ADASI Test

Candidate: David Esteban Imbajoa Ruiz

The complete code with video example media are available at:

https://github.com/deivy311/Test

Requirements

The requirements to run the code are:

Gstreamer 1.0

cairo

C++ 17

VLC >3.0.6

Setup

In order to set up, compile and run the code you have to follow the next steps:

- Install Gstreamer 1.0
- Install VLC >3.0.6
- Clone the repository:
    - Clone the repository using the next link:

      https://github.com/deivy311/Test

    - If you don't have the repository, you can download the zip file, attached to the email named ADASI_Test.zip and unzip it.
- Compile the code.
    - If your unzipping ADASI_test_just_code.zip, you have to add the video files in the folder "Files/" before to compile the code.

  Go to the folder where you clone the repository and run the next commands:

  mkdir build

  cd build

  cmake ../

  make

You should see something like the next image.

Ubuntu 22.04

For Linux these libraries must be installed

Follow the steps here: https://gstreamer.freedesktop.org/documentation/installing/on-linux.html?gi-language=c

And in addition, run the following commands:

sudo apt install libgirepository1.0-dev

sudo apt install gir1.2-gst-rtsp-server-1.0

auso apt install libgstreamer-plugins-good1.0-0 libgstreamer-plugins-good1.0-dev gstreamer1.0-plugins-good

sudo apt install gstreamer1.0-rtsp

sudo apt-get install libgstrtspserver-1.0-dev

sudo apt-get install libvdpau-dev vdpauinfo

sudo apt-get install libvdpau-va-gl1

Running the code

Go to the folder where you compile the code and run the executable file, for this a better explanation is in this section.

There are a total of 5 executables files, from de point_0 to point_5, each one of them has a different functionality. The parameters from Point 0 to Point 2_3 are the same, the only difference is the functionality of each one. And the parameters from Point 4 to Point 5 are the same.

In any case you must be placed in the root folder/build/bin in order to run the following commands

For **point_0, point_1 and point_2_3** the parameters are:

Source type: Defines if we want to stream the camera or a file, by default is **autovideosrc** , which refers to the camera, there are only two possible options:

videotestsrc or **autovideosrc**

Examples:

The input example for both cases are:

Camera case:

./point_1 -s videotestsrc

File case:

./point_0 -s autovideosrc

Host: Defines the host to stream by default is "127.0.0.1"

Examples:

Localhost case:

sudo ./point_0 -h 127.0.0.1

Port: Defines the port to stream by default is "5002"

Examples:

By default case:

sudo ./point_0 -p 5002

For **point_4, point_5 and point_2_3** the parameters are same as before but there a couple more:

RTSP port: Defines the RTSP port to stream by default is "5001"

Examples:

By default case:

sudo ./point_4 -p_rtsp 5001

RTSP video file path: Defines the RTSP video file path to stream by default is "**../Files/video_test_2.mp4**", these videos are in the Files folder and in the build/bin folder.

Examples:

By default case:

