2024-SMART-08

# International 2024 SMART Competition

## Sustainable Mission with Art, Responsibility and Technology (SMART 2024)

*presents*

# Incentive Prize

*To*

**안강민, 송용재, 한효원, 박상준**

*(TEAM : 알잘딱깔센 )*

*for their ITEM* **"컴퓨터 비전을 이용한 실시간 로봇 동작 데이터 추출"**

*Presented at the SMART 2024 Competition Held in Online.*

*July 31, 2024*

*Prof. Ki-Bok Min*

*President of the Academic Society for Appropriate Technology*

# Real-Time Robot Motion Data Extraction based on Yolov8 with Catmull-Rom Spline

Yong-Jae Song† , Sang-Joon Park, Kang-Min Ahn, Hyo-Won Han
Department of Industrial Engineering, Konkuk University

# Yolov8 기반 실시간 로봇 객체 탐지 동작 정보 추출 및 결측 값 보간

송용재† , 박상준, 안강민, 한효원*
건국대학교 산업공학과

This study aims to train Yolov8 models utilizing robot video data from real-world corporate processes via open-source platforms, to achieve real-time object detection. We propose an optimization method that enhances data extraction by tracking, annotating, and interpolating detected motion data. YOLO, which predicts multiple bounding boxes and class probabilities simultaneously through convolutional neural network, excels in speed by formulating object detection as a regression problem. In standardized industrial environments, this methodology offers significant advantages for detecting numerous objects efficiently.

Keywords : Object Detection, Convolutional Neural Network, Catmull-Rom Spline

## 1. 서 론

디지털 트윈 혁신을 화두로, 복잡한 공정 환경에서의 로봇 동작 인식 및 개선, 신속한 공정 의사결정이 요구되고 있다. 이러한 요구에 대응하기 위해 CNN(Convolutional Neural Network)의 유효성을 기점으로, 컴퓨터 비전 기술의 성능 검증은 활발히 진행되고 있다. 현대 제조업에서 로봇의 역할은 수많은 기술이 복합적으로 구현된 생산 공정의 핵심 요소이다. 로봇의 작업 환경을 최적화하고, 수집한 데이터를 기반으로 실시간 관리를 통해 생산 효율성을 높이는 것은 중요한 과제다. 현장에서도 유사한 과제 해결을 위해, 수많은 자본과 기술력을 투입하여 생산 공정을 개선하고 발전시키고 있다.

본 연구는 오픈소스를 사용하여 실제 기업 내 공정에서 사용 중인 로봇 영상 정보를 Yolov8 모델을 통해 학습하고, 실시간으로 정확한 탐지 객체의 인식을 목표한다. 추적된 동작 정보를 타점하고, 결측 값 보간 및 최적화 방법을 통한 정보 제공 방법을 제안한다. 효과적 로봇 작업 환경을 기반으로, 디지털 트윈의 가능성을 제시하며 디지털 전환 및 관리 자동화를 기대한다.

## 2. 연구 접근

### 2.1 최소 기능 제품(MVP)

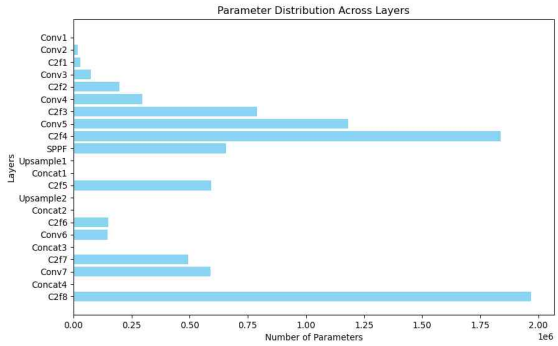
높은 수준에서의 디지털 트윈 구현을 위해서는 이미지 증강을 통해 합성 데이터를 활용하여 CNN

