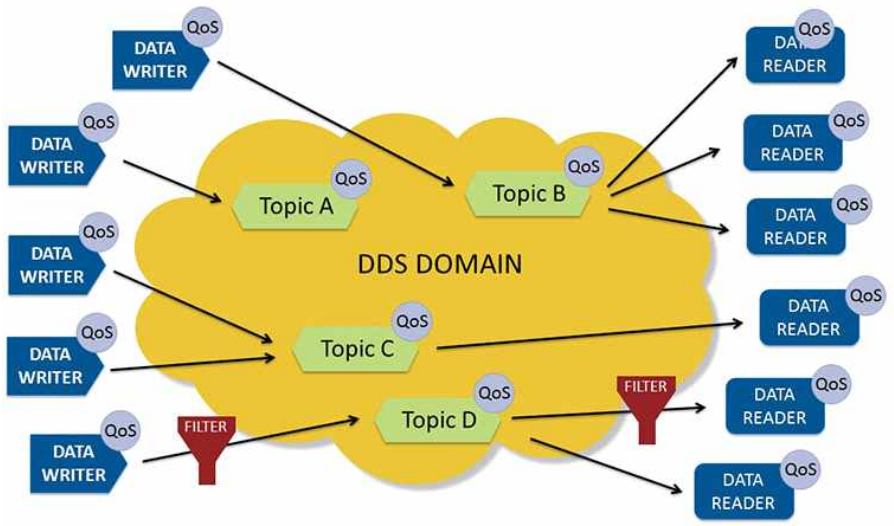
영상 전송을 위한 인코딩 및 통신 방식 비교 연구

1. 연구 배경 및 필요성

최근 영상 기술은 CCTV 감시, 자율주행차, 드론 영상 스트리밍, 원격 의료 진단 등 다양한 산업 분야에서 핵심 요소로 자리 잡고 있으며, 이러한 환경에서 고해상도 영상 데이터를 실시간으로 다수의 수신자에게 안정적으로 전송하는 기술은 필수 요건이 되고 있다. 그러나 이러한 실시간 고해상도 영상 전송은 다음과 같은 문제점들을 동반한다:

- 고해상도 영상으로 인한 대역폭 부담 증가
- 수신자 수 증가에 따른 전송 지연 증가
- 패킷 손실 발생 시 디코딩 오류 및 전체 프레임 재생 오류
- 인코딩 복잡도 증가로 인한 실시간성 저하

기존 소켓 기반 통신은 TCP 또는 UDP 프로토콜을 기반으로 하나의 수신자 또는 다수의 수신자를 동시에 처리할 수 있지만, 수신자 수가 증가할수록 서버 부하가 기하급수적으로 증가하고 멀티캐스트 환경에서 관리가 어렵다. 반면 DDS(Data Distribution Service)는 publish - subscribe 구조의 미들웨어로서 QoS 기반의 안정적인 멀티캐스트를 지원하고, 네이티브 분산 아키텍처에서 확장성이 뛰어나며 지연 보장, 신뢰성, 패킷 관리 등의 내장 QoS 정책을 활용할 수 있다.



[그림 1] DDS(Data Distribution Service)의 Publish-Subscribe 구조

따라서 본 연구에서는 다양한 영상 인코딩 방식(H.264, H.264 Intra-only, H.265, JPEG, AV1)과 전송 구조(소켓 vs DDS)의 성능을 비교하여, 실시간 고해상도 영상 전송 환경에서 최적의 조합을 도출하고자 한다.

2. 연구 목적

본 연구는 다음 세 가지 핵심 질문을 중심으로 설계되었다:
- 영상 인코딩 방식에 따라 실시간 전송 성능은 어떻게 달라지는가?
- 전통적인 소켓 방식과 DDS(Data Distribution Service) 방식은 다수 수신자 환경에서 어떤 차이를 보이는가?
- 실시간성과 안정성을 동시에 만족시키는 최적의 파라미터 조합은 무엇인가?

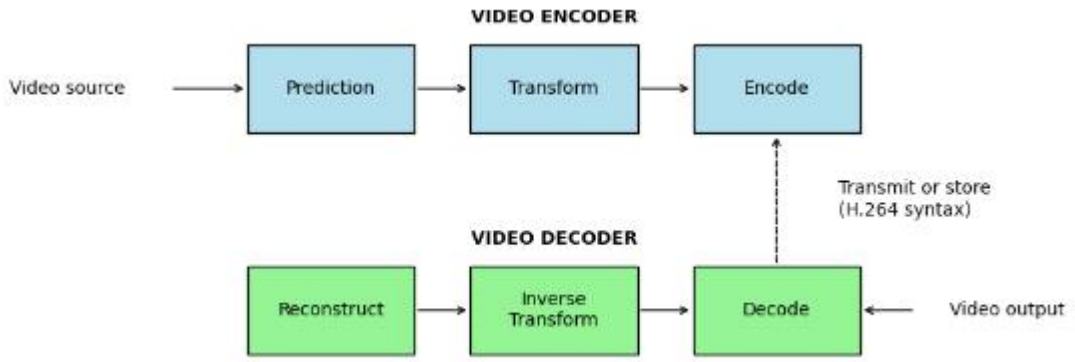
3. 실험 대상 영상

본 연구에서 실험에 사용된 영상은 총 용량이 약 486MB에 달하는 고해상도 영상으로, 해상도는 1920×1080(Full HD), 29.97FPS로 구성되어 있다. 전체 프레임 수는 약 6,162장으로 영상의 총 길이는 약 3분 25초(총 205.57초)에 해당한다. 이러한 영상은 실제 CCTV 감시, 자율주행 차량 탑재 카메라, 드론 스트리밍 등에서 사용되는 고해상도 데이터를 반영하기 위해 선정되었으며, 실시간 전송 실험에서 인코딩 효율성과 통신 구조의 성능 차이를 명확히 비교하기에 적절한 영상이다.

4. 인코딩 방식 기술 종류 및 설명

(1) H.264 / AVC (Advanced Video Coding)

H.264는 MPEG-4 Part 10 또는 AVC(Advanced Video Coding)라고도 불리며, 현재 전 세계에서 가장 널리 사용되고 있는 영상 압축 표준이다. 이 표준은 ITU-T 산하의 Video Coding Experts Group(VCEG)과 ISO/IEC 산하의 Moving Picture Experts Group(MPEG)이 공동으로 개발한 것으로, 예측, 변환, 부호화를 결합한 하이브리드(hybrid) 방식의 구조를 갖는다. 이를 통해 영상 내 시간적·공간적 중복을 효과적으로 제거함으로써 높은 압축 효율을 실현한다.

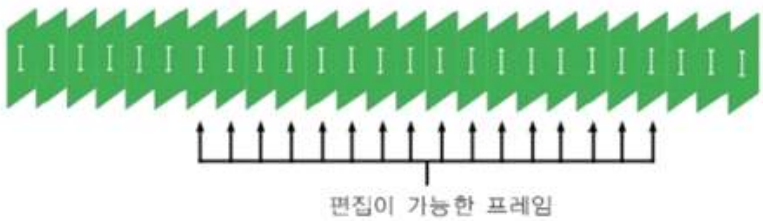


[그림 2] H.264/AVC 비디오 코덱의 인코더-디코더 구조

H.264의 프레임 구조는 크게 I-frame, P-frame, B-frame으로 구성된다. I-frame은 Intra-coded 프레임으로, 완전한 이미지 정보를 포함하고 있어 다른 프레임의 참조 없이 독립적으로 디코딩이 가능한 기준 프레임이다. P-frame은 Predictive-coded 프레임으로, 이전의 I-frame 또는 P-frame을 기준으로 예측하여 차이 정보를 저장함으로써 압축 효율을 높인다. B-frame은 Bi-predictive-coded 프레임으로, 이전과 이후의 프레임을 모두 참조하여 예측하는 방식으로 가장 높은 압축 효율을 제공한다.

P-frame과 B-frame은 Inter-frame 압축 기법을 기반으로 한다. Inter-frame 압축은 하나의 프레임이 이전 또는 이후의 프레임을 참조하여 변화된 정보만을 저장하는 방식으로, 연속된 프레임들 사이에서 중복되는 영상 정보를 제거하여 높은 압축률을 실현할 수 있도록 한다.

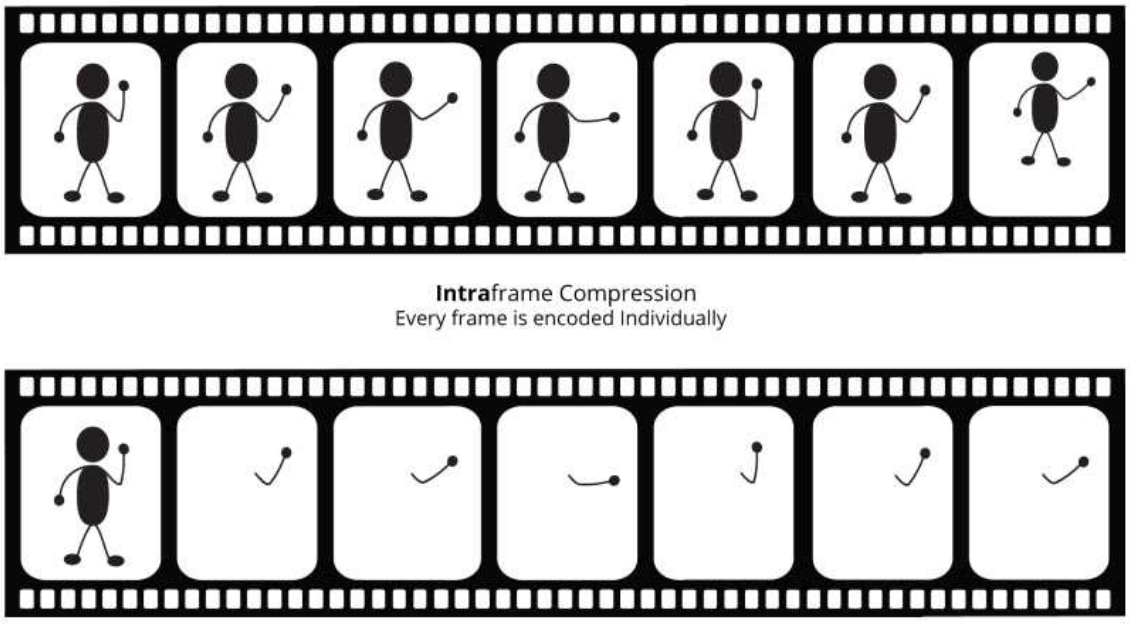
〈인터프레임〉



〈인트라프레임〉

[그림 3] 인터프레임(Inter-frame)과 인트라프레임(Intra-frame) 압축 방식 비교

이러한 프레임들은 일반적으로 GOP(Group of Pictures) 라는 구조로 묶여 압축의 기본 단위를 구성한다. GOP는 하나의 I-frame과 그 뒤를 따르는 여러 개의 P-frame과 B-frame으로 이루어지며, 예를 들어 GOP 길이가 12인 경우 I-frame은 매 13번째 프레임마다 등장하고, 그 사이에는 I B B P B B P B B I … 와 같은 패턴으로 프레임이 배열된다.



