

1. 인성 질문 10개 및 답변

1) 본인의 강점과 약점은 무엇인가요?

꼬리질문: 약점을 보완하기 위해 어떤 노력을 하고 있나요?

답변:

저의 강점은 문제 해결 능력과 논리적인 사고입니다. 어려운 문제를 분석하고 최적의 해결책을 도출하는 데 강합니다. 예를 들어, 3D 렌더링 모델을 최적화할 때 다양한 논문을 분석하고 성능을 유지하면서도 경량화하는 방법을 찾아 적용했습니다.

약점은 지나치게 세부적인 부분에 신경 쓰다 보니 속도가 늦어질 때가 있다는 점입니다. 이를 보완하기 위해 우선순위를 정하고 중요한 작업부터 처리하는 습관을 기르고 있습니다.

2) 갈등 상황을 겪었을 때 어떻게 해결하나요?

꼬리질문: 협업 과정에서 상대방과 의견이 맞지 않는다면 어떻게 조율하나요?

답변:

졸업 프로젝트에서 팀원 간 코드 병합 규칙이 달라 병합 과정에서 문제가 발생했습니다. 팀원들과 집중 회의를 통해 문제를 분석하고 Git 관리 프로토콜을 정립하여 해결했습니다.

협업 중 의견이 다를 때는 상대방의 논리를 이해한 후 객관적인 데이터를 기반으로 해결책을 제시하는 방식으로 조율합니다.

3) 스트레스를 받았을 때 어떻게 극복하나요?

꼬리질문: 실습 중 높은 압박이 주어진다면 어떻게 대처할 계획인가요?

답변:

저는 스트레스를 해결할 때 문제의 원인을 분석하고, 해결 가능한 부분과 그렇지 않은 부분을 구분하여 우선적으로 해결할 부분에 집중합니다. 또한, 짧은 산책이나 운동을 통해 리프레시하는 것도 효과적인 방법으로 활용하고 있습니다.

실습 중 압박이 클 경우, 중요한 작업을 우선 정리하고 협업을 통해 해결 가능한 부분을 적극적으로 논의하며 효율적인 해결책을 찾겠습니다.

4) 도전했던 경험을 말해주세요.

꼬리질문: 실패한 경험이 있다면 무엇을 배웠나요?

답변:

3D 렌더링 모델을 최적화할 때 기존 성능을 유지하면서도 경량화하는 것이 도전이었습니다. 논문을 분석하고 여러 실험을 거쳐 56% 경량화와 25% 추론 시간 단축에 성공했습니다.

초기에는 모델의 성능 저하가 발생하는 실패를 겪었지만, 이를 해결하기 위해 다양한 접근법을 시도하며 최적의 방법을 찾았습니다.

5) 본인의 장기적인 목표는 무엇인가요?

꼬리질문: 해당 실습이 목표 달성을 어떻게 기여할 수 있을까요?

답변:

장기적으로는 CV 기반 장치 SW 개발자로 성장하여 실시간 렌더링과 센서 데이터 처리 기술을 활용하는 전문가가 되고 싶습니다.

이번 실습에서는 센서 데이터 처리 및 SW 최적화 경험을 쌓으며, 실무에서 요구되는 기술을 익히는 것이 목표입니다.

2. 자소서 기반 질문 20개 및 답변

1) CV 기반 센서 제어 프로젝트에서 가장 중요한 기술적 요소는 무엇이었나요?

답변:

CV 기반 센서 제어에서 가장 중요한 요소는 실시간 데이터 처리와 HW-SW 간의 유기적인 인터페이스입니다. 센서에서 수집한 데이터를 빠르게 분석하고, 이에 맞춰 적절한 제어 신호를 HW에 전달해야 하기 때문에 인터럽트 기반 처리 방식과 최적화된 알고리즘이 필수적이었습니다.

2) 프로젝트를 수행하며 예상치 못한 문제가 발생했을 때 어떻게 대응했나요?

답변:

센서에서 들어오는 데이터의 노이즈가 커서 인식률이 저하되는 문제가 발생했습니다.

이를 해결하기 위해 Kalman Filter와 Gaussian Smoothing을 적용하여 노이즈를 제거하고, 데이터를 정제한 후 분석하여 성능을 개선했습니다.

꼬리질문:

노이즈 제거를 위해 Kalman Filter와 Gaussian Smoothing을 적용했다고 했는데, 두 기법의 차이점은 무엇인가요?

답변:

Kalman Filter는 현재 상태를 예측하고 새로운 데이터를 반영하여 최적의 상태를 추정하는 알고리즘입니다.

Gaussian Smoothing은 단순한 평균 필터링 방식으로, 주변 데이터를 활용해 부드러운 값을 생성합니다.

Kalman Filter는 동적 환경에서 변화에 적응할 수 있는 반면, Gaussian Smoothing은 정적인 노이즈 제거에 더 적합합니다.

3. 자소서를 제외한 직무 관련 질문 20개 및 답변

1) 3D 스캐너에서 가장 중요한 기술적 요소는 무엇인가요?

답변:

정확한 데이터 획득을 위한 **광학 센서 보정, 3D Reconstruction** 알고리즘, 실시간 데이터 처리 등이 중요합니다. 특히, 실시간 렌더링 성능을 확보하는 것이 핵심 과제입니다.

