	10.296404359	127.0.0.1	127.0.0.1	QUIC	180	Protected Payload (KDP), DCID=cac4e717f80bfab7
156	10.29723497	127.0.0.1	127.0.0.1	QUIC	87	Protected Payload (KDP), DCID=aad11255954637c
157	10.297458014	127.0.0.1	127.0.0.1	QUIC	75	Protected Payload (KDP), DCID=aad11255954637c
158	10.297658401	127.0.0.1	127.0.0.1	QUIC	83	Protected Payload (KDP), DCID=cac4e717f80bfab7
159	10.298468005	127.0.0.1	127.0.0.1	TCP	505	44133 -> 42028 [PSH, ACK] Seq=814 Ack=814 Win=512 Len=397 Tsvail=119984162 TSecr=119984051
160	10.30024035	127.0.0.1	127.0.0.1	TCP	383	44133 -> 42028 [PSH, ACK] Seq=2224 Ack=814 Win=512 Len=315 Tsvail=119984218 TSecr=119984051

Frame 144: 1244 bytes on wire (9952 bits), 1244 bytes captured (9952 bits) on interface any, id 0
Linux capture v1
Internet Protocol Version 4, Src: 127.0.0.1, Dest: 127.0.0.1
User Datagram Protocol, Src Port: 4433, Dst Port: 4546
QUIC IETF
QUIC IETF

```

0000  00 00 03 04 00 06 50 50 50 50 50 50 00 00 00 00    ...S...
0010  45 00 04 cc 6e 3d 40 00 40 11 c9 71 7f 00 00 01    E...n...
0020  09 00 02 11 53 1a 04 3e 02 c0 00 00 00 00 00 00    ....N...
0030  01 00 04 ca 46 71 70 b4 f4 b3 07 08 46 a1 71 25 59    Np...M..Y
0040  65 7c 7e 00 00 00 00 00 00 00 00 00 00 00 00 00    Tel...N...
0050  0e 5f 54 a9 5c ba d1 80 6f 77 33 03 36 27 39 6c    .f...od G
0060  47 1a 12 38 bc 3e 30 36 39 70 07 0c 38 0e c7      ..B...g B
0070  0e 00 02 00 38 1f 71 d0 ba bc 44 86 47 39 6c     ..B...G
0080  50 02 90 9b 9f 47 c7 26 5c 80 73 f9 80 85 70     Z...w
```

Experimentation with Dash

Simulation steps

- a. Split the video and create manifest file using ffmpeg -

```
ffmpeg -i classroom.mp4 -map 0 -map 0 -c:a aac -c:v libx264 -b:v:0 800k  
-b:v:1 300k -var_stream_map "v:0,name:800k v:1,name:300k" -f dash  
-dash_segment_type mp4 -single_file 1 classroom_manifest.mpd
```

- b. We can simply stream it to a server using a HTML file where we can add a source for video as a manifest file created from the above command.

Used DASH to stream normal video to client from server using TCP - for basic understanding

Objectives of this project

1. Main objective is to replicate the modules, integrate and experiment with the architecture proposed by our reference paper.
2. Extended goal will be to experiment with different modules for performance gains.
3. Improve head tracking using saliency map of VR video and current head position to predict the head movement.

Our code and progress can be found here: <https://github.com/PushkaIM11/CN-Project>