[그림 4] 인트라프레임과 인터프레임 압축 방식 비교

H.264는 다양한 기술적 구성 요소를 포함한다. 첫째, 예측 기법으로는 Intra 예측과 Inter 예측이 있다. Intra 예측은 블록 내의 공간적 상관관계를 활용하여 현재 블록을 예측하는 방식이며, Inter 예측은 시간축상의 연속 프레임 간 움직임을 보상(Motion Estimation and Compensation)하여 현재 프레임을 예측하는 방식이다.

둘째, 변환 및 양자화 단계에서는 4×4 또는 8×8 크기의 정수형 DCT(Discrete Cosine Transform)를 적용한 후, 양자화된 계수를 압축한다. 셋째, 엔트로피 부호화 단계에서는 CAVLC(Context-Adaptive Variable Length Coding) 또는 CABAC(Context-Adaptive Binary Arithmetic Coding) 방식을 사용하여 추가적인 압축을 수행한다. 특히 CABAC은 높은 압축 효율을 제공하지만 연산량이 많아 Main 또는 High 프로파일에서만 사용된다. 넷째, Loop Filtering은 블록 경계에서 발생할 수 있는 시각적 이질감을 완화하여 영상 품질을 개선하는 과정이며, 이때 주로 Deblocking Filter가 사용된다.

H.264의 가장 큰 장점은 기존 MPEG-2 대비 약 2배 이상의 압축 효율을 제공한다는 점이다. 또한 스마트폰, GPU, IP 카메라 등 다양한 플랫폼에서 하드웨어 가속이 폭넓게 지원되며, 실시간 인코딩 및 디코딩이 가능하여 라이브 스트리밍 환경에 적합하다. 오픈소스 구현체로는 대표적으로 x264, FFmpeg, OpenH264 등이 있으며, 이들은 누구나 자유롭게 사용·수정·배포할 수 있는 형태로 제공된다. 이러한 오픈소스들과 더불어 다양한 상용 구현체도 존재한다는 점은 실용성 측면에서 큰 강점으로 작용한다.

반면, H.264는 HEVC(H.265)나 AV1과 같은 최신 코덱에 비해 상대적으로 낮은 압축 효율을 보인다. 또한 GOP 설정이 복잡할 경우 지연(latency)이 증가할 수 있으며, 고해상도 및 고프레임 영상에서는 비트레이트 부담이 발생할 수 있다는 단점이 존재한다.

H.264는 현재 다양한 분야에서 폭넓게 활용되는 영상 압축 표준이다. 대표적으로 YouTube, Zoom, Skype, Netflix와 같은 인터넷 기반 영상 스트리밍 서비스에서 널리 사용되며, 고화질 콘텐츠를 안정적으로 전송하기 위한 핵심 기술로 자리잡고 있다. 또한, DVB(Digital Video Broadcasting)와 ATSC(Advanced Television Systems Committee) 등 지상파 디지털 방송 시스템에서도 주요 코덱으로 채택되고 있다. 이외에도 CCTV나 차량용 블랙박스 등 영상 저장 장치에서 효율적인 데이터 보관을 위해 활용되며, HLS(HTTP Live Streaming)와 MPEG-DASH(Dynamic Adaptive Streaming over HTTP) 같은 적응형 스트리밍 포맷에서도 실시간 전송 환경에 최적화된 압축 방식으로 사용된다.

(2) H.265 / HEVC (High Efficiency Video Coding)

H.265는 H.264의 차세대 영상 압축 표준으로, 동일한 화질을 유지하면서 최대 50%까지 비트레이트를 절감할 수 있도록 설계된 고효율 비디오 압축 기술이다. 본 표준은 ITU-T의 Video Coding Experts Group(VCEG)과 ISO/IEC 산하의 Moving Picture Experts Group(MPEG)이 공동으로 개발하였으며, 특히 4K 및 UHD급 영상 전송을 위한 핵심 기술로 자리잡고 있다.

[그림 5] H.264와 H.265 인코딩 방식 차이

기술적인 구조 측면에서 H.265는 기존 H.264에서 사용되던 16×16 크기의 매크로블록(macroblock) 대신, 최대 64×64 크기의 CTU(Coding Tree Unit)를 기본 단위로 사용한다. CTU는 다시 Coding Unit(CU), Prediction Unit(PU), Transform Unit(TU)으로 계층적으로 분해되며, 더욱 유연한 블록 분할 구조를 통해 영상의 세부 정보를 보다 정밀하게 표현할 수 있다.

Intra 예측 방식에서는 최대 35개의 예측 방향을 지원하여, 기존 H.264의 9개 방향에 비해 공간적 예측의 정밀도가 대폭 향상되었다. Inter 예측, 즉 움직임 보상(Motion Estimation) 기술 또한 강화되어 MERGE 모드, AMVP(Advanced Motion Vector Prediction) 모드 등을 통해 보다 정교한 움직임 예측이 가능하다. 또한, 가변 블록 크기 기반의 모션 보상과 서브픽셀 정밀도를 지원함으로써 압축 효율을 높이고 움직임이 많은 영상에서도 품질 저하를 최소화할 수 있다.

변환 및 양자화 단계에서는 4×4부터 32×32까지 다양한 크기의 Integer DCT(Discrete Cosine Transform, 이산 코사인 변환)를 사용하며, 확장된 양자화 파라미터(QP) 범위를 통해 다양한 비트레이트 조건에서도 유연한 압축이 가능하다. 엔트로피 부호화는 CABAC(Context-Adaptive Binary Arithmetic Coding)을 전용 방식으로 채택하여, 이전 표준 대비 향상된 압축 효율을 제공한다.

병렬 처리 효율을 극대화하기 위해 H.265는 Tiles와 WPP(Wavefront Parallel Processing) 기술을 도입하였다. 이러한 기술들은 인코딩 병렬화를 지원하여 고해상도 영상에서도 빠른 처리를 가능하게 한다.

H.265의 주요 장점은 고압축 효율이다. 동일한 화질 조건에서 H.264에 비해 약 40~50%의 비트레이트 절감 효과를 얻을 수 있으며, 4K 또는 8K와 같은 초고해상도 영상의 저장 및 전송에 특히 적합하다. 또한, 더 넓은 예측 방향과 정밀한 블록 분할 구조를 통해 세부 영상 묘사력이 향상되었으며, Loop Filter 및 Sample Adaptive Offset(SAO)과 같은 후처리 기술을 통해 블로킹 및 링잉(고주파 잡음) 현상을 효과적으로 억제할 수 있다.

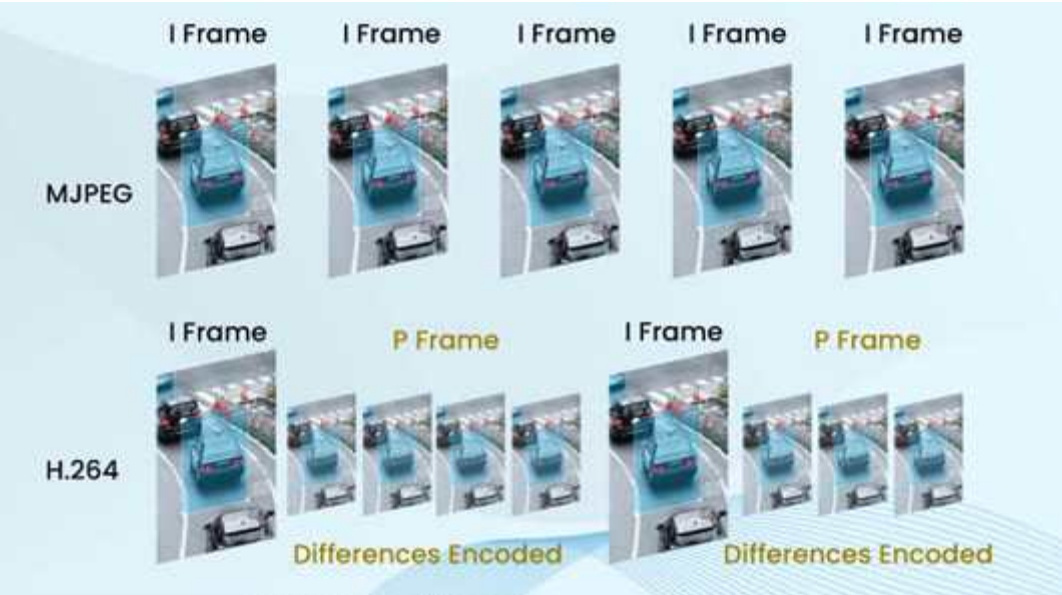
반면, H.265는 인코딩과 디코딩 복잡도가 크게 증가하였다. 특히 인코딩 과정은 H.264에 비해 약 2~10배 이상의 연산 자원을 필요로 하며, 고성능의 CPU 또는 GPU를 요구하는 경우가 많다. 또한, 상용화에 있어서 MPEG LA, HEVC Advance, Velos Media 등 복수의 특허 풀로 인해 라이선스 구조가 복잡하고 비용 부담이 크다는 단점이 존재한다.

H.265는 다양한 산업 분야에서 활용되고 있다. 대표적으로 UHD/4K 방송 분야에서는 ATSC 3.0, DVB-T2 등의 지상파 및 위성 방송 표준에 채택되고 있으며, Netflix와 Amazon Prime Video와 같은 OTT 스트리밍 서비스에서도 HEVC를 지원하는 디바이스에 한해 콘텐츠를 제공하고 있다. 또한, 고해상도 CCTV와 같은 보안 및 감시 시스템에서 저장 공간 절약을 위해 널리 사용되며, 스마트폰이나 드론과 같은 모바일 카메라 장치에도 하드웨어 인코더 형태로 내장되어 있다.