꼬리질문:

실시간 렌더링을 최적화하기 위해 어떤 방법을 사용할 수 있나요?

답변:

LOD(Level of Detail) 기법을 적용하여 렌더링 부하를 줄이고, 중요도가 낮은 영역의 폴리곤 수를 줄이는 방식을

활용할 수 있습니다. 또한, GPU 병렬 연산을 활용해 처리 속도를 높일 수도 있습니다.

2) 광학 센서 데이터를 처리할 때 가장 어려운 점은 무엇인가요?

답변:

광학 센서는 조명 환경, 반사율, 물체 표면의 특성 등에 따라 노이즈가 발생할 수 있습니다. 이를 보정하기 위해 **다중 프레임 합성 기법과 딥러닝 기반 노이즈 제거 모델**을 적용할 수 있습니다.

고리질문:

딥러닝을 활용한 노이즈 제거 모델은 어떻게 구현할 수 있을까요?

답변:

CNN 기반의 Denoising Autoencoder를 활용하여 센서 데이터의 패턴을 학습하고, 특정 노이즈를 제거하는 방식으로 구현할 수 있습니다.

이 외에도 추가적인 질문과 답변이 필요하시면 말씀해주세요!

구강 스캐너 SW 개발 직무 면접 대비 정리

1. 구강 스캐너 SW 개발 직무에서 중요한 기술

구강 스캐너 소프트웨어에서는 **딥러닝 기반 슈퍼 레졸루션, 3D 재구성 기술, 실시간 데이터 최적화**가 중요합니다.

1.1 딥러닝 기반 슈퍼 레졸루션

- 저해상도 스캔 데이터를 고해상도로 변환하여 치아 및 구강 내부의 작은 구조까지 선명하게 포착할 수 있도록 함.
- 센서 해상도 한계를 극복할 수 있어, 진단과 치료 계획에 도움을 줄 수 있음.

1.2 3D 재구성 기술

- 다각도에서 취득한 포인트 클라우드를 정합(Registration)하여 **일관된 3D 모델**을 생성하는 과정이 필요.
- **ICP(Iterative Closest Point), NERF(Neural Radiance Fields), Gaussian Splatting** 등의 최신 AI 기반 기법을 활용 가능.

1.3 실시간 데이터 최적화

- 구강 스캐너는 **의료 환경**에서 실시간으로 데이터를 제공해야 하므로, **지연(latency)**을 최소화해야 함.
- **Pruning**과 **Quantization** 같은 경량화 기법이 필수적.
- **EAGLES** 논문의 **pruning** 방식을 적용하여, PSNR과 SSIM을 99% 유지하면서 **모델 크기를 56%** 감소, 실시간 응답 속도를 **20%** 향상시킨 경험 있음.

2. 3D 데이터 처리 시 고려할 점

2.1 정확도

- 치아 상태를 명확하게 확인하기 위해, 3D 데이터의 정확성이 중요.
- 이를 위해 노이즈 제거 및 필터링 기술이 필요.

2.2 연산 최적화

- 실시간 렌더링이 가능하도록 연산 효율을 높이고 **GPU 가속을 활용해야 함.**
- 딥러닝 모델 최적화 및 양자화(**Quantization**) 기법 적용을 고려.

2.3 파일 크기 감소

- **양자화(Quantization)** 또는 딥러닝 모델 최적화 과정을 거쳐 데이터 크기를 줄여야 함.

3. 실시간 데이터 렌더링 최적화 방법

3.1 AI 기법 활용

- **NERF(Neural Radiance Fields), Gaussian Splatting** 등을 적용하여 실시간 3D 렌더링 성능을 최적화.

3.2 Pruning 및 Quantization

- **Pruning(가중치 제거):** 중요하지 않은 데이터나 가우시안 포인트 제거.
- **Quantization(양자화):** FP16, INT8 같은 저비트 연산 적용.

3.3 실전 적용 사례

- **EAGLES** 논문을 기반으로 투명도를 이용한 가우시안 포인트 제거를 적용.
- **PSNR, SSIM** 유지하면서 모델 크기를 56% 감소, 실시간 응답 속도 20% 향상.

4. SW 테스트에서 정확성, 안정성, 유지보수성을 확인하는 방법

4.1 정확성 확인 방법

- **Baseline Comparison:** 정상적인 출력과 비교하여 오차 검출.
- **Unit Test & Integration Test:** 단위 테스트 및 시스템 테스트 수행.

4.2 안정성 확인 방법

- **Stress Testing:** 극한의 부하를 가하여 안정성 검증.
- **Long-run Testing:** 장시간 실행 후 발생하는 메모리 누수 등을 검출.

4.3 유지보수성 확인 방법

- **정적 코드 분석:** 코드 품질 평가 및 유지보수성 검토.
- **Refactoring & Documentation:** 코드 리팩토링 및 문서화.

5. CI/CD 과정

5.1 CI (Continuous Integration)

- 코드 변경 사항을 정기적으로 통합하고 자동 빌드 및 테스트 실행.

5.2 CD (Continuous Deployment / Continuous Delivery)

- 코드가 배포 준비 상태로 유지되도록 자동화된 테스트 및 검증 수행.