sudo ./point_5 -f ../Files/video_test_2.mp4

Results

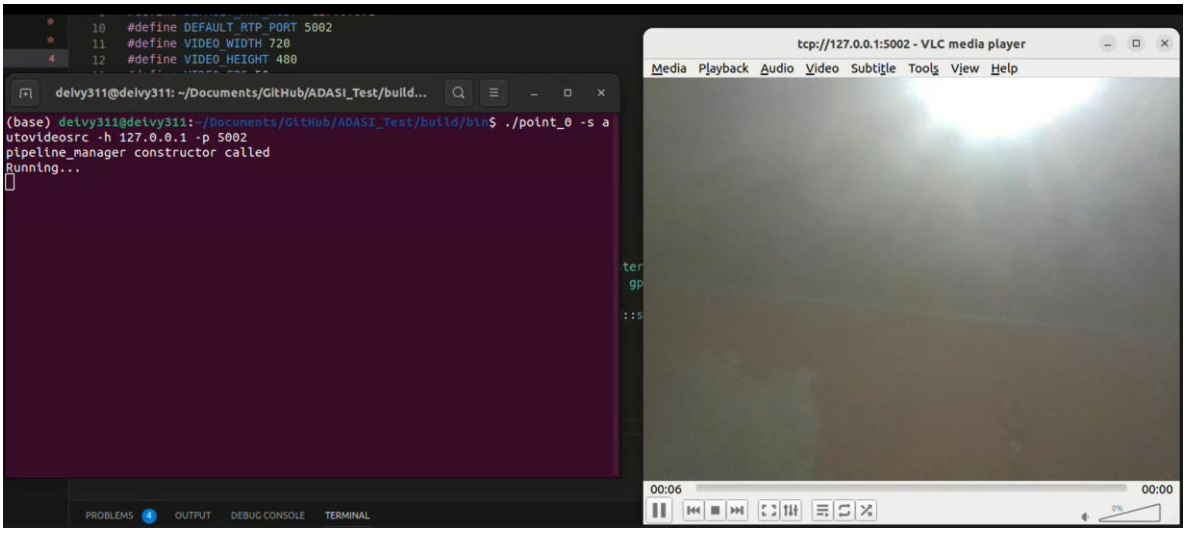
The executables **point_0 and point_1**, can show the results for the literal 1, it is a simple RTP pipeline where can stream a video file or a camera.

Case 1.1: RTP with no overlay

Given the next command input:

./point_0 -s autovideosrc -h 127.0.0.1 -p 5002

We can see the next results:



In order to see the results by yourself you have to open VLC and go to Media -> Open Network Stream and put the next address:

tcp://127.0.0.1:5002

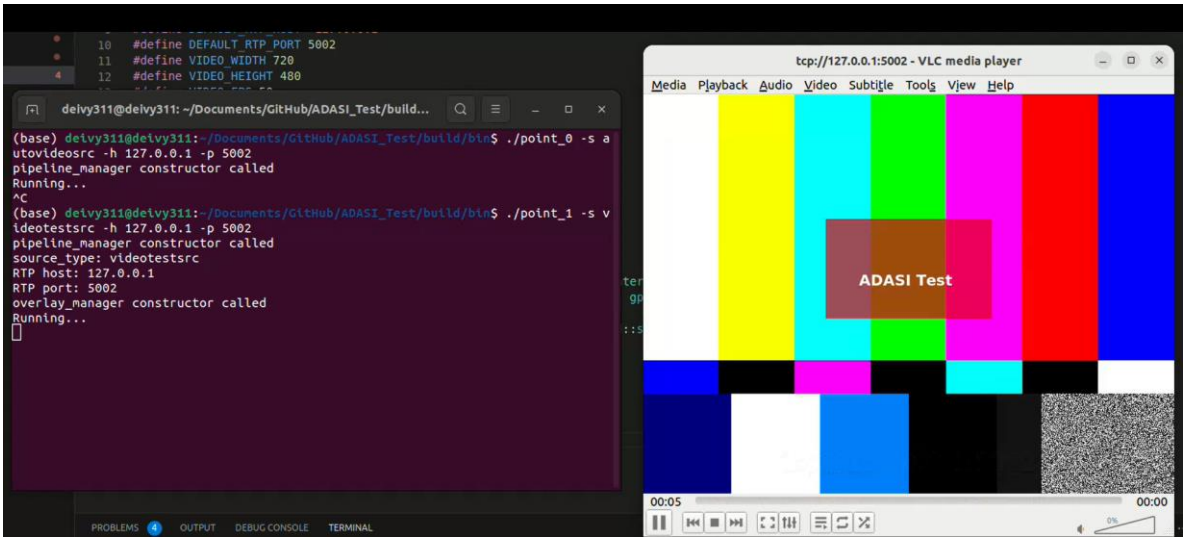
The video related to this is in the Folder Results, named point_1_no_overlay.webm which means it only streams video without any overlay.

