Algorithm for Multiple stream synchronization

Algorithm is based on the master slave scheme where audio stream is master stream, because of two reasons humans are more senstive to audio than other media stream like video stream in our case and audio stream need less processing that video stream and probably reach earlier than video stream to receiver, Thus audio stream are considered as the master stream and preventive techniques are used to avoid the intra stream synchronization error in the audio stream and reactive techniques are used to avoid inter stream synchronization for video or other streams. Algorithm also consider the content change rate when we are dropping the MDUs for slave stream because of inter stream synchronization like if content change rate of a stream is less for a stream than even late MDUs can contain fresh information otherwise if content change rate is more than even small inter synchronization error would lead to inconvience in users view.

Here are some notations for algorithm described below

$t_j^g(i)$ = generation time of i^{th} MDU of j^{th} stream here j can be (audio, video, or any other stream).
$t_j^{tp}(i)$ = target playout time of i^{th} MDU of j^{th} stream.
$t_j^p(i)$ = playout time of the i^{th} MDU of j^{th} stream.
$t_{talk}^g(j)$ = generation time of the first MDU of j^{th} talkspurt.
$t_{talk}^p(j)$ = playout time of the first MDU of the j^{th} talkspurt.
$B_p(j)$ = playout delay for talkspurt j.
T_{err} = inter synchronization error of audio stream with slave stream(in case of more than one slave stream go with maximum inter sync error)
T_{audio} = timestamp of the audio stream's MDU.
T_{video} = timestamp of the videos stream's MDU.

Audio Stream

For the audio stream (consider as master stream), intra stream synchronization is important, so we evaluate the playout time for each MDU of the audio stream based on the playout delay and genration time of the MDU. (Here playout time does not mean MDU will played out on the calcuated time, this time is like a ordering in which we should queue it to audio processor, as we can't gurante the excat playout time due the load on local devices). Before that we have to calculate the playout delay before calucating the playout time for first MDU of the talkspurt.playout delay is calculated based on the netword delay estimated up to last received MDU of the previous talkspurt.

$$B_p(j) = \text{estimated\_average\_delay} + 3 * \text{estimated\_delay\_std\_delay}$$

$$B_p(j) = \max(B_p(j-1) + T_{err}, B_p(j))$$

T_{err} = inter synchronization error

For first MDU of the talkspurt the playout time is calculated

$$T^p_{talk}(j) = T^g_{talk}(j) + B_p(j)$$

For reset of the MDU of the talkspurt the playout time is calculated

$$T^p_a(i) = t^g_a(i) + B_p(j)$$

now we have the playout time of each MDU of the audio we can calculate the intra SPD (synchronization phase distortion) for audio stream.

$$\text{Intra strem SPD} = \sqrt{\Sigma_{n=2}^{n=N} [(t_a^p(n) - t_a^p(n-1)) - (t_a^g(n) - t_a^g(n-1))]^2}$$

Video Stream:

video stream as being an slave stream will follow the master stream.
Followings are some scnerio will be encounterd while playout the video's MDU stream
1. if newer frame than the video frame MDU's received has been played than received video frame is outdated and must be discared.
2. if video MDU's is ahead of audio stream, mean current playing audio stream is newer than video stream MDU considered for the playout i.e ($T_{audio} < T_{video}$) than we will look at the rate of change of video frame (if rate of change of video if less than the threshold we can display the frame other wise hold the frame, Here we used the content driven inter stream synchronization)
3. if audio stream MDU is ahead of video stream, mean curent playing audio stream is older than video stream MDU considered for the playout i.e ($T_{audio} > T_{video}$) than we look at the difference between the timestamp of audio and video MDU and compare it with threshold value *th* if difference is less than *th* video stream MDU will be playedout otherwise video stream MDU is virtualy droped and content change rate of video stream will be considered if content change rate is less than a threshold value than video stream MDU will be played (slow content rate changes means, even if MDU frame is late it might containt the fresh data) if content change rate is more than *threshold* value than video stream MDU will be dropped.

Content change rate will be embeded along with video stream MDU, audio stream MDU as most of times audio stream is ahead of the video stream so, content change rate in the audio stream will considered for take the decision of whether to drop the MDU or not.
If video stream MDU satisfy condition for playing out it's playout time will be decided as follow

$$\mathbf{t_v^{tp}(i) = t_{talk}^{p}(j) + [t_v^g(i) - t_{talk}^g(j)]}$$

talkspurt considered is currently playing audio talkspurt. As we have the playout time of the video stream we can calculate the inter syncrhonization error.

$$\mathbf{S_{err} = t_v^p(i) - t_v^{tp}(i)}$$

$$\mathbf{T_{err} = max(S_{err})}$$

we need to formulate the content change rate threshold, this threshold should be impacted by the different betweeen the timestamp of the video and audio stream, and avg. Different between audio and video stream MDU.

Δt_{inst} = actual timestamp difference cacluated between an audio and the video frame.

Δt_{avg} = average timstap difference between an audio and video frame.

$Th = (\Delta t_{avg} / \Delta t_{inst}) * \Delta V_{avg}$ (content rate change threshold)

this threshold value singnify if there is current timestamp different is less than average timestamp difference value than a bit more content variation can be accepted it, if timestamp different is more than the average timestamp difference than only small content variation is accepted.

Number of video frame dropped because later arrival of dropped virtually is counted and calculate the fracation of frame dropped from total frame received if this droped fraction and SPD (stream phase distoration) for inter stream synchronization exceed threshold than reciver should notify the sender for reduce the frame rate as receiver can't handel frames at current rate.
For any other stream the algorithm followed for video stream can be used.

Implementation :

for implementation socket programming in python is used, where each media stream is sent using a different UDP socket connection on different port. All media stream's MDU are packed in binary using the struct module of python, this binary of MDU will contain the followings.
1. sequence number.
2. timestamp.
3. media content change rate.
4. payload size
5. payload
6. generation time of media unit.

Sender side:

sender will capture the media streams and packetize them in the MDUs contain sequence number, timestamp, generation time, media content change rate, payload size and paylaod all these content are serelized in binary using the struct module of python which convert the python or c lanuage type struct in binary, now this binary will be sent to the receiver through the UDP connection, every media stream is send through different UDP connection. For each there is different thread handel the communication of the media stream, one specific thread is there to recieve the request for congetion control from the reciever. This is all from the sender side.
So flow at the sender side:
1. Each thread which is dedicated for the specific media stream capture the media stream, we are using opencv for caputure and process the video frame, pyaudio module is used for the audio capturing and capture, **video stream is converted to base64 notation for communicating over network.**
2. once we have captured the media stream, we have to pack the payload and other metadata of stream's MDU in packeta and send it, for this struct module is used. Each metadata is given a specific data type to pack it in the packet. .
3. after packing the payload and metdata in packet send the packet to the receiver UDP connection.
4. if recieve any congestion control flag from reciever than need to reduce the rate of sending the frame.

Receiver side:

receiver have separte UDP connection for each media stream, there is dedicated buffer for each media stream. Pick the media stream in order of there arrival and calculate the playout for each MDU as described in the algorithms, if there is more than threshold drop of frames due to inter sync error than send congesion control flag to receiver.
Here is flow at the receiver side.
1. each thread dedicated of meadia stream receive the media stream.
2. assign playout time to each MDU using the playout algorithm described.
3. calculate SPD for intra and inter stream, and find frame loss rate if any of SDP and frame loss ratio exceed the threshold congestion control flag should be sent to sender.
3. buffer overflow and idle condition need to also considered because of late arrival or late service rate.