5.3 구강 스캐너 SW의 CI/CD 적용 예시

- 자동화된 3D 렌더링 테스트, 실시간 응답 성능 테스트 등을 포함.
-

6. LPIPS와 SSIM의 차이, 신뢰 가능한 수치 기준

6.1 SSIM (Structural Similarity Index Measure)

- 밝기, 대비, 구조 정보를 고려하여 품질을 평가.
- 0.95 이상이면 원본과 거의 동일한 품질.

6.2 LPIPS (Learned Perceptual Image Patch Similarity)

- 딥러닝 모델을 활용한 인지적 유사성 평가.
 - 0.1 이하일 때 원본과 매우 유사.
-

7. 필터링 설명 (특히 가우시안 필터)

7.1 필터링이 필요한 이유

- 노이즈 제거 및 데이터 스무딩(Smoothing).
- 엣지 보존(Edge Preservation).

7.2 가우시안 필터(Gaussian Filter)

- 가우시안 분포를 기반으로 블러 효과를 제공.
 - 노이즈를 줄이고, Ringing 현상을 최소화.
-

8. 히스토그램 스테레칭 및 평활화 활용

8.1 히스토그램 스테레칭

- 픽셀 값의 범위를 확장하여 명암 대비 향상.
- 구강 스캐너에서 깊이 맵의 대비 조절에 사용 가능.

8.2 히스토그램 평활화

- 픽셀 값의 분포를 균등하게 조정하여 명암 대비 개선.
 - 치아 표면이 너무 밝거나 어두운 경우, 전체적인 명암 균형 조정.
-

9. 구강 스캐너 데이터의 레이턴시를 줄이는 방법

- GPU 가속(CUDA, Vulkan)
 - Pruning 및 Quantization 적용
 - 적응형 렌더링(Adaptive Rendering)
 - 데이터 압축 및 네트워크 최적화
-

10. 3D 모델을 모바일 디바이스에서도 렌더링할 수 있도록 경량화하는 전략

- Mesh Simplification
 - Level of Detail (LOD) 적용
 - 딥러닝 기반 모델 압축 (TensorRT, TFLite)
 - glTF, Draco 포맷 변환
-

11. 실시간 렌더링과 후처리 렌더링의 차이 및 적용 방식

11.1 실시간 렌더링 (Real-time Rendering)

- 1/60초(16.67ms) 이내에 화면을 그려야 함.
- 게임 엔진, 구강 스캐너 실시간 미리보기.

11.2 후처리 렌더링 (Offline Rendering)

- 고품질 이미지를 생성하지만 속도가 느림.
- 3D 애니메이션, 의료용 정밀 모델링.

11.3 구강 스캐너에서 적용 방식

- 실시간 렌더링: 스캔 중 사용자에게 미리보기 제공.
 - 후처리 렌더링: 스캔 완료 후 정밀한 3D 모델 생성.
-

12. 가우시안 필터는 사이드로브가 적어 Ringing 현상이 줄어드는가?

- ✓ 가우시안 필터는 사이드로브가 적기 때문에 Ringing 현상을 줄이는 데 효과적이다.
 - ✓ 다만, 너무 큰 σ 값을 사용하면 엣지가 과도하게 흐려질 수 있으므로 적절한 값 조정이 필요.
-

❖ 최종 정리

- 구강 스캐너 SW에서 딥러닝 및 3D 데이터 최적화가 중요.
- Pruning, Quantization을 활용한 실시간 렌더링 최적화 가능.
- 가우시안 필터 및 히스토그램 변환을 활용하여 이미지 품질 개선 가능.

추가 보완이 필요하면 알려줘! ↗

1. TensorRT란?

1.1 TensorRT 개요

TensorRT는 NVIDIA에서 제공하는 딥러닝 모델 최적화 및 추론 가속화 라이브러리입니다. 이를 활용하면 딥러닝 기반 구강 스캐너 SW의 실시간 성능을 향상시킬 수 있습니다.

1.2 TensorRT의 핵심 기능

- 모델 최적화
 - Quantization (FP32 → FP16 또는 INT8 변환)
 - Layer Fusion (여러 연산을 하나의 커널로 병합)
 - Kernel Auto-Tuning (하드웨어 최적화된 커널 자동 선택)
- 연산 속도 향상
 - Tensor Core 활용 (GPU의 병렬 연산 극대화)
 - Reduced Memory Footprint (메모리 사용량 감소)

1.3 구강 스캐너 SW에서 TensorRT 활용 예시

- 실시간 3D 재구성 가속화
 - 딥러닝 기반 3D 복원 모델을 TensorRT로 최적화하여 실시간 성능 향상.
- 실시간 딥러닝 기반 노이즈 제거
 - CNN 기반 이미지 필터링 모델을 경량화하여 실시간 적용 가능.
- 치아 및 잇몸 분할(Segmentation) 모델 최적화
 - 의료 데이터에서 영역을 분할하는 AI 모델을 고속화하여 실시간 진단 지원.

2. 가우시안 필터란?