Case 1.2: RTP with overlay, demo file

Given the next command input:

./point_1 -s videotestsrc -h 127.0.0.1 -p 5002

We can see the next results:



In order to see the results by yourself you have to open VLC and go to Media -> Open Network Stream and put the next address:

tcp://127.0.0.1:5002

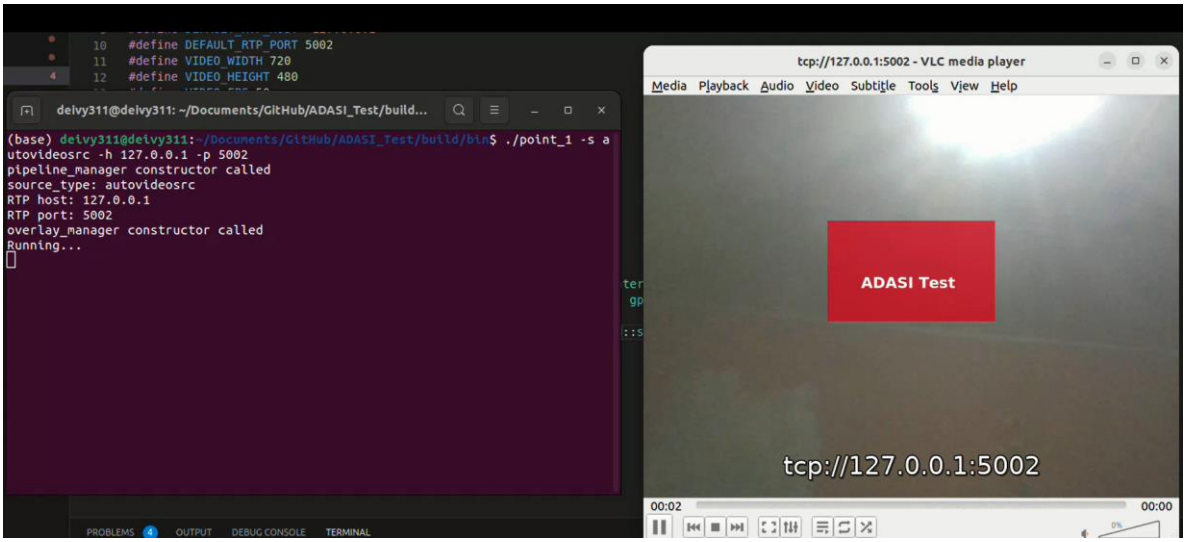
The video related to this is in the Folder Results, named point_1_overlay_file.webm which means it only streams video with overlay, in this case a red box.

Case 1.3: RTP with overlay, camera

Given the next command input:

./point_1 -s autovideosrc -h 127.0.0.1 -p 5002

We can see the next results:



In order to see the results by yourself you have to open VLC and go to Media -> Open Network Stream and put the next address:

tcp://127.0.0.1:5002

The video related to this is in the Folder Results, named point_1_overlay_camera.webm which means it only streams a camera video with overlay, in this case a red box.

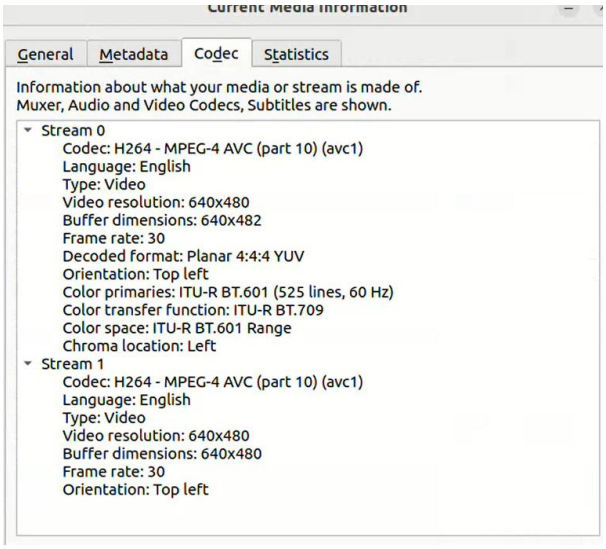
The executable **point_2_3**, can show the results for the literal 2 and 3, it is a RTP pipeline where can stream a video file or a camera and also can stream a video file or a camera with dynamic overlay and output the video in H264.