하드웨어 및 소프트웨어 지원 측면에서는 인텔 6세대 이상의 CPU에서 Quick Sync Video 기능을 통해 H.265 인코딩을 지원하며, NVIDIA Maxwell 이후 세대에서는 NVENC, AMD에서는 VCN을 통해 가속 처리가 가능하다. 오픈소스 소프트웨어 중에서는 FFmpeg, GStreamer, VLC 등에서 libx265 플러그인을 통해 H.265 인코딩 기능을 제공하고 있다.

(3) JPEG / MJPEG

JPEG는 DCT를 기반으로 한 손실 압축 방식으로, 영상을 프레임 단위로 개별 처리하는 구조를 가진다. MJPEG(Motion JPEG)은 이러한 JPEG 프레임을 연속적으로 저장하거나 전송하여 마치 동영상처럼 구현한 방식이다. 각 프레임이 독립적으로 압축되고 처리되기 때문에 구조가 단순하며, 예측이나 참조 프레임이 없는 것이 특징이다.



[그림 6] MJPEG와 H.264의 프레임 압축 방식 비교

기술적으로는 입력 RGB 영상을 YCbCr 색공간으로 변환한 후, DCT 변환을 적용하고 양자화 및 엔트로피 부호화(Run-Length Encoding 및 허프만 부호화)를 통해 압축한다. 이 구조는 실시간 처리가 용이하며, 하드웨어 가속이 비교적 쉽게 구현된다는 장점이 있다. 다만, 각 프레임이 독립적으로 처리되기 때문에 영상 내 블로킹 아티팩트(블록 경계 왜곡)가 발생할 가능성이 있으며, 텍스트나 도표가 포함된 영상에서 시각적인 품질 저하가 나타날 수 있다.
JPEG 계열 포맷의 확장형으로는 여러 가지가 존재한다. JPEG-LS는 무손실 또는 근손실 압축을 지원하는 방식으로, 주로 의학 영상과 같은 고정밀 이미지에 사용된다. JPEG 2000은 DWT(Discrete Wavelet Transform) 기반의 압축 기술로 무손실 압축

이 가능하며, JPEG 대비 더 높은 화질과 유연한 압축 제어가 가능하다. JPEG-XS는 초저지연 전송을 목적으로 설계된 포맷으로, 방송 및 산업용 실시간 영상 전송 환경에 적합하다.

JPEG 및 MJPEG 방식은 CCTV, 웹캠, 드론 영상 등 실시간 처리가 중요한 환경에서 활용되며, 특히 의료 영상 저장(예: CT, MRI 등)과 디지털 시네마(DCP)에서도 영상의 개별 프레임 품질을 중시하는 용도에 적합하게 사용된다.

(4) AV1 (AOMedia Video 1)

AV1은 AOMedia(Alliance for Open Media)에서 개발한 차세대 고효율 영상 코덱으로, Royalty-Free를 목표로 설계되었다. 기존의 HEVC(H.265)에 비해 약 20~30% 향상된 압축 효율을 제공하며, 고화질 영상을 보다 낮은 비트레이트로 전송하거나 저장할 수 있는 것이 주요 강점이다.

[그림 7] 영상 코덱별 인코딩 품질 비교 (H.264 / HEVC / AV1)

기술적으로 AV1은 Hybrid 코덱 구조를 기반으로 하며, 기존 방식보다 확장된 예측 도구를 도입하고 있다. 대표적인 예측 기술로는 OBMC(Overlapped Block Motion Compensation), Warped Motion, CfL(Chroma from Luma), CDEF(Constrained Directional Enhancement Filter) 등이 있으며, 다양한 움직임과 색상 정보에 유연하게 대응할 수 있다. AV1은 최대 7개의 참조 프레임을 지원하며, 기존 코덱이 사용하는 이진 산술 부호화 대신 비이진 기반의 ANS(Asymmetric Numeral Systems) 계열 엔트로피 부호화를 사용함으로써 효율성과 자유도를 동시에 높인다. 또한 Loop Restoration과 같은 후처리 기법을 통해 영상 품질 향상과 블록 기반 압축의 시각적 문제를 보완한다.

AV1의 가장 큰 장점은 무료로 사용 가능한 오픈 포맷이라는 점과 높은 압축 효율에 있다. 이는 WebM, DASH(Dynamic Adaptive Streaming over HTTP) 기반의 적응형 스트리밍 환경에 매우 적합하며, 특히 네트워크 대역폭이 제한된 환경에서 강점을 발휘한다.

그러나 AV1은 매우 복잡한 인코딩 구조를 가지고 있어 인코딩 속도가 느리고, 실시간 영상 전송에는 부적합하다는 한계가 있다. 실제 구현에서도 실시간 환경보다는 저장형 콘텐츠 스트리밍에 초점이 맞추어져 있다.

현재 AV1은 YouTube 4K 콘텐츠 및 Netflix의 절전 모드 스트리밍에서 사용되고 있으며, Google Chrome, Mozilla Firefox, Android WebRTC 등 주요 플랫폼에서도 지원되고 있다.

(5) 프레임 단위 전송 방식

프레임 단위 전송 방식은 전체 영상을 개별 프레임 단위로 나누어 독립적으로 전송하는 방식이다. 각 프레임은 완전한 디코딩이 가능한 단위로 유지되며, 전송 중 일부 프레임이 손실되더라도 전체 스트림의 재생에는 치명적인 영향을 미치지 않는다는 장점이 있다. 이러한 구조는 DDS와 같은 퍼블리시-서브스크라이브 기반 통신 환경에서 프레임 손실 여부를 명확히 식별할 수 있도록 하며, 프레임 단위의 성능 분석을 가능하게 한다.

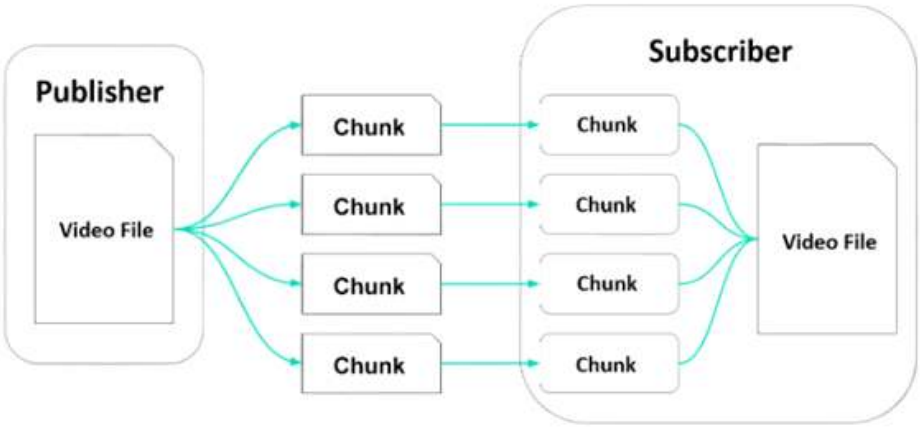
[그림 8] 영상 프레임 단위 분할 및 전송 구조

다만, 프레임 간 예측이나 압축이 적용되지 않을 경우 전체적인 압축 효율은 낮아질 수 있으며, 비트레이트가 증가함에 따라 대역폭 소비가 커질 수 있다는 단점이 존재한다. 또한 프레임별 메타데이터를 개별 관리해야 하므로 시스템 설계 시 구조적 고려가 필요하다.

이 방식은 주로 실시간 CCTV 영상 전송, 드론 감시 시스템, 차량 내 객체 탐지와 같이 프레임 손실에 강인한 처리가 필요한 실시간 영상 처리 시스템에 적합하게 사용된다. 특히 디코딩 속도가 빠르고, 단일 프레임의 유실이 전체 영상 복원에 큰 영향을 주지 않기 때문에 에러 복원력이 요구되는 환경에서 유리하다.

(6) 청크 단위 전송 방식

청크 단위 전송 방식은 인코딩된 영상 데이터를 일정한 바이트 크기의 조각(청크, chunk)으로 나누어 전송하는 구조이다. 각 청크는 전송 순서, 프레임 내 위치, 마지막 조각 여부 등을 나타내는 메타데이터와 함께 전송되며, 수신 측에서는 이를 기반으로 전체 프레임 혹은 전체 영상을 재구성한다. 본 방식은 프레임보다 더 세분화된 단위로 데이터를 분할하여 처리하므로, 대용량 영상의 전송 시에도 유연한 흐름 제어가 가능하다.



[그림 9] 청크 단위 기반 영상 전송 과정

청크 단위 구조는 특히 DDS와 같은 미들웨어의 QoS 정책과 잘 결합된다. 예를 들어, Reliability, History, Lifespan과 같은 정책을 통해 데이터 손실 복구, 반복 전송, 유효 시간 제어 등이 가능해지며, 네트워크 혼잡 상황에서도 데이터의 안정적인 수신을 보장할 수 있다.

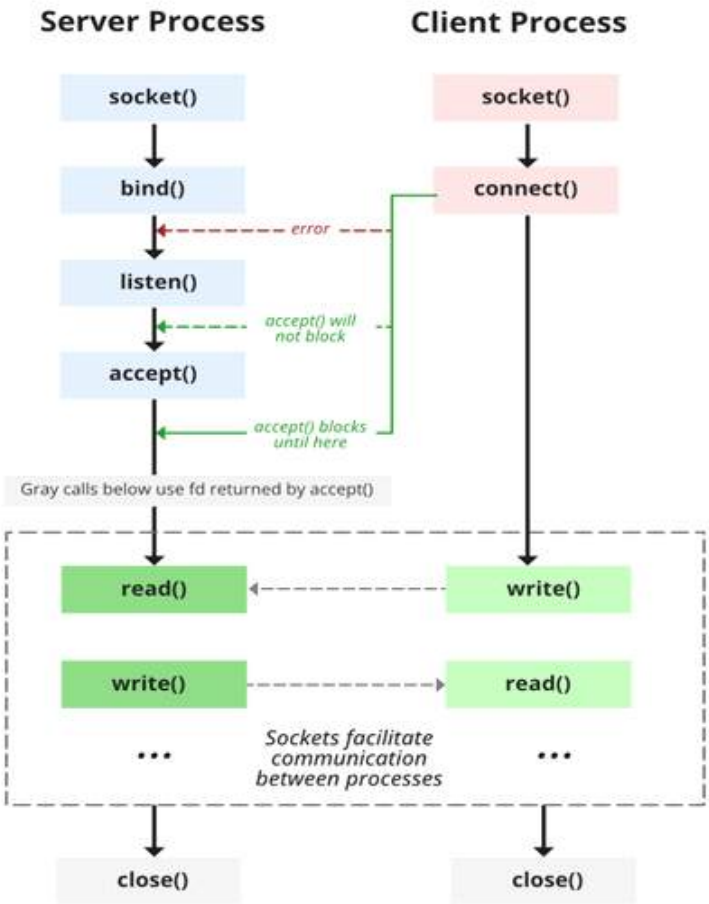
장점으로는 네트워크 환경이 불안정한 상황에서도 패킷 단위로 유실 검출 및 복구가 가능하며, 일정한 전송 단위를 유지함으로써 전송 지연을 정밀하게 제어할 수 있다는 점이 있다. 반면, 전송 지연이 누적될 경우 프레임 재구성이 지연되며, 모든 청크가 도달하지 못한 경우 해당 프레임 전체가 복원되지 못하는 단점이 존재한다.