2.1 개요

가우시안 필터(Gaussian Filter)는 가우시안 분포(정규분포)를 기반으로 하는 스무딩(Smoothing) 필터입니다. 주로 이미지 처리에서 노이즈 제거 및 블러 효과를 생성하는 데 사용됩니다.

2.2 가우시안 필터의 특징

- 선형 필터(Linear Filter)
 - 필터를 적용한 결과가 원본 데이터의 선형 조합으로 표현됨.
- 부드러운 블러 효과
 - 고주파 성분(급격한 변화)을 완만하게 제거하여 노이즈를 감소.
- Ringing 현상이 적음
 - 필터의 주파수 응답이 점진적으로 감소하여 엣지 부분에서 올림(Ringing) 현상이 적게 발생.

2.3 가우시안 필터 공식

가우시안 필터는 다음과 같은 2D 가우시안 분포를 따르는 커널을 사용합니다.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}}$$

- (σ): 표준 편차 (값이 클수록 블러 효과 증가)

2.4 구강 스캐너에서 가우시안 필터의 활용

- 포인트 클라우드(3D 데이터) 노이즈 제거
 - 치아 스캔 시 발생하는 작은 노이즈를 부드럽게 보정.
- 스캔 이미지에서 잡음 제거
 - 의료 영상에서 발생하는 센서 노이즈를 줄이기 위해 사용 가능.
- 3D 렌더링 데이터 부드럽게 처리
 - 표면이 거친 모델을 보정하여 더 자연스러운 3D 모델 생성.

3. 구강 스캐너에서 사용될 수 있는 필터링 기술

구강 스캐너 SW에서는 노이즈 제거, 대비 향상, 3D 데이터 보정을 위해 다양한 필터링 기법을 적용할 수 있습니다.

3.1 가우시안 필터 (Gaussian Filter)

- 적용 위치: 3D 포인트 클라우드 노이즈 제거, 이미지 부드럽게 보정
- 특징: 노이즈를 부드럽게 제거하며 Ringing 현상이 적음.

3.2 미디언 필터 (Median Filter)

- 적용 위치: 치아 스캔 이미지의 노이즈 제거
- 특징: 주변 픽셀의 중간 값을 적용하여 소금-후추(Salt & Pepper) 노이즈 제거에 효과적.

3.3 Bilateral Filter (양방향 필터)

- 적용 위치: 치아 윤곽 및 경계 보존 필터링
- 특징: 노이즈를 제거하면서도 엣지(경계선)는 보존 가능.

3.4 Wiener Filter (위너 필터)

- 적용 위치: 센서 노이즈 감소 및 이미지 선명화
- 특징: 가우시안 필터보다 신호 보존 능력이 뛰어나며, 신호 대 잡음비(SNR)를 개선.

3.5 히스토그램 스테레칭 (Histogram Stretching)

- 적용 위치: 3D 렌더링 전 깊이 맵(Depth Map) 명암 조절
- 특징: 픽셀 값의 범위를 확장하여 대비(Contrast)를 향상.

3.6 히스토그램 평활화 (Histogram Equalization)

- 적용 위치: 치아 및 잇몸 영상의 명암 균형 조정
- 특징: 전체적인 픽셀 값을 균등하게 조정하여 세부 정보를 복원.

4. 필터링 기술 적용 예시

필터링 기법	적용 위치	기대 효과
가우시안 필터	3D 포인트 클라우드, 이미지 노이즈 제거	노이즈 감소, Ringing 최소화
미디언 필터	치아 스캔 이미지	Salt & Pepper 노이즈 제거
Bilateral 필터	치아 윤곽 강조	엣지를 보존하면서도 노이즈 제거
Wiener 필터	센서 노이즈 제거	신호 대 잡음비(SNR) 개선
히스토그램 스테레칭	깊이 맵 명암 조절	명암 대비 향상
히스토그램 평활화	치아 영상 명암 균형	픽셀 값 분포 균등화

⚖️ 최종 정리

- ✓ TensorRT는 구강 스캐너의 AI 모델을 최적화하여 실시간 성능을 개선하는 데 사용 가능
- ✓ 가우시안 필터는 노이즈를 제거하면서도 Ringing 현상을 줄여 부드러운 결과를 제공
- ✓ Bilateral Filter, Wiener Filter 등 다양한 필터링 기법이 치아 영상 및 3D 데이터 최적화에 활용 가능

추가 질문이 있으면 알려줘! 💬

⚖️ 포트폴리오 기반 면접 예상 질문 및 답변

1. 3DGS 관련 (Improve EAGLES 프로젝트)

Q1. Improve EAGLES 프로젝트에서 기존 EAGLES 모델을 개선한 핵심 방법은 무엇인가요?

답변:

- 기존 EAGLES 모델의 **Pruning(가지치기)** 방식이 정적으로 적용되어, 학습 후반에는 품질 유지 때문에 Pruning이 잘 적용되지 않는다는 문제를 발견했습니다.
- 이를 개선하기 위해 **Pruning** 적용 시점을 학습 일정 단계 이후로 조정하고, 적용 간격을 동적으로 변화시키는 방식을 도입했습니다.
- 초기에는 **Pruning**을 강화하여 모델을 경량화하고, 후반부에는 품질 재구성을 강화하는 방법으로 접근했습니다.