학습을 진행한다. Camera-to-Robot(C2R) 연구[1]에서는 연구주체가 수집한 초기 데이터에, 정의역 난수화(DR)[8]를 삽입하여 빅데이터를 구축한다. 합성 데이터 생성에는 UE4 엔진 플러그인이 공개되어 있다. 이때, 잡음의 의도적 추가[2]를 통한 학습망의 순방향 과정과 이것의 역방향을 학습하는 구조는[3], 시퀀스의 정상성을 높일 뿐만 아니라, Diffusion Model 등, 실험적으로 높은 성능이 증명된 다양한 모델로의 확장을 가능케 한다[32]. 확산 모델(Diffusion Model)의 손실 함수는 이진 크로스-엔트로피(Cross-Entropy)[23]의 형식을 가지며, 이는 Yolov8의 분류 손실 함수[14]에서도 유사하게 사용된다. 이에 더 확장하여 로봇 작업 환경의 동역학계를 로봇의 제어 값을 활용해 정확한 추정을 설계한다[5]. 로봇의 자세 추정(Pose Estimation)에는 PnP 규칙으로 비선형 정사영 매칭 쌍을 최적화하고[6], 리만-기하학 가속도계 규칙(Riemannian Motion Policies)을 통해 움직임의 민감도를 확정 범위 내에서 보간 한다[9].

로봇의 관절(Joint)의 개수에 의해 로봇 움직임의 자유도가 결정되며, 생략 연결 구조를 활용하여 리만-기하학 적용 연구가 확장중이다[16]. 하지만, 고비용을 초래하는 대규모의 학습망은 다차원 계산이 요구되는 3D 정사영 계산[5]에 사용이 적합하며, C2R 구조처럼, 고비용 광학 카메라의 빛의 심도 계산[4]을 통해 접근을 해야 하는 문제이다. 이때, 최소 기능 제품으로써 실시간 처리의 강점을 갖는 YOLO는 산업현장 적용에 있어 모델 학습, 배포가 용이하다[10]. 또한, YOLO-SAM 연구[28] 등, 구조를 단순화하고 재구성하여 연구주체가 저비용의 방향으로 자율적인 개발이 가능하다[26].

## 2.2 학습 구조

Yolov8의 구조는 기존의 검출 박스(Anchor Box) 구조의 지나치게 의존적인 단점을 보완함과 동시에 향상된 실시간 영상 처리 정확도에 주목 받는다[21]. 자율적 검출(Anchor-Free) 특성을 통해 정규화 능력이 향상되어 Decoding 처리가 간단해지고[7], 실시간 처리 등 사후 처리의 시간 복잡도 또한 상당히 줄어든다. 이에 대한 연구는 지속적으로 이뤄진다[17].




<그림1> 모델 Params 분포

Yolov8은 수정된 CSPDarknet53 (Backbone Architecture)(i.e. VGG, EfficientNet, ResNet)[15]을 사용한다. C2F 모듈은 Yolov5에서 사용되는 CSPLayer를 대체한다[23]. PAN(Path Aggregation Network) 구조는 생략 연결(Skip Connection)로 주석 작업을 통합하고 간단하게 만든다[12]. 또한, SPF(Spatial pyramid Pooling Fast) 계층은 배치(Batch)마다 고정된 픽셀 값으로 빠르게 풀링(Pooling) 하도록 가속화하는데[24], 이는 Fast R-CNN에서 제시되었으며, 각 피라미드(pyramid)에서 최대 풀링(Max Pooling)을 하는 RoI(Region of Interest)에 기반한다[11]. 또한 각 합성곱(*) 마다 SiLU를 활성화 함수로 사용한다. 마지막으로, 신경망의 헤드(head)는 객체탐지, 분류, 박스 회귀(Box Regression) 등의 기능으로 구별되어 나간다. 제공된 모델 학습의 손실함수로 Complete-IoU[33] 및 DFL(Distribution Focal Loss)[34]을 활용하여 박스의 손실을 최적화한다. 이러한 구조는 특히 더 작은 객체를 처리할 때 객체 탐지의 성능을 향상시켰다.