이 방식은 UAV 영상 전송, 고해상도 지도 스트리밍, 군사/항공 미션 데이터 공유 등과 같이 안정성 중심의 고신뢰 네트워크 전송 환경에서 주로 사용된다. 특히, 전송의

신뢰성과 복원력이 최우선시되는 시스템에서 청크 기반 구조는 매우 효과적인 대안이 된다.

5. 통신 구조 비교: 소켓 vs DDS

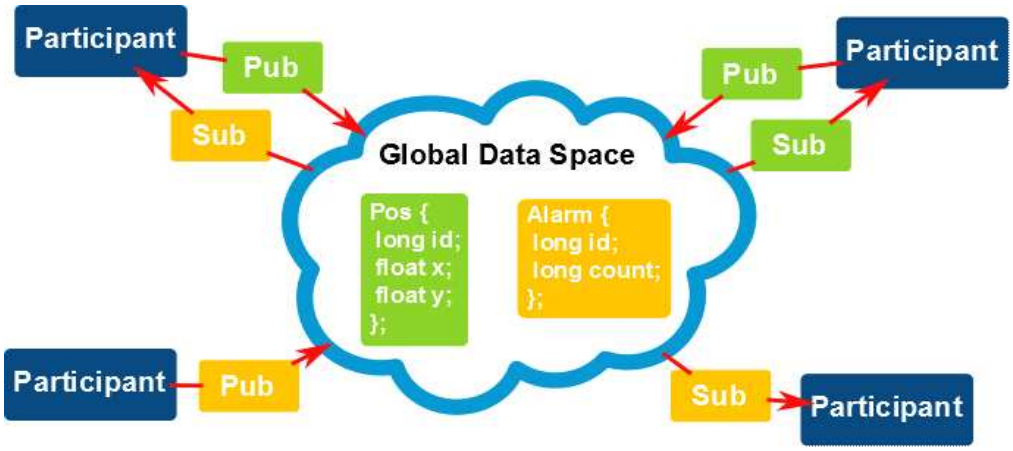
영상 전송을 위한 통신 구조는 크게 소켓 기반 방식과 DDS(Data Distribution Service) 기반 방식으로 구분할 수 있다. 소켓 기반 통신은 전통적으로 사용되어 온 TCP 또는 UDP 프로토콜을 활용하는 클라이언트-서버 모델 구조를 따른다. 이 방식은 구현이 상대적으로 단순하며, Java 나 C++, Python와 같은 네이티브 언어에서 직접 제어가 가능하다는 장점이 있다. 특히 1:1 또는 제한된 수의 수신자에게 데이터를 전송하는 환경에서는 비교적 효율적인 방식이다.



[그림 10] 소켓 기반 통신의 클라이언트-서버 구조

그러나 수신자 수가 증가함에 따라 서버의 부하가 선형적으로 증가하며, 멀티캐스트 구성 시 브로드캐스트 처리를 수동으로 설정해야 하는 등 확장성 측면에서 한계를 가진다. 또한, UDP 기반 통신에서는 패킷 손실 복구 메커니즘이 부족하며, 지연 보장, 신뢰성, 우선순위와 같은 QoS(Quality of Service) 기능을 자체적으로 제공하지 않는다는 단점이 존재한다.

반면, DDS는 RTPS(Real-Time Publish-Subscribe) 프로토콜을 기반으로 하는 publish - subscribe 아키텍처를 채택한 미들웨어 기술이다. DDS는 IDL(Interface Definition Language)을 이용하여 메시지 구조를 자동으로 생성할 수 있으며, QoS 기반의 세부적인 전송 정책 설정이 가능하다는 점에서 고신뢰, 고확장성 통신을 지원한다. 예를 들어, Reliable/Best-effort 신뢰 수준, Deadline, History, Durability 등 다양한 QoS 프로파일을 통해 전송 특성을 세밀하게 조절할 수 있다.



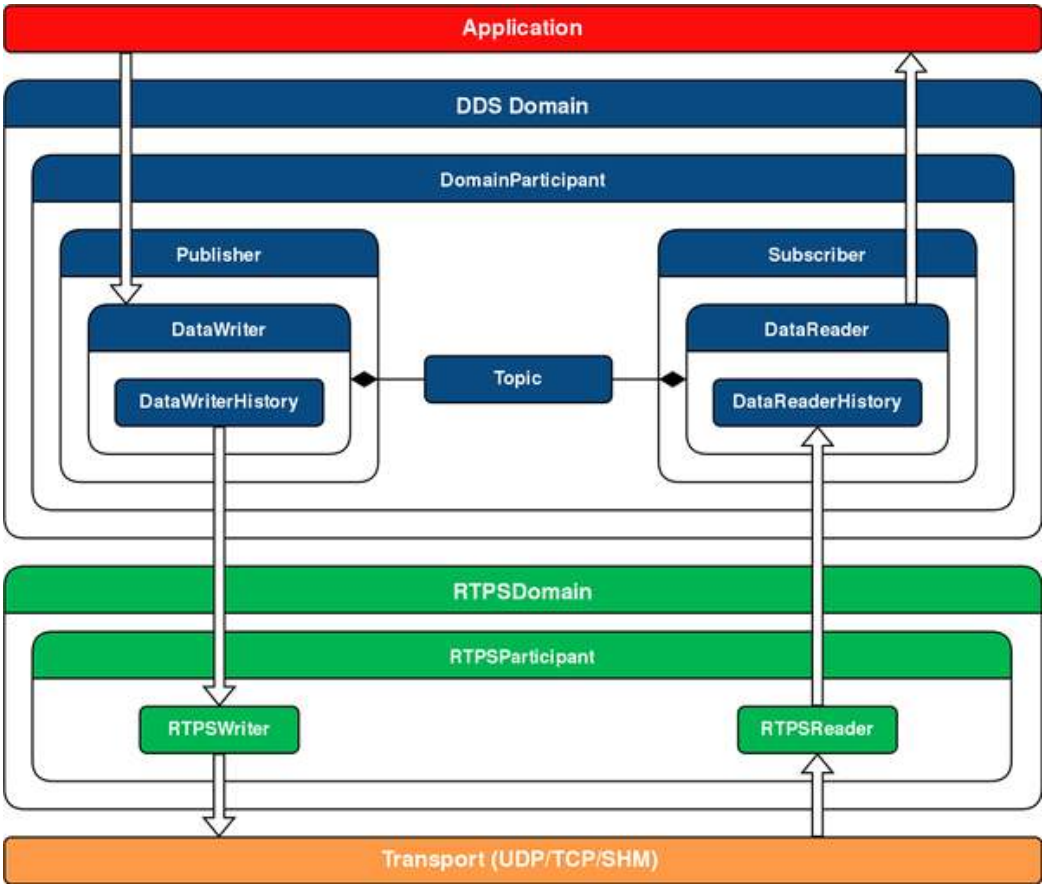
[그림 11] DDS Publish-Subscribe 구조)

DDS는 송신자 수, 수신자 수에 관계없이 높은 확장성을 확보할 수 있으며, 네트워크 구조 변화에 유연하게 대응할 수 있는 분산 구조를 제공한다. 특히 멀티캐스트 통신이 자동으로 최적화되어 대규모 다자간 통신에서도 성능 저하 없이 안정적인 데이터 전송이 가능하다.

이러한 특성 덕분에 DDS는 군사 무기체계, 자율주행 차량, 산업용 IoT, 로봇 통신, 항공기 미션 데이터 처리 등과 같이 실시간성과 신뢰성이 동시에 요구되는 시스템에서 널리 사용되고 있다. 특히 복잡한 네트워크 구조와 다수의 노드가 존재하는 환경에서도 높은 일관성과 실시간성을 유지할 수 있다는 점에서 차세대 통신 인프라로 주목받고 있다.

[표 1] 소켓 기반 통신과 DDS 기반 통신 비교

항목	소켓 기반	DDS 기반
프로토콜	TCP/UDP	RTPS 기반 Pub/Sub
수신자 확장성	제한적 (브로드캐스트 수작업 필요)	멀티캐스트 자동 관리
지연	수신자 수에 따라 선형 증가	수신자 수와 무관하게 일정
Qos	미지원	내장된 QoS 설정 (Reliability, Deadline, History 등)
메시지 구조	사용자 직접 설계 필요	IDL 기반 자동 생성



[그림 12] DDS(Data Distribution Service)의 구성 요소 및 통신 구조)

6. 실험 설계

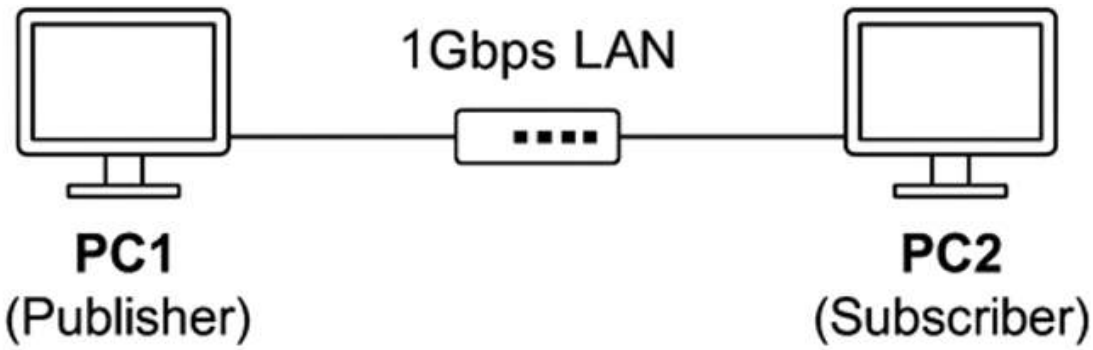
본 연구에서는 영상 인코딩 방식과 통신 구조에 따른 전송 성능을 비교 분석하기 위해 두 가지 유형의 실험 방식을 설계하였다.