Q2. 동적 Pruning 기법이 기존 정적 Pruning보다 효과적인 이유는 무엇인가요?

답변:

- 정적 Pruning은 일정한 간격으로 적용되기 때문에, 학습 초기에 과도한 Pruning이 발생하면 필요한 Gaussian 포인트까지 제거될 수 있습니다.
- 동적 Pruning은 학습 진행 정도(iteration 수)에 따라 Pruning 비율과 적용 간격을 조절하기 때문에, 초기에는 학습 속도를 높이면서도, 후반에는 품질을 유지할 수 있습니다.

Q3. 이 프로젝트의 성능 향상 결과를 요약해 주세요.

답변:

- 파일 크기 감소: Ply 및 PKL 파일 크기를 각각 **56%** 감소.
 - 연산 성능 향상: 학습 시간 **14.36% 단축**, 렌더링 시간 **20.08% 단축**.
 - 품질 유지: SSIM(0.9221), PSNR(28.207)로 기존 품질을 유지하면서도 모델을 경량화하는 데 성공.
-

Q4. Pruning이 과도하게 적용될 경우 발생하는 문제점과 해결 방법은?

 답변:

- 문제점:
 - Gaussian 포인트가 과도하게 제거되면, 모델이 복잡한 구조를 표현하지 못하고 품질이 저하될 수 있음.
 - 특히, 학습 초기에 너무 많은 Pruning이 적용되면 **Gaussian 포인트의 재구성 속도가 느려지고, 공간 분포가 비효율적으로 유지될 수 있음**.
 - 해결 방법:
 - Pruning 적용 시점을 조정하고, 동적 간격 계산 방식을 활용하여 학습 단계별로 최적화된 Pruning 을 적용.
 - PSNR, SSIM 등 품질 지표를 실시간으로 모니터링하면서 최적의 Pruning 임계값을 조정.
-

2. 임베디드 시스템 프로젝트 (AI 모터 & LED 디바이스 드라이버 통합)

Q5. AI 모델과 C 기반 임베디드 시스템을 연동하는 과정에서 가장 어려웠던 점은 무엇이었나요?

 답변:

- Python 기반 AI 모델(AI.py)과 C 기반 메인 프로그램(main.c)을 리눅스 환경에서 커널 모듈을 활용하여 연동하는 과정이 필요했습니다.
 - AI 모델의 출력을 실시간으로 하드웨어에 적용하려면 **IPC(Inter-Process Communication)** 방식이 필요했는데, 이를 위해 **파이프(Pipe)**를 활용한 데이터 전달 방식을 적용했습니다.
 - 또한, 커널 모듈이 특정 커널 버전에 따라 빌드되지 않는 문제를 해결하기 위해, 커널 헤더와 **GCC** 버전을 맞춰 디바이스 드라이버를 빌드했습니다.
-

Q6. 임베디드 시스템에서 커널 모듈을 사용한 이유는 무엇인가요?

 답변:

- LED 및 모터와 같은 하드웨어 디바이스를 제어하기 위해서는 커널 레벨에서의 접근이 필요했습니다.
 - 커널 모듈을 사용하면 디바이스 드라이버를 동적으로 로드할 수 있고, 시스템 리소스를 효율적으로 관리 할 수 있습니다.
 - 또한, **AI 모델의 판단을 실시간으로 반영하여, 모터 및 LED의 동작을 동적으로 조절할 수 있도록 설계했습니다**.
-

Q7. 이 프로젝트에서 성능 최적화를 위해 고려한 요소는?

답변:

- 커널 모듈 최적화:
 - 불필요한 I/O 연산을 줄이기 위해 메모리 맵핑(Memory Mapping) 기법을 활용하여 커널과 유저 공간 간 데이터 교환을 최소화.
 - 데이터 전송 최적화:
 - AI 모델의 출력을 FIFO(First In, First Out) 방식으로 처리하여 연산 지연을 최소화.
 - 모터 및 LED 동작 최적화:
 - AI 모델의 출력을 일정 주기로 업데이트하여, 모터 및 LED가 너무 빈번하게 동작하지 않도록 주기 조절.
-

3. 멀티미디어 프로그래밍 (MMP 프로젝트 - OpenCV 영상 처리)

Q8. OpenCV를 활용하여 영상 데이터를 처리할 때 가장 효과적이었던 필터링 기법은 무엇이었나요?

답변:

- Bilateral Filter(양방향 필터):
 - 노이즈를 제거하면서도 엣지 정보를 보존하는 데 효과적이었음.
 - 치아의 윤곽을 유지하면서 잡음을 제거할 수 있어 의료 영상 처리에 적합.
 - Fourier 변환을 활용한 주파수 필터링:
 - 특정 패턴을 강조하거나 제거하는 데 효과적이었음.
 - 특히 Hough Transform을 적용하여 치아 윤곽선을 검출하는 실험을 수행함.
-

Q9. 히스토그램 평활화를 적용한 이유와 효과는 무엇인가요?