Case 2.1: RTP with dynamic overlay in H264 format video output, demo file

Given the next command input:

./point_2_3 -s videotestsrc -h 127.0.0.1 -p 5002

We can see the next results:



In order to see the results by yourself you have to open VLC and go to Media -> Open Network Stream and put the next address:

tcp://127.0.0.1:5002

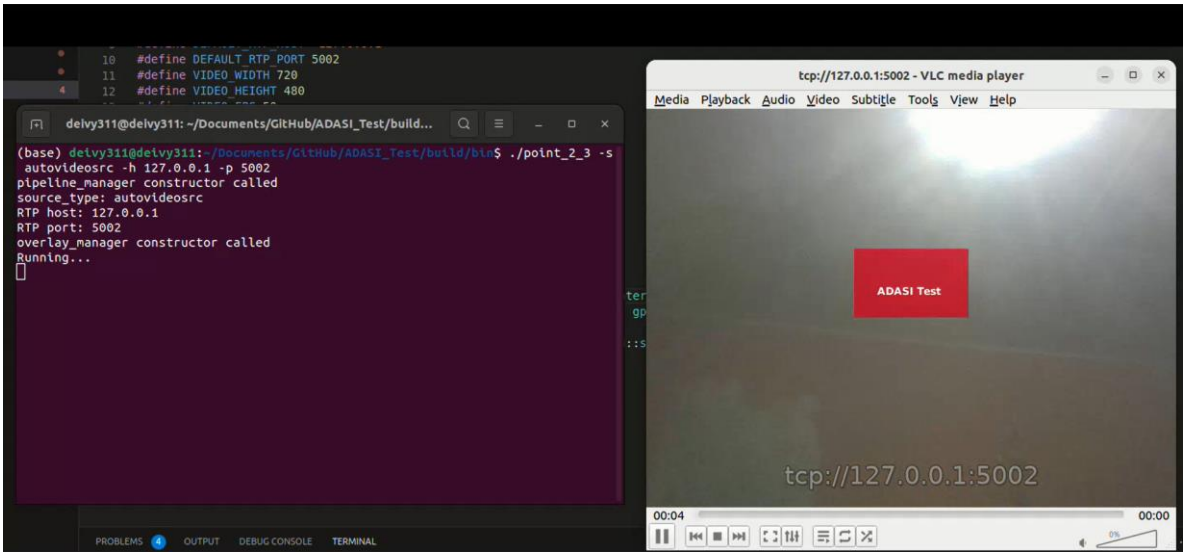
The video related to this is in the Folder Results, named point_2_3_dinamic_overlay_file.webm which means it only streams a video file with dynamic overlay and output the video in H264 format.

Case 2.2: RTP with dynamic overlay in H264 format video output, camera.

Given the next command input:

./point_2_3 -s autovideosrc -h 127.0.0.1 -p 5002

We can see the next results:



In order to see the results by yourself you have to open VLC and go to Media -> Open Network Stream and put the next address:

tcp://127.0.0.1:5002

The video related to this is in the Folder Results, named point_2_3_dinamic_overlay_camera.mp4 which means it only streams a camera video with dynamic overlay and output the camera video in H264 format.

The Executable **point_4**, can show the results for the literal 4, it is a RTSP pipeline over RTP where can stream a video file or a camera with dinamica overlay and output the video in H264.