첫 번째 방식은 영상 데이터를 인코딩한 후 디스크에 저장하고, 저장된 파일을 전송하는 구조로, 각 인코딩 방식 및 통신 방식에 대해 동일한 실험을 10회 반복하여 평균적인 성능을 측정하였다. 이 방식은 저장형 콘텐츠 전송 환경에서 소켓과 DDS의 전송 성능을 평가하는데 목적이 있다.

각 시스템의 하드웨어 사양은 다음과 같다. 송신자는 Intel Core i7-12700F CPU, 32GB RAM, Micron 2210S NVMe SSD(512GB) 및 WDC HDD(2TB)를 장착하였으며, NVIDIA GeForce GTX 1660 SUPER GPU(전용 VRAM 6GB)를 사용하였다. 수신자는 Intel Core i5-1135G7 CPU, 32GB RAM, SSSTC NVMe SSD(256GB) 및 HGST HDD(1TB)를 장착하였고, Intel Iris Xe Graphics 내장 GPU를 사용하였다. 두 시스템 모두 Windows 11 Pro 운영체제를 기반으로 실험이 진행되었다.

[그림 13] 소켓 기반 통신 실험 환경

DDS 소프트웨어 환경은 RTI Connext DDS 7.3.0을 기반으로 하였으며, 송신자와 수신자 코드는 Python 3.12.10으로 작성되었다. DDS QoS 설정은 QOS_PROFILES.xml 파일을 통해 제어하였으며, 본 연구에서는 History QoS 정책을 적용하여 Data Writer 측면에서 KEEP_LAST_HISTORY_QOS 모드를 선택하고, depth 값을 1로 설정하였다. 이는 송신자가 마지막 N개의 샘플만 유지하도록 하여 실시간 전송 시 불필요한 지연을 방지하고, 최신 데이터의 신속한 전달을 보장하기 위함이다.



[그림 14] DDS 기반 통신 실험 환경

두 번째 방식은 실시간 인코딩과 전송을 동시에 수행하는 구조로, 실제 스트리밍 환경과 유사한 조건을 구현하였다. 해당 실험은 각 조건에서 총 100회를 반복 수행함으로써 인코딩 지연, 네트워크 처리 시간, 그리고 전체 전송 시간의 평균값과 변동성을 평가하였다. 이를 통해 각 인코딩 방식과 통신 방식이 실시간 요구사항을 충족하는지에 대한 정량적 분석을 가능하게 하였다.

또한, 실험에서는 수신자 수(N)를 1에서 5, 10, 15, 20으로 점진적으로 증가시키며, 수신자 수 증가에 따른 통신 방식 별 구조의 확장성 및 성능 변화를 측정하였다. 수신자 수의 변화에 따라 전체 전송 시간이 얼마나 증가하는지, 혹은 안정성을 유지하는지를 비교함으로써 통신 구조의 확장성과 병목 현상 여부를 정밀하게 분석할 수 있

도록 설계하였다.
모든 실험에서는 다음의 네 가지 주요 항목을 측정 지표로 사용하였다:

첫째, 인코딩 시간은 원본 영상을 주어진 인코딩 방식으로 변환하는 데 소요된 시간으로, 코덱의 복잡도와 효율성을 반영한다.
둘째, 전송 시간은 인코딩된 데이터를 네트워크를 통해 송신하는 데 걸리는 시간으로, 통신 구조의 효율성과 네트워크 혼잡도에 영향을 받는다.
셋째, 전체 시간은 인코딩 시간과 전송 시간을 합산한 값으로, 사용자가 영상을 수신하여 저장하기까지의 지연을 의미한다.
넷째, 표준편차는 각 반복 실험 간 성능의 일관성을 나타내는 지표로, 시스템의 안정성과 예측 가능성을 평가하는 데 활용된다.

이와 같은 실험 설계를 통해 본 연구는 다양한 인코딩 방식과 통신 방식이 영상 전송 성능에 미치는 영향을 체계적으로 분석하고, 실시간 환경에서 최적의 조합을 도출하는 데 목적을 두고 있다.

7. 실험 결과

[표 2]는 소켓 기반 통신에서 인코딩 후 파일을 저장하고 이를 전송했을 때의 평균 소요 시간을 나타낸다. H.264는 전체 39.15초로 가장 효율적인 성능을 보였으며, JPEG와 AV1은 각각 90.24초, 270.81초로 시간이 크게 증가하였다. 인코딩과 전송 시간의 비율을 보면, H.264와 H.264_intra는 전송 시간이 지배적이었으나, AV1은 인코딩에 90% 이상이 소요되어 실시간 전송에는 부적합한 결과를 나타냈다. 이는 코덱 복잡성이 전송 성능에 큰 영향을 미침을 시사한다.

[표 2] 소켓 기반 1대 1 통신에서 인코딩 방식별 전송 성능 (10회 평균)

	평균 인코딩 시간	**평균 전송 시간**	**평균 전체 시간**
H.264	14.19초	24.96초	39.15초
H.264_intra	15.75초	38.02초	53.77초
H.265	27.73초	24.85초	52.58초
JPEG	58.31초	31.93초	90.24초
AV1	242.85초	27.96초	270.81초

[표 3] 전체 전송 시간에서 인코딩과 전송의 비중 비교)

인코딩 방식	인코딩 비율	전송 비율
H.264	36.2%	63.8%
H.264_intra	29.3%	70.7%
H.265	52.8%	47.2%
JPEG	64.6%	35.4%
AV1	89.7%	10.3%

[표 4]는 동일한 조건에서 DDS를 활용했을 때의 성능을 제시한다. 전체 전송 시간은 소켓 대비 다소 길었으나, H.265의 경우 50.20초로 소켓 대비 오히려 단축되는 결과를 보였다. 이는 DDS가 제공하는 효율적인 데이터 관리 메커니즘이 특정 코덱과 결합될 때 성능 개선으로 이어질 수 있음을 의미한다. 특히 AV1은 소켓과 마찬가지로 인코딩 시간이 지배적이었으며, DDS 전송에서는 전송 지연이 상대적으로 줄어드는 경향을 보였다.

[표 4] DDS 기반 1대 1 통신에서 인코딩 방식별 전송 성능 (10회 평균)

	평균 인코딩 시간	평균 전송 시간	평균 전체 시간
H.264	14.19초	27.17초	41.36초
H.264_intra	15.75초	33.34초	49.09초
H.265	27.73초	22.47초	50.20초
JPEG	58.31초	30.51초	88.82초
AV1	242.85초	24.32초	267.17초

[표 5] 전체 전송 시간에서 인코딩과 전송의 비중 비교

인코딩 방식	인코딩 비율	전송 비율
H.264	34.3%	65.7%
H.264_intra	32.1%	67.9%
H.265	55.2%	44.8%
JPEG	65.6%	34.4%
AV1	90.9%	9.1%

(3) 소켓 vs DDS 통신 비교 정리

[표 6]은 저장형 전송 환경에서 소켓과 DDS의 성능을 코덱별로 비교한 결과를 나타낸다. H.264와 H.264_intra에서는 DDS가 소켓에 비해 전송 시간이 길게 나타나면서 전체 시간 또한 증가하였다. 반면 H.265에서는 DDS가 소켓보다 전송 시간이 짧게 측정되어 전체 시간에서도 소폭의 개선이 확인되었다. JPEG와 AV1의 경우, 인코딩

시간은 소켓에서 더 길었으나 전송 시간은 DDS에서 다소 효율적으로 나타났으며, 결과적으로 전체 시간은 양측 간 큰 차이가 발생하지 않았다. 요약하면, 대부분의 코덱에서는 DDS가 소켓보다 전송에서 불리한 성능을 보였으나, H.265에서는 오히려 DDS가 우수한 결과를 보였고, JPEG와 AV1에서도 제한적이지만 전송 효율성 측면에서 DDS의 강점이 확인되었다.

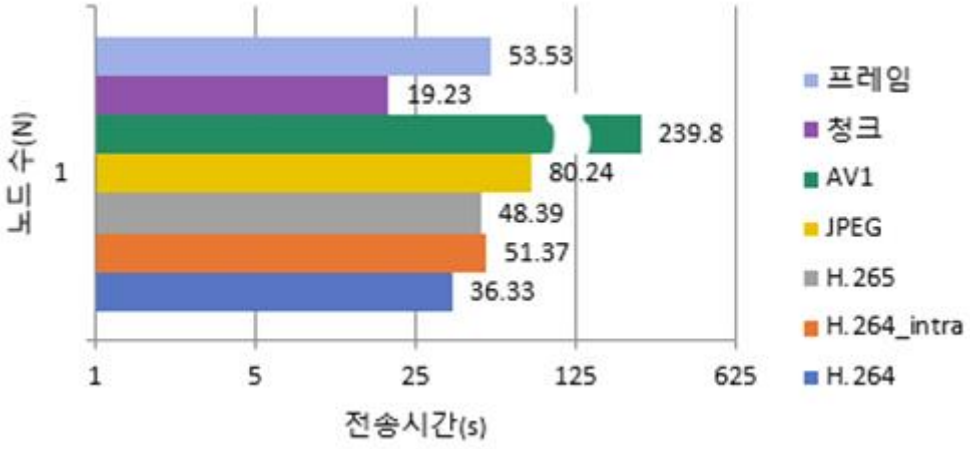
[표 6] 저장형 전송(인코딩 후 디스크)에서 인코딩 방식별 성능 (1대 1, 10회 평균)

	소켓	DDS	소켓	DDS	소켓	DDS
	평균 인코딩 시간		전송 시간 평균		전체 시간	
H.264	14.19초		24.96초	27.17초	39.15초	41.36초
H.264_intra	15.75초		38.02초	33.34초	53.77초	49.09초
H.265	27.73초		24.85초	22.47초	52.58초	50.20초
JPEG	58.31초		31.93초	30.51초	90.24초	88.82초
AV1	242.85초		27.96초	24.32초	270.81초	267.17초

[표 7]은 인코딩과 전송을 동시에 진행했을 때의 소켓 기반 결과를 나타낸다. H.264는 36.33초로 가장 빠른 실시간 전송 성능을 보였으며, AV1은 239.8초로 가장 느렸다. JPEG는 80.24초로 비교적 높은 지연을 기록하였다. 흥미롭게도 청크 단위 전송은 19.23초로 가장 짧은 지연을 나타내어, 대규모 데이터 전송에서 청크 방식이 유리함을 확인할 수 있었다.

