**For optimization:**

<https://www.wowza.com/community/questions/3023/advice-on-lowering-latency-with-gstreamer.html>

# **Sending Using Gstreamer:**

## Usb camera:

Save this as a (send\_cam.sh) and make it executable

To execute

> ./send\_cam.sh

#!/bin/sh

gst-launch-1.0 v4l2src device=/dev/video1 ! \

video/x-raw,width=640,height=480 ! \

timeoverlay ! \

tee name="local" ! \

queue ! \

autovideosink local. ! \

queue ! jpegenc! rtpjpegpay ! \

udpsink host=192.168.43.203 port= 7000

## Raspi camera:

Save this as a (send\_picam.sh) and make it executable

To execute

> ./send\_picam.sh

#!/bin/sh

raspivid -n -w 640 -h 480 -b 1200000 -fps 30 -t 0 -o - | \

gst-launch-1.0 -v fdsrc \

! h264parse \

! rtph264pay config-interval=1 pt=96 \

! udpsink host=192.168.43.203 port=5000

## To send both camera feed simultaneously:

Save this as a (send.sh) and make it executable

To execute

> ./send.sh

#!/bin/sh

/home/openquad/send\_cam.sh &

/home/openquad/send\_picam.sh &

# **Receive Using Gstreamer:**

## Usb camera:

Save this as a (receive\_cam.sh) and make it executable

To execute

> ./receive\_cam.sh

gst-launch-1.0 udpsrc port=7000 ! \

application/x-rtp,\

encoding-name=JPEG,payload=26 ! \

rtpjpegdepay ! jpegdec ! autovideosink

## Raspi camera:

Save this as a (receive\_picam.sh) and make it executable

To execute

> ./receive\_picam.sh

gst-launch-1.0 -v udpsrc port=5000 caps='application/x-rtp, media=(string)video, clock-rate=(int)90000, encoding-name=(string)H264' \

! rtph264depay \

! avdec\_h264 \

! videoconvert \

! autovideosink sync=false

## To receive both camera feed simultaneously:

Save this as a (receive.sh) and make it executable

To execute

> ./receive.sh

#!/bin/sh

/home/ark/receive\_cam.sh &

/home/ark/receive\_picam.sh

**NOTE:**

* Don’t use video 0 for raspi cam, use raspivid instead
* Ports number on sending and receiving end must be same
* To ensure low latency use low resolution
* udpsink host is the IP address of the receiver(Your computer generally)

# **Receiving Video feed in OpenCV:**

import cv2

import numpy as np

cap\_frontcam = cv2.VideoCapture("udpsrc port=7000 ! application/x-rtp,media=video,payload=26,clock-rate=90000,latency=0,buffer-duration=1,buffer-size=1,encoding-name=JPEG,framerate=30/1 ! rtpjpegdepay ! jpegdec ! videoconvert ! appsink", cv2.CAP\_GSTREAMER)

cap\_bottomcam = cv2.VideoCapture("udpsrc port=6000 ! application/x-rtp,media=video,payload=26,clock-rate=90000,encoding-name=JPEG,framerate=30/1 ! rtpjpegdepay ! jpegdec ! videoconvert ! appsink", cv2.CAP\_GSTREAMER)

while True:

\_ , frame\_frontcam = cap\_frontcam.read()

\_ , frame\_bottomcam = cap\_bottomcam.read()

stacked\_frame = np.hstack((frame\_frontcam , frame\_bottomcam))

cv2.imshow('Frontcam & Bottomcam' , stacked\_frame)

if cv2.waitKey(1)& 0xFF == ord('q'):

break

cv2.destroyAllWindows()

**NOTE:**

* Use different ports for different cameras
* Doesn’t work with raspi cams
* To ensure low latency use low resolution