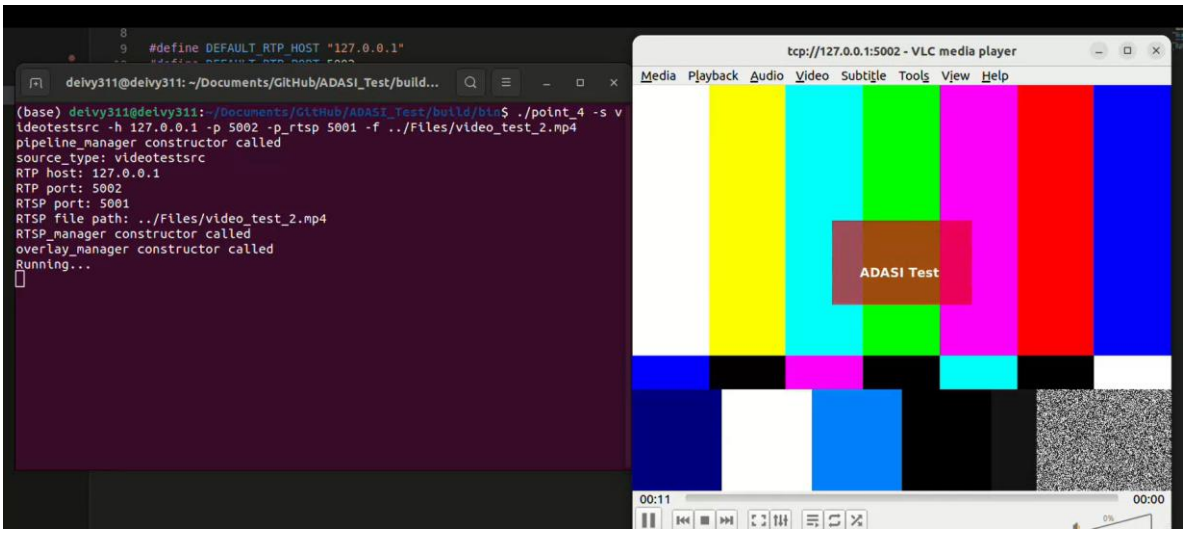
Case 4.1: RTSP with dinamic overlay in H264 format video output, demo file

Given the next command input:

./point_4 -s videotestsrc -h 127.0.0.1 -p 5002 -p_rtsp 5001 -f ../Files/video_test_2.mp4

We can see the next results:

RTP server

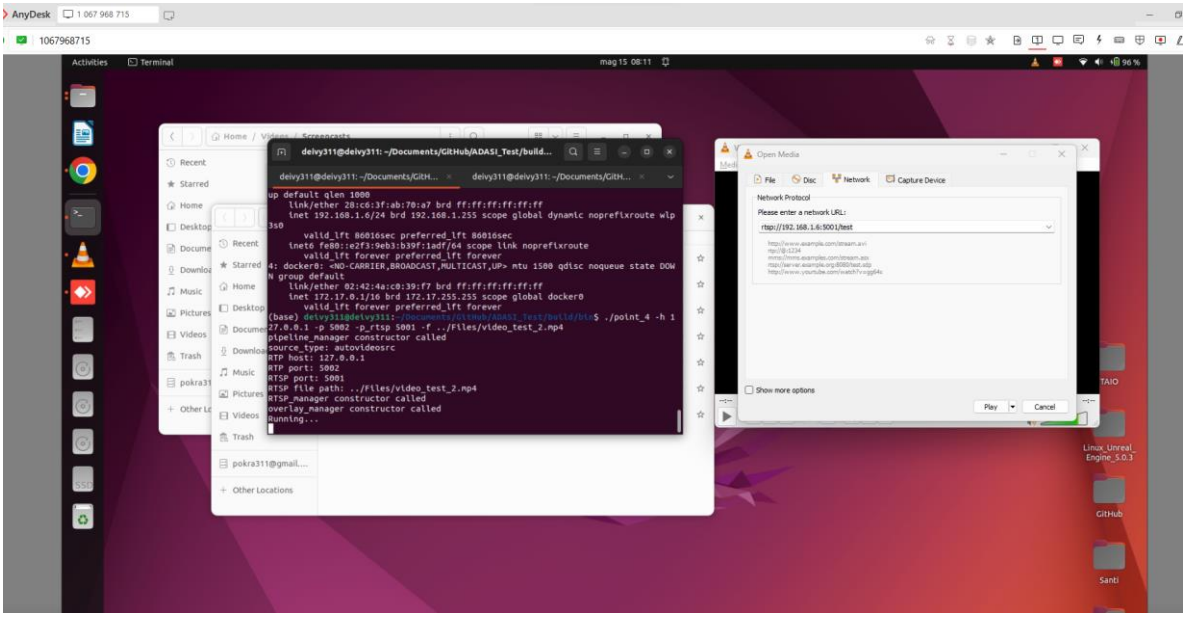


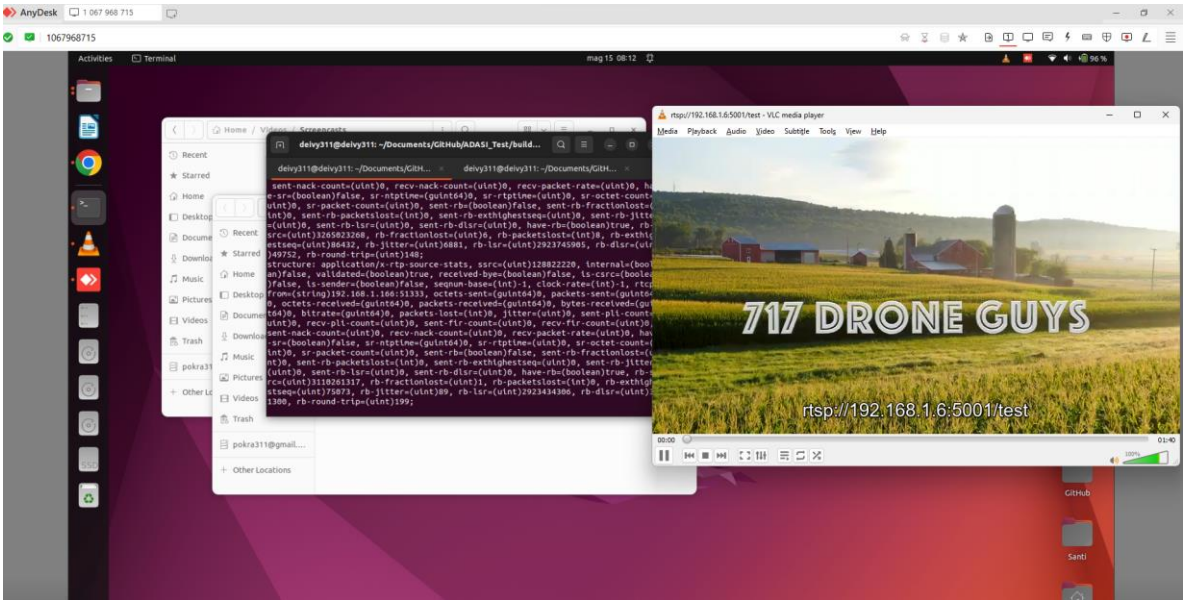
The video related to this is in the Folder Results, named point_4_RTP_dinamic_overlay_file.webm which means it streams a video file with dynamic overlay and output the video in H264 format.

The VLC command to see the results is:

tcp://127.0.0.1:5002

RTSP server


AnyDesk
1067968715
Activities
Terminal
mag 15 08:11
deivy311@deivy311: ~/Documents/GitHub/ADASI_Test/build...
Open Media
Network Protocol
Please enter a network URL:
rtsp://192.168.1.6:5001/test
Play
Cancel
Show more options
Linux_Unreal_Engine_5.0.3
GitHub
Santi


AnyDesk
1067968715
Activities
Terminal
mag 15 08:12
deivy311@deivy311: ~/Documents/GitHub/ADASI_Test/build...
rtsp://192.168.1.6:5001/test - VLC media player
Media
Playback
Audio
Video
Subtitle
Tools
View
Help
717 DRONE GUYS
rtsp://192.168.1.6:5001/test
GitHub
Santi

The video related to this is in the Folder Results, named point_4_RTSP_dinamic_overlay_file.mp4 which means it streams a video file with dinamic overlay and output the video in H264 format.

The VLC command to see the results is:

rtsp://192.168.1.6:5001/test # this is the ip of the same computer but viewed from

The Executable **point_5**, can show the results for the literal 5, it is a RTSP pipeline over RTP where can stream a video file or a camera with dynamic overlay and output the video in H264. An as additional feature the video streamed is stored.

Case 5.1: Storing file.