[표 7] 소켓 기반 실시간 전송 성능 (1대 1, 100회 평균)

	전체 시간	표준편차
H.264	36.33초	0.45초
H.264_intra	51.37초	0.64초
H.265	48.39초	0.6초
JPEG	80.24초	0.99초
AV1	239.8초	2.97초
청크	19.23초	0.24초
프레임	53.53초	0.87초



[그림 15] 소켓 기반 1대 1 전송에서 인코딩 방식별 전송 시간 비교

[표 8]은 동일 조건에서 DDS 기반 결과이다. 모든 코덱에서 소켓 대비 전송 시간이 단축되었으며, 특히 H.264는 28.21초로 가장 우수한 결과를 보였다. 청크 단위 전송은 23.72초로 소켓 대비 느렸으나, 프레임 단위 전송은 41.17초로 효율적이었다. 이는 DDS가 실시간 스트리밍 환경에서 QoS를 기반으로 안정적인 성능을 제공함을 보여준다.

[표 8] DDS 기반 실시간 전송 성능 (1대 1, 100회 평균)

	전체 시간	표준편차
H.264	28.21초	0.97초
H.264_intra	31.26초	1.21 초
H.265	35.56초	1.60초
JPEG	66.71초	2.85 초
AV1	252.17초	3.12초
청크	23.72 초	0.06초
프레임	41.17초	2.01 초

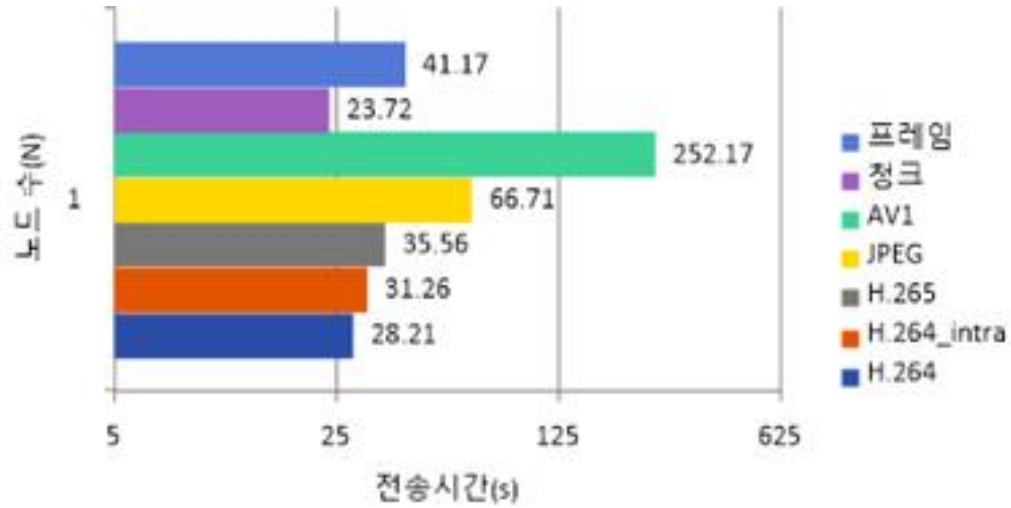
(6) 소켓 vs DDS 통신 비교 정리

[표 9]는 실시간 전송에서 두 통신 방식의 차이를 요약한다. 대부분의 코덱에서 DDS가 소켓보다 빠른 결과를 나타냈으며, 특히 H.264, H.264_intra, H.265 모두 약 20~30%의 시간 단축 효과를 보였다. 반면 AV1은 소켓 대비 오히려 성능이 저하되었는

데, 이는 코덱 자체의 인코딩 복잡성이 통신 방식의 장점을 상쇄했기 때문으로 해석된다.

[표 9] 실시간 전송 성능 (1대 1, 100회 평균)

	소켓	DDS
	전체 시간	
H.264	36.33초	28.21초
H.264_intra	51.37초	31.26초
H.265	48.39초	35.56초
JPEG	80.24초	66.71초
AV1	239.8초	252.17초
청크	19.23초	23.72 초
프레임	53.53초	41.17초



[그림 16] DDS 기반 1대 1 전송에서 인코딩 방식별 전송 시간 비교

[표 10]은 수신자 수를 늘렸을 때 소켓 기반 전송 성능을 보여준다. H.264는 N=1에서 36.33초였으나, N=20에서 389.33초로 약 10배 이상 증가하였다. AV1은 수신자 증가에 따라 239.8초에서 3477.1초까지 폭발적으로 증가하여 멀티캐스트 환경에 부적합함을 나타냈다. 반면 청크 단위 전송은 수신자 증가에도 비교적 선형적으로 증가하여 확장성 측면에서 상대적으로 안정적인 결과를 보였다.

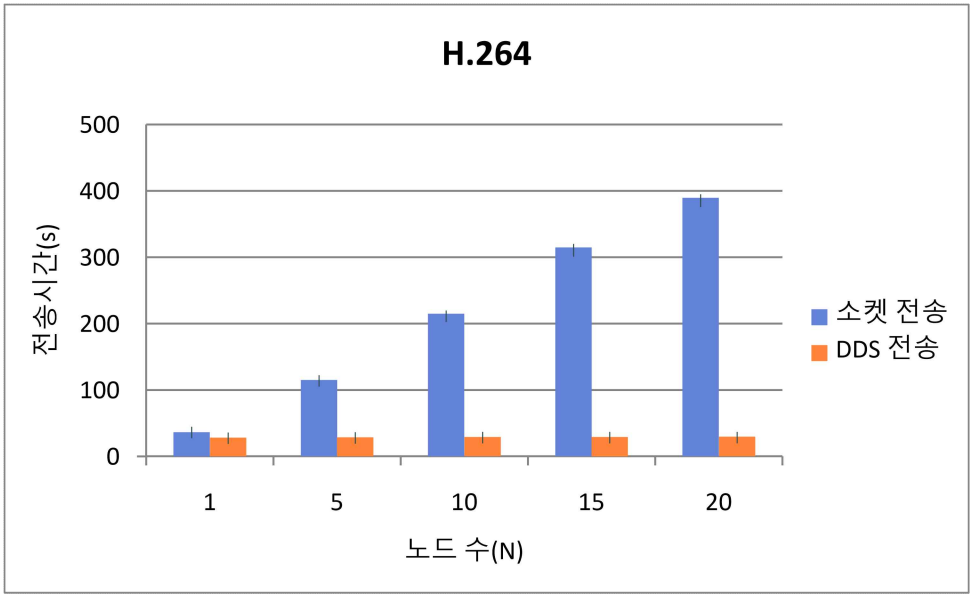
[표 10] 소켓 기반 실시간 전송 성능 (1대 N, 100회 평균)

	N = 1	표준편차	N = 5	표준편차	N = 10	표준편차	N = 15	표준편차	N = 20	표준편차
H.264	36.33초	0.45초	114.77초	1.25초	214.61초	3.56초	314.45초	5.32초	389.33초	5.32초
H.264_intra	51.37초	0.64초	169.24초	2.08초	321.32초	5.13초	473.4초	6.54초	587.46초	6.54초
H.265	48.39초	0.6초	128.42초	1.53초	227.82초	3.54초	327.22초	5.51초	401.77초	5.51초
JPEG	80.24초	0.99초	187.51초	2.58초	315.23초	5.64초	442.95초	6.59초	538.74초	6.59초
AV1	239.8초	2.97초	1031.14초	6.8초	1870.44초	10.5초	2685.76초	14.1초	3477.1초	17.6초
청크	19.23초	0.24초	77.16초	0.72초	154.08초	2.13초	231초	3.54초	288.69초	3.54초
프레임	53.53초	0.87초	215초	2.65초	429.11초	6.21초	643.23초	8.75초	803.82초	8.75초

[표 11]은 동일 조건에서 DDS 기반 성능을 보여준다. 모든 코덱에서 수신자 수 증가에도 전체 시간이 거의 일정하게 유지되었으며, H.264는 N=1에서 28.21초, N=20에서도 29.48초로 큰 차이가 없었다. 이는 DDS가 멀티캐스트 환경에서 뛰어난 확장성을 제공함을 직접적으로 입증한다. 특히 소켓 방식에서 병목 현상이 심했던 AV1도 DDS에서는 완만한 증가만을 보여, 다수 수신자 환경에서 DDS의 우수성이 명확히 드러났다.

[표 11] DDS 기반 실시간 전송 성능 (1대 N, 100회 평균)

	N = 1	표준편차	N = 5	표준편차	N = 10	표준편차	N = 15	표준편차	N = 20	표준편차
H.264	28.21초	0.97초	28.63초	1.03초	28.92초	1.1초	29.2초	1.17초	29.48초	1.25초
H.264_intra	31.26초	1.21초	31.73초	1.28초	32.04초	1.37초	32.35초	1.46초	32.67초	1.55초
H.265	35.56초	1.6초	36.09초	1.7초	36.45초	1.82초	36.8초	1.94초	37.16초	2.06초
JPEG	66.71초	2.85초	67.71초	3.02초	68.38초	3.23초	69.04초	3.45초	69.71초	3.66초
AV1	252.17초	3.12초	255.95초	3.31초	258.47초	3.54초	261초	3.78초	263.52초	4.01초
청크	23.72초	0.06초	24.08초	0.06초	24.31초	0.07초	24.55초	0.07초	24.79초	0.08초
프레임	41.17초	2.01초	41.79초	2.13초	42.2초	2.28초	42.61초	2.43초	43.02초	2.58초

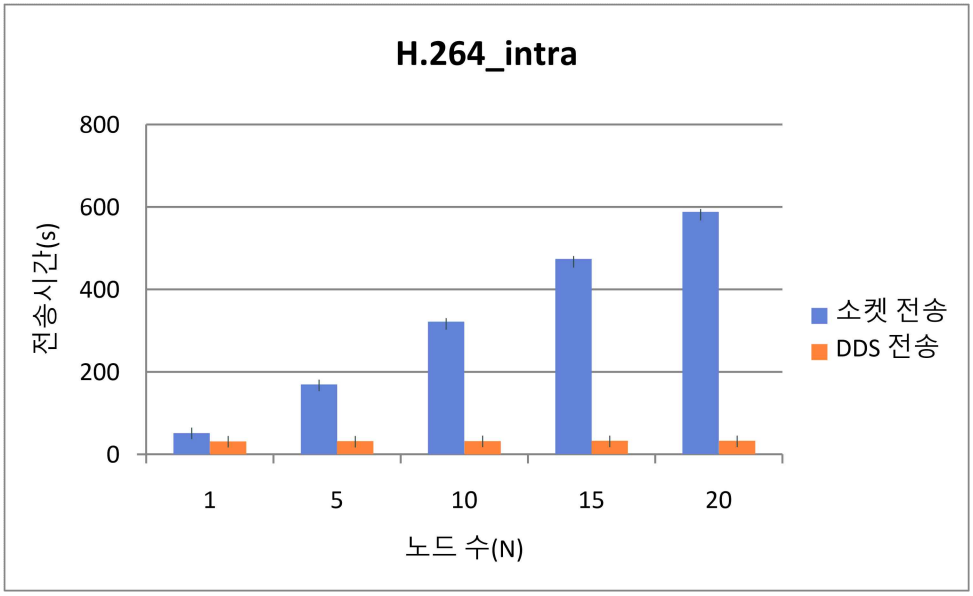


[그림 17] H.264 코덱 기반 소켓 통신과 DDS 통신의 전송 시간 비교 (1대 N 전송)

[그림 17]은 H.264 인코딩 방식에서 수신자 수(N)를 1, 5, 10, 15, 20으로 점차 증가시켰을 때의 평균 전송 시간을 보여준다. 소켓 기반 통신(파란색)은 수신자 수가 증가함에 따라 전송 시간이 급격히 증가하였으며, N=1에서 약 36초 수준이던 전송 시간이 N=20에서는 약 389초까지 증가하였다. 이는 소켓 통신이 다중 수신자 환경에서 부하가 선형적으로 누적되는 구조적 한계를 잘 보여준다.