답변:

- 치아 및 잇몸의 명암 대비가 일정하지 않아, 특정 영역이 너무 어둡거나 밝아지는 문제가 있었음.
 - 히스토그램 평활화를 적용하여 픽셀 값 분포를 조정하고, 전체적인 명암 균형을 조절함으로써 세부 정보를 강조하는 데 성공함.
 - 특히, 치아와 잇몸의 경계를 더욱 선명하게 표현할 수 있도록 개선됨.
-

4. 종합적인 추가 질문

Q10. 포트폴리오의 프로젝트를 기반으로, 실시간성이 중요한 시스템을 설계할 때 고려해야 할 사항은 무엇인가요?

답변:

- 연산 최적화:
 - TensorRT를 활용하여 AI 모델을 최적화하고, 연산 속도를 증가시켜 실시간성을 확보.
 - 메모리 최적화:
 - 임베디드 환경에서는 RAM이 제한적이므로, Quantization을 통해 모델 크기를 줄이는 것이 필수적.
 - 하드웨어 가속 활용:
 - GPU 가속(CUDA), 멀티스레딩 등을 활용하여 실시간 데이터 처리를 강화.
-

Q11. Pruning과 Quantization을 적용하면서 성능과 품질을 동시에 유지하는 방법은?

답변:

- Pruning 시 투명도를 기준으로 Gaussian 포인트를 제거하는 방법을 적용하여 품질 유지.
 - Quantization 적용 시, PSNR, SSIM 등의 품질 지표를 모니터링하여 최적의 비트 수를 선택.
 - 동적 Pruning 간격 조정을 통해 초기에는 모델 크기를 줄이고, 후반에는 품질 재구성을 강화하는 방식으로 조정.
-

🔨 최종 정리

- ✓ 포트폴리오의 프로젝트별 핵심 기술을 요약하고, 최적화 방법을 논리적으로 설명할 준비 필요.
- ✓ 실제 프로젝트에서 발생한 문제와 해결 방법을 구체적으로 정리하면 면접 답변이 더욱 강력해질 것.

추가 질문이 필요하면 알려줘! 💬

🔨 3D Gaussian Splatting(3DGS) 방식 설명

1. 3D Gaussian Splatting(3DGS)란?

- **3D Gaussian Splatting(3DGS)**은 점 기반(**Point-Based**) 렌더링 방식으로,
 - 3D 공간에서 각 포인트를 가우시안(**Gaussian**) 형태의 스플랫(**Splat**)으로 표현하여 렌더링하는 기법입니다.
 - 기존의 메쉬 기반 3D 렌더링보다 효율적이며, 신경망 기반 렌더링(**NERF**)보다 빠르게 실시간 렌더링이 가능합니다.
-

2. 3D Gaussian Splatting의 주요 개념

2.1 Gaussian Splat란?

- 3DGS에서는 3D 공간상의 각 점(**Point**)을 2D 가우시안 블롭(**Gaussian Blob**)으로 표현.
 - 기존 점(**Point Cloud**)보다 부드러운 표현이 가능.
 - 각 Gaussian 포인트는 위치(**Position**), 크기(**Scale**), 방향(**Orientation**), 색상(**Color**), 투명도(**Opacity**), 불투명도(**Alpha**) 등의 속성을 가짐.
-

3. 3D Gaussian Splatting 방식의 핵심 요소

3.1 가우시안 기반 점 렌더링

- 기존 점 기반 렌더링 방식(예: Point Cloud Rendering)과 다르게,
 - 각 점을 Gaussian 분포를 따르는 2D 형태로 표현.
 - 점을 가우시안 블롭(Splat)으로 변환하여 화면에 뿌리듯(Splat) 렌더링.
 - 장점:
 - 점 크기와 투명도를 조정하여 부드러운 3D 모델 표현 가능.
 - 메쉬 기반보다 데이터가 간결하여 실시간 처리에 적합.
-

3.2 Attribute Optimization (속성 최적화)

- 각 Gaussian Splat은 다음과 같은 **속성(attribute)**을 가짐.
 - 위치(Position)**: (x, y, z) 좌표.
 - 크기(Scale)**: 가우시안 분포의 크기 ($\sigma_x, \sigma_y, \sigma_z$).
 - 방향(Orientation)**: 3D 공간에서 회전 정보.
 - 색상(Color) 및 투명도(Opacity)**: 렌더링 색상 표현.
 - 불투명도(Alpha)**: 광선과의 상호작용 반영.
 - 속성 최적화의 목표:
 - 가우시안 포인트의 크기, 투명도를 동적으로 조절하여 **렌더링 품질과 성능을 최적화**.
 - 필요하지 않은 Gaussian을 제거(Pruning)하여 메모리 사용량을 절감.
-

3.3 가우시안 블렌딩 및 합성(Compositing)

- 여러 개의 Gaussian Splat을 투명도(Opacity)를 고려하여 합성(Blending).
 - 핵심 개념:
 - 화면 상에 렌더링될 때 각 Gaussian 포인트가 부드럽게 섞이도록 **Alpha Blending** 수행.
 - 기존 Point Cloud는 점이 독립적이지만, **3DGS는 Gaussian의 중첩 효과를 이용하여 부드러운 이미지 생성**.
-