Given the next command input:

./point_5 -s autovideosrc

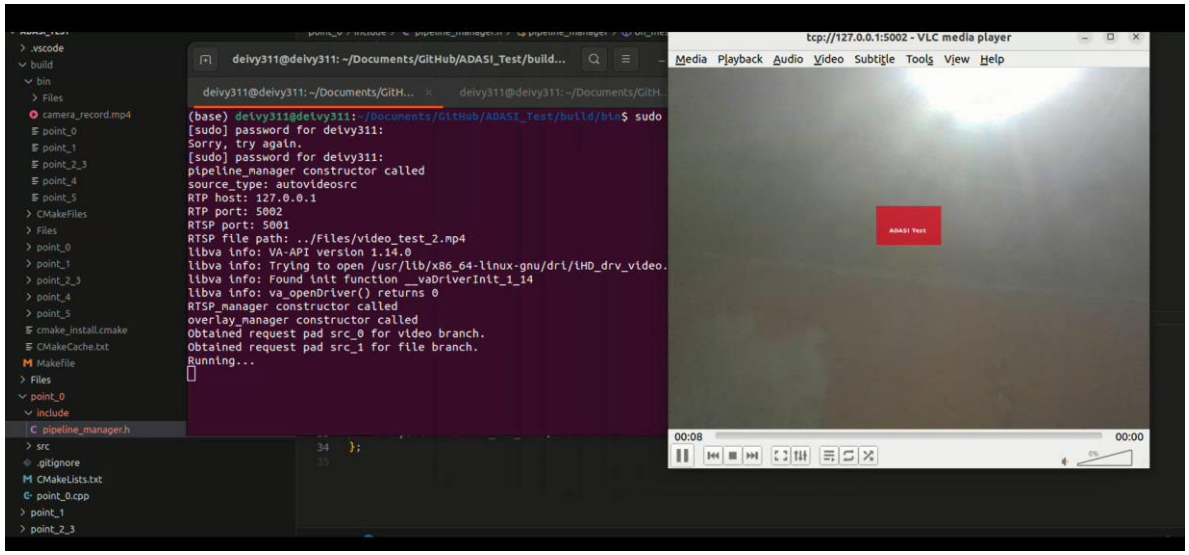
The video related to this is in the Folder Results, named point_4_RTP_dinamic_overlay_camera.mp4 which means it streams a camera video with dinamic overlay and output the video in H264 format.

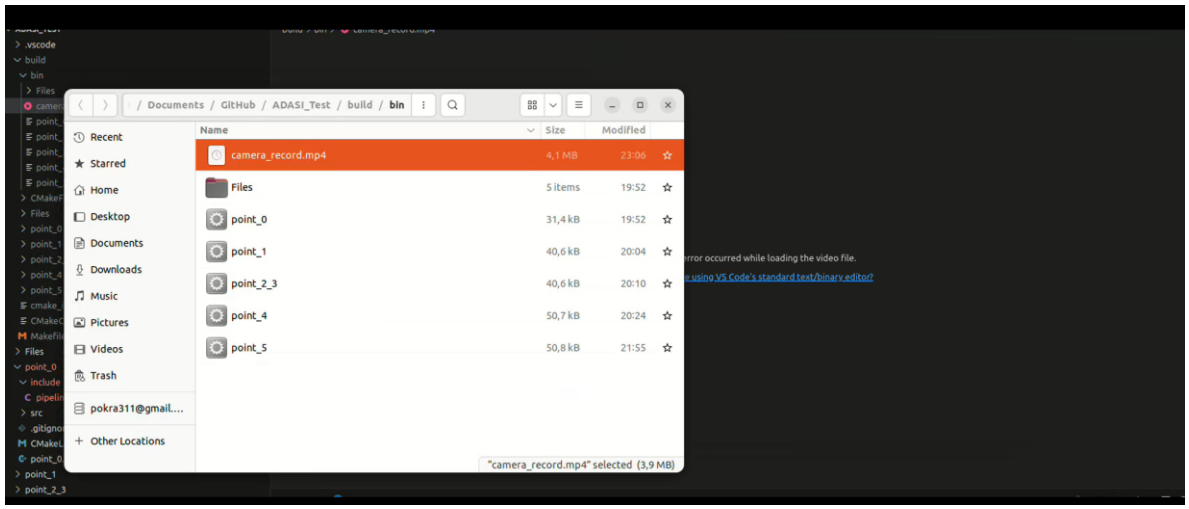
The VLC command to see the results is:

tcp://127.0.0.1:5002

rtsp://127.0.0.1:5001/test another

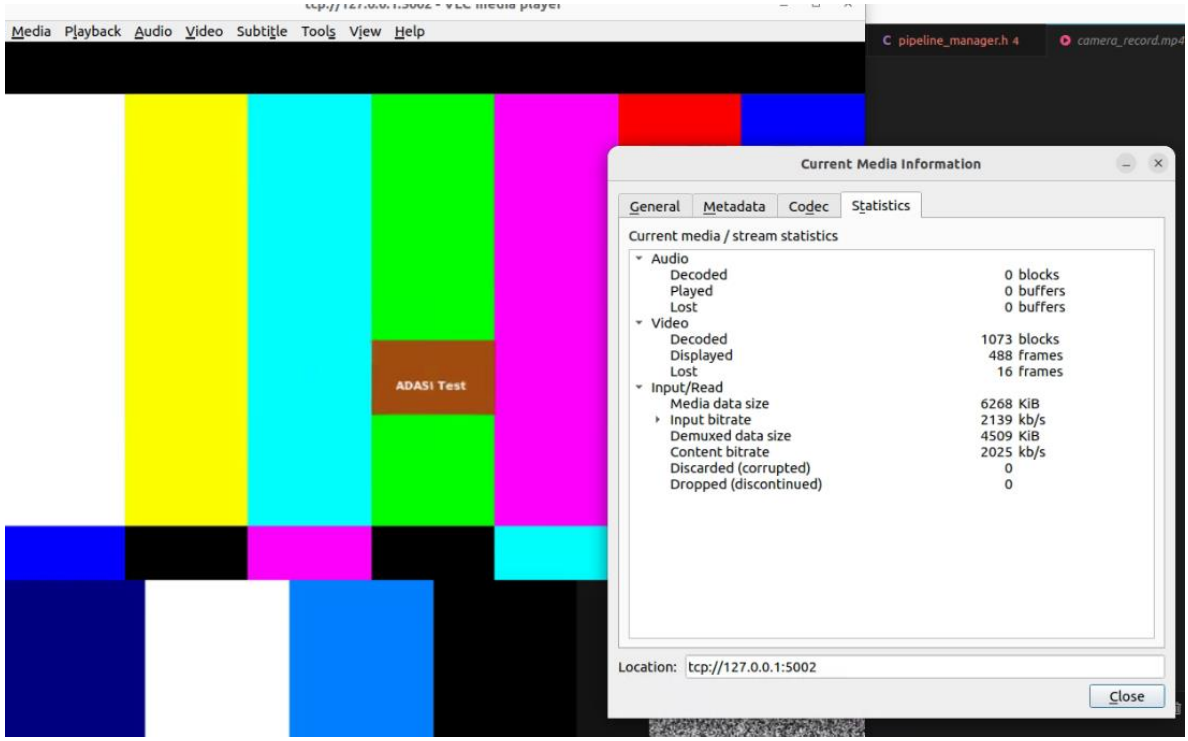
The video related to this is in the Folder Results, named point_5_RTSP_RTP_dinamic_overlay_camera.mp4 which means it streams a camera video with dynamic overlay and output the video in H264 format.





Metrics:

In the next image we can se the metrics computed for the point_4 and point_5


Media Playback Audio Video Subtitle Tools View Help
ADASI Test
pipeline_manager.h 4
camera_record.mp4
Current Media Information
General Metadata Codec Statistics
Current media / stream statistics
Audio
Decoded 0 blocks
Played 0 buffers
Lost 0 buffers
Video
Decoded 1073 blocks
Displayed 488 frames
Lost 16 frames
Input/Read
Media data size 6268 KiB
Input bitrate 2139 kb/s
Demuxed data size 4509 KiB
Content bitrate 2025 kb/s
Discarded (corrupted) 0
Dropped (discontinued) 0
Location: tcp://127.0.0.1:5002
Close