반면, DDS 기반 통신(주황색)은 수신자 수 증가에도 불구하고 전송 시간이 거의 일정하게 유지되었다. N=1에서 28초였던 평균 전송 시간은 N=20에서도 약 29초로 큰 변화가 없었다. 이는 DDS가 QoS 기반 멀티캐스트를 지원하여 수신자 수와 무관하게 안정적인 전송 성능을 보장함을 의미한다.

따라서 H.264 인코딩 환경에서 다수 수신자 실시간 전송을 고려할 경우, 소켓 기반 방식은 확장성 측면에서 한계를 가지는 반면, DDS는 수신자 수에 관계없이 일정한 성능을 유지할 수 있는 우수한 대안임을 확인할 수 있다.

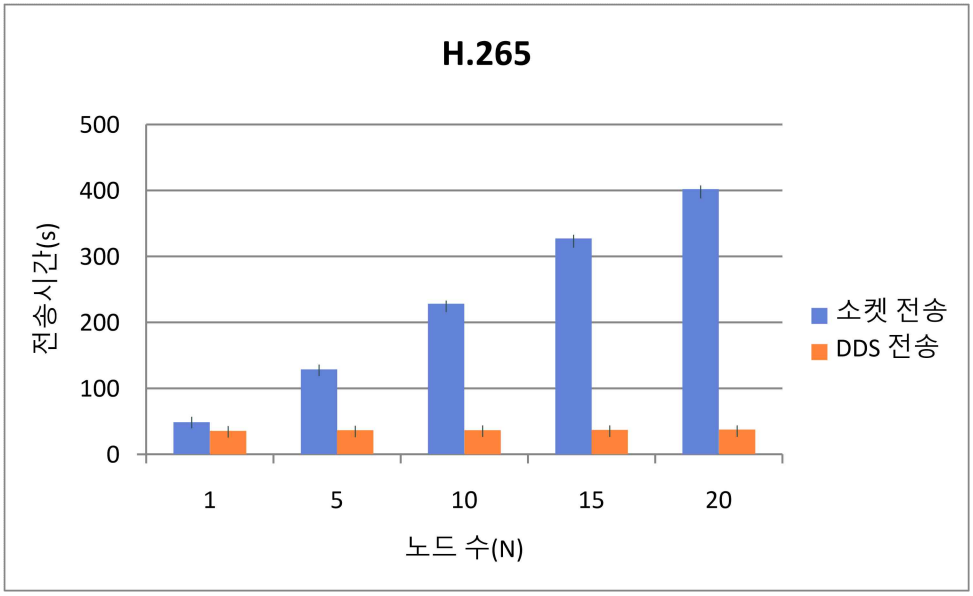


[그림 18] H.264_intra 코덱 기반 소켓 통신과 DDS 통신의 전송 시간 비교 (1대 N 전송)

[그림 18]은 H.264_intra 인코딩 환경에서 수신자 수(N)가 증가함에 따라 소켓 통신과 DDS 통신의 전송 시간 변화를 비교한 결과를 제시한다. 소켓 기반 통신(파란색)은 수신자 수가 늘어날수록 전송 시간이 기하급수적으로 증가하였다. N=1에서는 약 51초였으나, N=10에서는 321초, N=20에서는 약 587초까지 증가하였다. 이러한 결과는 소켓 기반 구조가 멀티 수신자 환경에서 확장성 한계를 가지며, 수신자가 늘어날수록 서버 부하가 직접적으로 누적됨을 잘 보여준다.

반면, DDS 기반 통신(주황색)은 수신자 수 증가에도 불구하고 전송 시간이 거의 일정하게 유지되었다. N=1에서 31초였던 평균 전송 시간은 N=20에서도 32초 수준에 머물러 큰 변화가 없었다. 이는 DDS가 멀티캐스트 전송을 네이티브로 지원하고, QoS 기반의 데이터 관리 정책을 통해 네트워크 부하를 효과적으로 분산하기 때문으로 해석된다.

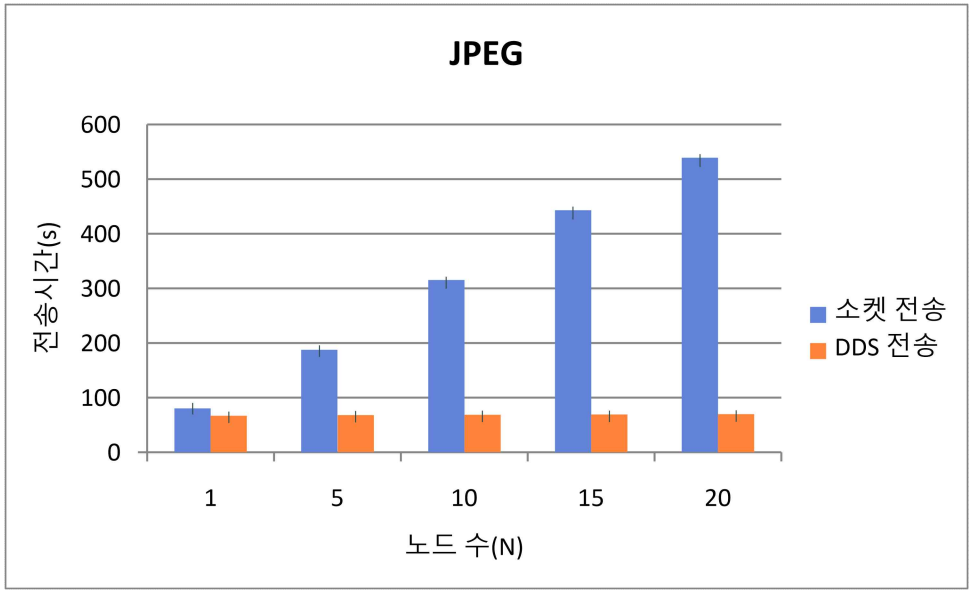
따라서 H.264_intra 인코딩 방식에서도 소켓과 DDS 간의 성능 격차는 명확히 드러나며, 특히 다수 수신자 환경에서는 DDS가 훨씬 안정적이고 확장 가능한 전송 구조임을 입증한다.



[그림 19] H.265 코덱 기반 소켓 통신과 DDS 통신의 전송 시간 비교 (1대 N 전송)

[그림 19]는 H.265 인코딩 방식에서 수신자 수(N)가 증가할 때 소켓 통신과 DDS 통신의 평균 전송 시간을 비교한 결과를 나타낸다. 소켓 기반 통신(파란색)의 경우, 수신자 수가 늘어날수록 전송 시간이 급격히 증가하였다. N=1에서 약 48초였던 전송 시간은 N=10에서 227초, N=20에서는 약 402초까지 증가하였다. 이러한 결과는 소켓 통신이 멀티캐스트 환경에서 확장성 한계를 보이며, 수신자 수 증가에 따라 서버 처리 부하가 크게 누적됨을 보여준다.

반면 DDS 기반 통신(주황색)은 수신자 수가 증가해도 전송 시간이 거의 일정하게 유지되었다. N=1에서 36초였던 전송 시간은 N=20에서도 약 37초로 큰 차이가 없었다. 이는 DDS가 멀티캐스트 기능을 네이티브로 지원하며, QoS 기반 전송 관리 정책을 통해 부하를 효과적으로 분산하기 때문으로 해석된다.
따라서 H.265 인코딩 환경에서도 소켓과 DDS 간의 성능 격차는 뚜렷하게 확인되며, 특히 다수 수신자가 존재하는 환경에서는 DDS가 안정적이고 확장 가능한 전송 구조임을 다시 한번 입증한다.

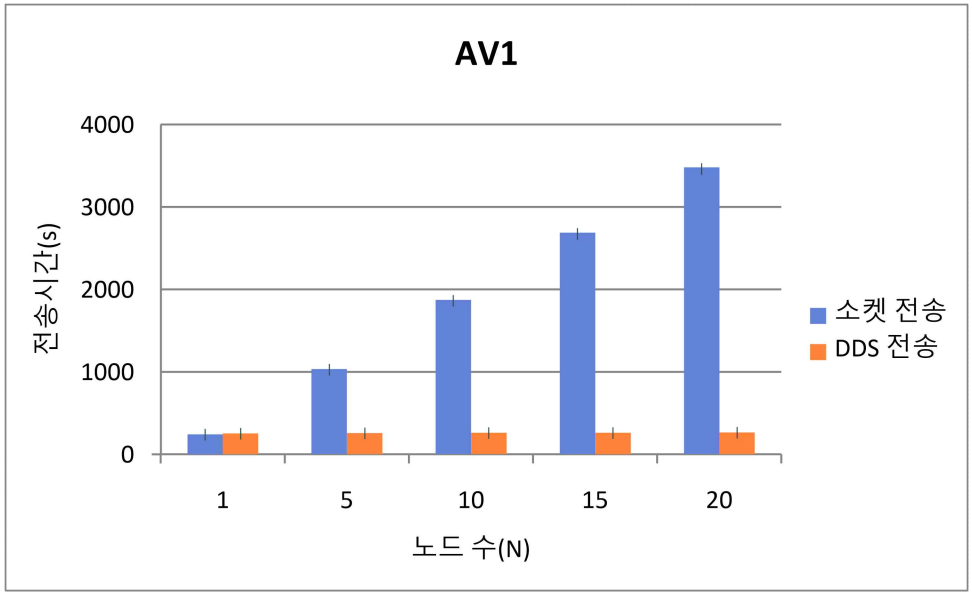


[그림 20] JPEG 코덱 기반 소켓 통신과 DDS 통신의 전송 시간 비교 (1대 N 전송)

[그림 20]은 JPEG 인코딩 방식을 적용했을 때, 수신자 수(N)의 증가에 따른 소켓 통신과 DDS 통신의 전송 시간 변화를 나타낸다. 소켓 기반 통신(파란색)의 경우, 수신자 수가 증가할수록 전송 시간이 기하급수적으로 증가하였다. N=1에서 약 80초였던 전송 시간은 N=10에서 315초, N=20에서는 538초를 넘어섰다. 이는 JPEG 코덱이 프레임 단위 처리 구조로 인해 데이터 크기가 크고, 소켓 통신이 멀티 수신자 환경에서 부하를 효율적으로 처리하지 못한다는 한계를 보여준다.