3.4 실시간 렌더링 최적화

- Gaussian Splat을 GPU에서 병렬 연산하여 **고속 렌더링 지원**.
 - 최적화 방법:
 - Depth Sorting**: 화면에서 가까운 순으로 Gaussian을 정렬하여 렌더링.
 - LOD(Level of Detail) 적용**: 멀리 있는 Gaussian은 크기를 줄여 연산량 최적화.
-

4. 기존 3D 렌더링 기법과 비교

レンダリング 方式	特徴	장점	단점
-----------	----	----	----

レンダリング 方式	特徴	장점	단점
Mesh-based Rendering	삼각형 메쉬를 기반으로 렌더링	높은 정밀도, 표면 표현 용이	연산량이 많고 메쉬 생성 필요
Point Cloud Rendering	점(Point) 단위로 3D 데이터를 표현	가볍고 데이터가 간단	점 간의 부드러운 연결 어려움
NeRF (Neural Radiance Fields)	신경망 기반 3D 볼륨 렌더링	고품질, 물리적으로 정확한 조명 반영 가능	연산량이 많아 실시간 처리 어려움
3D Gaussian Splatting	Gaussian 형태의 점을 렌더링	빠른 연산, 부드러운 표현 가능	Gaussian 포인트 조정 필요

5. 3D Gaussian Splatting의 장점

- 고속 렌더링 → GPU 병렬 처리 최적화로 실시간 렌더링 가능.
- 메쉬가 필요 없음 → 3D 데이터를 직접 Gaussian 형태로 표현 가능.
- 부드러운 시각적 효과 → Gaussian 분포 기반으로 자연스럽게 표현.
- 모바일 및 임베디드 환경에 적합 → 연산량이 적어 경량화 가능.

6. 구강 스캐너 SW에서 3D Gaussian Splatting 적용 가능성

6.1 적용 가능 분야

- 실시간 구강 3D 스캐닝
 - Gaussian Splat을 활용하여 치아와 잇몸의 구조를 부드럽고 빠르게 렌더링 가능.
- 경량화된 3D 데이터 처리
 - 기존의 무거운 메쉬 모델 대신 Gaussian 기반 3D 표현 방식 적용 가능.
- 실시간 의료 데이터 시각화
 - 빠른 렌더링이 필요하므로, 실시간으로 치아 데이터를 표시하는 데 유용함.

7. 3D Gaussian Splatting 적용 시 최적화 방안

Pruning(가지치기) 적용

- 불필요한 Gaussian 포인트를 제거하여 모델을 경량화.

LOD(Level of Detail) 조절

- 가까운 객체는 고해상도로, 멀리 있는 객체는 저해상도로 렌더링하여 연산 최적화.

Alpha Blending 최적화

- Gaussian 블렌딩 연산을 줄여 렌더링 속도 향상.

CUDA 기반 연산 가속

- GPU를 활용하여 Gaussian Splat을 빠르게 병렬 연산.

최종 정리

- ✓ 3D Gaussian Splatting(3DGS)은 기존 Point Cloud보다 부드러운 렌더링이 가능하며, NERF보다 빠름.
- ✓ Gaussian 블렌딩을 활용하여 부드러운 렌더링을 제공하며, GPU 병렬 연산으로 실시간 처리가 가능함.
- ✓ 구강 스캐너 SW에서 실시간 3D 스캔 렌더링 및 경량 3D 데이터 처리에 적합할 가능성이 있음.

추가로 보완할 내용이 있으면 알려줘! ↗

3D Gaussian Splatting(3DGS) 심층 분석

1. 메쉬 기반보다 부드러운 표현이 가능한 이유

- Gaussian Splatting은 3D 공간의 점(Point)을 Gaussian 분포를 따르는 블롭(Blob) 형태로 표현
- 기존 메쉬 기반 방식보다 부드러운 표현이 가능한 이유는 다음과 같습니다.

1.1 메쉬 기반 방식의 한계

- 삼각형(Polygon) 기반 렌더링 방식
 - 표면을 다각형으로 표현해야 하므로, 해상도가 낮거나 삼각형 수가 부족할 경우 계단 현상(Aliasing)이 발생.
- Interpolated Shading이 필요
 - 부드러운 효과를 내기 위해서는 추가적인 버텍스 쉐이딩(예: Gouraud Shading, Phong Shading)이 필요함.

1.2 3DGS의 부드러운 표현 방식

- Gaussian Splat(2D Gaussian Blob) 사용
 - 각 포인트를 부드러운 가우시안 형태로 표현 → 포인트 간의 자연스러운 연결 가능.
- Alpha Blending(투명도 블렌딩) 적용
 - 여러 Gaussian이 겹쳐지면서 자연스럽게 합성 → 경계 부분이 부드럽게 표현됨.
- 색상과 투명도 기반 최적화
 - Gaussian 포인트의 불투명도(Opacity)를 조절하여 부드러운 시각적 효과 제공.

☞ 결론:

→ 3DGS는 점을 Gaussian 형태로 확장하여 표현하므로, 삼각형 메쉬보다 경계가 부드럽고 자연스럽게 보인다.

2. 3DGS가 데이터가 간결한 이유

- 메쉬 기반 방식보다 데이터가 간결한 이유는 다음과 같습니다.