반대로 DDS 기반 통신(주황색)은 수신자 수 증가에도 불구하고 전송 시간이 안정적으로 유지되었다. N=1에서 약 67초였던 전송 시간은 N=20에서도 약 70초로 소폭 증가하는 수준에 그쳤다. 이는 DDS가 멀티캐스트와 QoS 기반 전송 관리 기능을 통해 데이터 분배를 효율적으로 수행함을 보여준다.

결과적으로 JPEG 인코딩 환경에서도 소켓 방식은 수신자 수 증가에 따라 급격한 성능 저하를 보인 반면, DDS는 높은 확장성과 안정성을 유지하며 대규모 다자간 전송에 적합한 통신 구조임이 확인되었다.

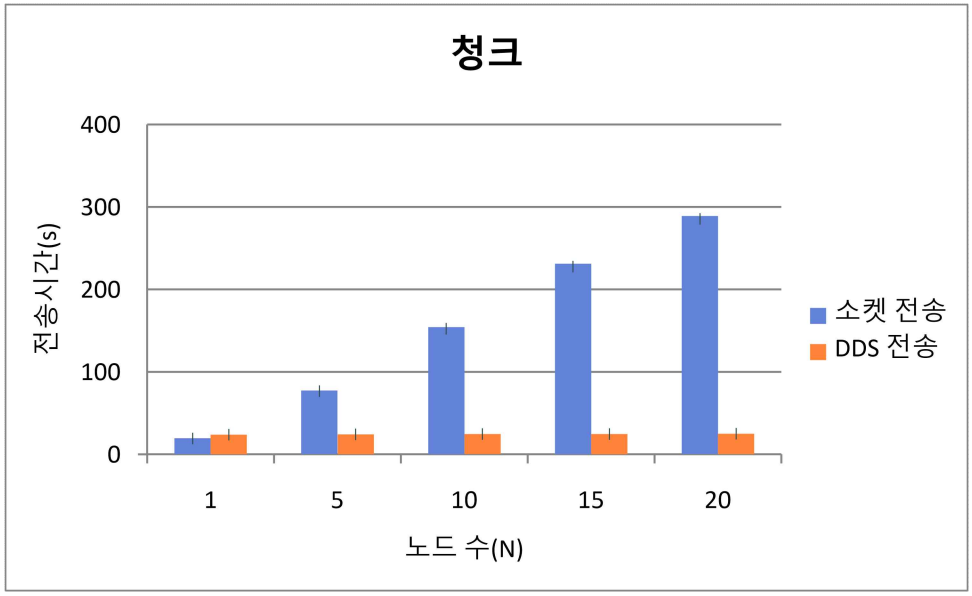


[그림 21] AV1 코덱 기반 소켓 통신과 DDS 통신의 전송 시간 비교 (1대 N 전송)

[그림 21]은 AV1 인코딩 방식을 적용했을 때, 수신자 수(N)의 증가에 따른 소켓 통신과 DDS 통신의 전송 시간 변화를 비교한 것이다. 소켓 기반 통신(파란색)은 수신자 수가 늘어날수록 전송 시간이 폭발적으로 증가하였다. N=1에서 약 240초였던 전송 시간은 N=10에서 약 1,870초, N=20에서는 무려 3,477초까지 치솟았다. 이는 AV1의 높은 인코딩 복잡도와 소켓 구조의 한계가 결합되어 멀티 수신자 환경에서 성능 저하가 극심하게 나타남을 보여준다.

반면, DDS 기반 통신(주황색)은 수신자 수 증가에도 불구하고 전송 시간이 안정적으로 유지되었다. N=1에서 약 252초였던 전송 시간은 N=20에서도 약 263초 수준에 머물러 큰 차이를 보이지 않았다. 이러한 결과는 DDS가 네이티브 멀티캐스트와 QoS 관리 기능을 통해 네트워크 부하를 효과적으로 분산할 수 있음을 다시 한번 입증한다.

따라서 AV1 인코딩 방식은 소켓 통신 환경에서는 실시간 다자간 전송에 극도로 부적합하지만, DDS 환경에서는 수신자 수 증가에도 안정적인 성능을 유지할 수 있음을 확인할 수 있다. 이는 DDS의 확장성이 AV1과 같은 고복잡도 코덱 환경에서도 여전히 유효함을 의미한다.

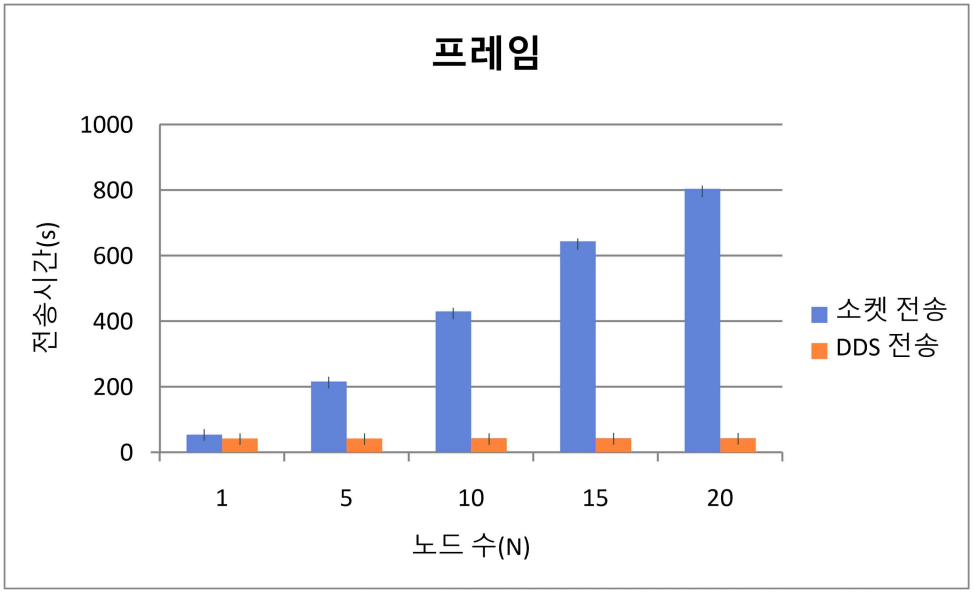


[그림 22] 청크 단위 기반 소켓 통신과 DDS 통신의 전송 시간 비교 (1대 N 전송)

[그림 22]는 청크 단위 전송 방식을 적용했을 때, 수신자 수(N)가 증가함에 따른 소켓 통신과 DDS 통신의 전송 시간 변화를 비교한 것이다. 소켓 기반 통신(파란색)은 수신자 수 증가에 따라 전송 시간이 선형적으로 증가하였다. N=1에서 약 19초였던 전송 시간은 N=10에서 154초, N=20에서는 약 289초까지 증가하였다. 이러한 결과는 소켓 방식이 수신자 수 증가에 비례하여 부하가 누적됨을 보여주지만, 다른 인코딩 방식(H.264, AV1 등)에 비해 증가 폭이 상대적으로 완만하다는 특징을 가진다.

반면 DDS 기반 통신(주황색)은 수신자 수가 증가해도 전송 시간이 거의 일정하게 유지되었다. N=1에서 약 23.7초였던 평균 전송 시간은 N=20에서도 약 24.8초로, 변화 폭이 미미했다. 이는 DDS가 멀티캐스트와 QoS 정책을 기반으로 데이터 흐름을 제어하여, 수신자 수 증가에 따른 성능 저하를 효과적으로 방지함을 보여준다.

따라서 청크 단위 전송 방식에서도 DDS는 소켓 대비 높은 확장성과 안정성을 제공하며, 특히 다자간 전송 환경에서 일관된 성능을 유지할 수 있는 강점을 갖고 있음을 확인할 수 있다.



[그림 23] 프레임 단위 기반 소켓 통신과 DDS 통신의 전송 시간 비교 (1대 N 전송)

[그림 23]은 프레임 단위 전송 방식을 적용했을 때, 수신자 수(N)가 증가함에 따른 소켓 통신과 DDS 통신의 전송 시간 변화를 비교한 것이다. 소켓 기반 통신(파란색)의 경우, 수신자 수가 늘어남에 따라 전송 시간이 급격히 증가하였다. N=1에서 약 54초였던 전송 시간은 N=10에서 429초, N=20에서는 약 804초까지 증가하였다. 이러한 결과는 프레임 단위 전송이 데이터 크기 부담이 크고, 소켓 구조가 멀티 수신자 환경에서 부하를 효과적으로 처리하지 못함을 보여준다.

반면 DDS 기반 통신(주황색)은 수신자 수 증가에도 불구하고 전송 시간이 안정적으로 유지되었다. N=1에서 약 41초였던 평균 전송 시간은 N=20에서도 약 43초로, 전체적으로 일정한 범위를 유지하였다. 이는 DDS의 QoS 기반 멀티캐스트 구조가 프레임 단위와 같이 데이터 크기가 큰 전송 환경에서도 안정적으로 작동함을 보여준다.

따라서 프레임 단위 전송 방식에서도 소켓 통신은 수신자 수 증가에 따라 급격한 성능 저하를 보이는 반면, DDS 통신은 일정한 성능을 유지하며 다자간 전송 환경에서 우수한 확장성을 제공함을 확인할 수 있다.