2.1 메쉬 기반 3D 모델의 데이터 구조

- 버텍스(Vertex) + 삼각형(Polygon) + 텍스처(Texture) 정보가 필요
- UV 맵핑(텍스처 좌표) 및 노멀 벡터 저장이 필수적
- 복잡한 표면을 표현하려면 다양한 삼각형이 필요함 → 데이터 크기가 커짐

2.2 3D Gaussian Splatting(3DGS)의 데이터 구조

- 각 Gaussian 포인트는 최소한의 속성만 유지
 - 위치 (x, y, z)
 - 크기 ($\sigma_x, \sigma_y, \sigma_z$)
 - 방향 (Orientation, 3D 회전값)
 - 색상 (RGB 및 Opacity)
 - 불투명도(Alpha, Transparency)
- 삼각형 메쉬 없이 Gaussian 포인트 자체만으로 모델을 표현 가능
- 추가적인 UV 맵핑 및 텍스처가 필요하지 않음

☞ 결론:

→ 3DGS는 복잡한 삼각형 메쉬 데이터를 필요로 하지 않으며, Gaussian 포인트 속성만 저장하면 되므로 데이터가 훨씬 간결하다.

3. 여러 개의 Gaussian Splat을 합성하는 알고리즘

- 3DGS는 화면에 렌더링될 때 여러 Gaussian을 혼합(Blending)하여 최종 이미지를 생성
- 각 Gaussian 포인트는 반투명한 형태로 겹쳐지고, 투명도를 고려하여 블렌딩됨.

3.1 Alpha Blending 기반 가우시안 합성

- 각 Gaussian Splat을 렌더링할 때, 투명도(Alpha)를 고려하여 합성
- 투명한 Gaussian은 뒤쪽 Gaussian과 자연스럽게 겹쳐지도록 조절됨

3.2 Gaussian Weighting (가중치 적용)

- Depth(깊이)와 크기(Scale)를 고려하여 Gaussian의 기여도를 결정
- 깊이가 가까운 Gaussian이 더 큰 영향력을 갖도록 가중치 적용
- 색상(Color)과 투명도(Opacity)도 가중합 방식으로 조정됨

3.3 최종 Alpha Compositing 수식

$$[C_{\text{final}} = C_1 \cdot \alpha_1 + C_2 \cdot \alpha_2 (1 - \alpha_1) + C_3 \cdot \alpha_3 (1 - \alpha_1)(1 - \alpha_2) + \dots]$$

- (C_{final}): 최종 색상 값
- (C_n): 각 Gaussian의 색상 값
- (α_n): 각 Gaussian의 투명도

☞ 결론:

→ Gaussian Splat을 합성할 때, Alpha Blending과 가중치 적용을 통해 자연스럽게 섞이도록 계산한다.

4. 전체 알고리즘 상세 설명

- 3D Gaussian Splatting 렌더링 과정은 다음과 같은 단계로 이루어짐.

- ◇ Step 1: 데이터 로드 및 전처리

1. 3D 스캔 데이터(Ply, PKL 파일)에서 Gaussian 포인트 로드.
 2. Gaussian 포인트의 속성(Position, Scale, Rotation, Color, Opacity) 초기화.
 3. Pruning을 적용하여 불필요한 Gaussian 제거.
-

❖ Step 2: Depth Sorting (깊이 정렬)

1. 화면에서 깊이(Depth) 순서대로 Gaussian을 정렬.
 2. 가까운 Gaussian이 뒤쪽 Gaussian을 가리도록 순서 결정.
 3. LOD(Level of Detail) 기법을 적용하여 먼 Gaussian은 단순화.
-

❖ Step 3: Gaussian Projection (2D 변환)

1. 3D 공간의 Gaussian을 2D 화면에 투영(Projection).
 2. Gaussian의 크기(Scale)와 방향(Orientation)에 따라 **2D 가우시안 커널 생성**.
 3. Gaussian 포인트의 투명도를 기반으로 가중치 부여.
-

❖ Step 4: Alpha Blending (가우시안 합성)

1. 투명도(Alpha)를 고려하여 Gaussian을 렌더링.
 2. 각 Gaussian의 영향을 가중합하여 최종 색상을 결정.
 3. Depth Sorting된 Gaussian 순서대로 Blending 수행.
-

❖ Step 5: 최종 이미지 렌더링

1. 모든 Gaussian이 블렌딩된 결과를 Framebuffer에 저장.
 2. 필요 시, 후처리(Post-processing)를 통해 품질 보정.
 3. 최종 화면에 렌더링하여 사용자에게 출력.
-

❖ 최종 정리

- ✓ Gaussian Splatting은 기존 메쉬 기반보다 부드러운 렌더링이 가능하며, Alpha Blending을 통해 자연스러운 합성이 이루어짐.
- ✓ 데이터 구조가 간결하여 메쉬 없이도 3D 모델을 표현 가능함.
- ✓ Depth Sorting, Gaussian Projection, Alpha Blending의 단계를 거쳐 최종 이미지를 생성함.

☞ 추가 질문이 있으면 알려줘! 그림 결론은 다각형으로 표현하는 메쉬 기반보다 각 포인트를 가우시안 형태로 표현해서

1. 경계가 부드러움, 2 점을 확장하며 더 적은 포인트로 인해 데이터가 적음이라는 장점이 있는건가