$$SiLU(x) = x\left(\frac{1}{1+e^{-x}}\right)$$

<식1> 활성화 함수

<식1> SiLU함수는 출력에 평활성을 도입하여 기울기 소실 문제를 개선한다. ReLU함수와 비교하여, 0 근방에서 미묘한 비선형성을 제공하며, 전파 값이 항상 0인 상태로 갇히는 ReLU의 문제를 개선한다. 그러나, 추가된 계산 리소스에 따라 공간 복잡도가 증가하지만, 하드웨어 자원의 발전에 따라 문제의 심각도는 줄어들고 있다[35].

## 3. 연구 모델

### 3.1 손실 함수 및 갱신 규칙

손실 함수의 결과값이 계산 리소스의 부족을 반증한다면, 각 세트마다 30개의 Epoch을 추가하여 Epoch의 잠재적인 성능을 평가한다. 훈련 과정에서 분류 손실은 다른 손실 간 균형을 맞추기 위해 사용된다[14]. 단위 당 λcls=0.5, λbxs=7.5, λdfl=1.5 지표를 최적점으로 선택했다. 일반화된 손실 함수 및 갱신 규칙 절차는 다음과 같다:

$$L(\theta)=\frac{\lambda_{bxs}}{N_{pos}}L_{bxs}(\theta)+\frac{\lambda_{cls}}{N_{pos}}L_{cls}(\theta)+\frac{\lambda_{dfl}}{N_{pos}}L_{dfl}+\varnothing\parallel\theta\parallel_2^2 \quad (1)$$

$$V^t=\beta V^{t-1}+\nabla_\theta L(\theta^{t-1}) \quad (2)$$

$$\theta^t=\theta^{t-1}-\eta V^t \quad (3)$$

<식2> 손실 함수 및 갱신 규칙

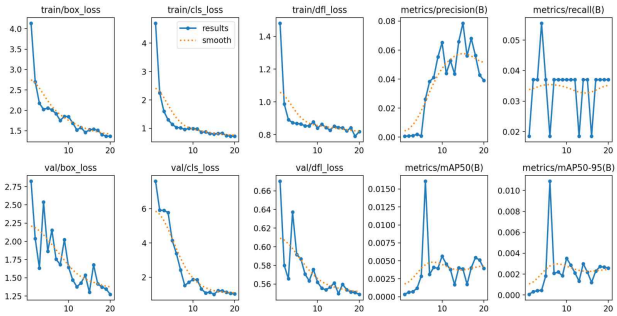
<식2>의 (1)은 가중치 붕괴 ϕ(phi)가 존재하는 개별 손실 가중치를 통합한 정규화 된 함수이고, (2)는 운동량(momentum) β(beta)가 존재하는 속도항, 그리고 (3)은 학습률 η(eta)가 존재하는 갱신 규칙 항이다. 이때, Npos는 객체를 포함하게 되는 셀(cell)의 총합 개수를 나타낸다. 운동량 β는 박스의 일관성을 반증하는 지표인데[24], x좌표, y좌표, 너비, 높이를 포함하는 튜플(Tuple) 형태로 존재하며, 추정의 대상이다.

### 3.2 설계 방안

R-CNN[11]과 달리, YOLO는 단일 CNN이 여러 경계 박스(Bounding Box)와 해당 박스에 대한 클래스 확률을 동시에 예측하며, 객체 탐지 작업이 회귀 문제로 형식화되기 때문에 매우 빠르다[22]. 통일된 규격이 존재하는 산업 현장에서는 대량의 객체를 탐지를 하는데 큰 이점이 있다. 단일 학습을 통해 동일 규격의 여러 객체의 탐지 적용 및 배포가 가능하다. 현장의 니즈에 따라 Custom dataset으로 파인-튜닝(Fine-Tuning) 하여, 객체의 표면적 외형 이외에도, 다양한 인식표를 부착, 색상 구분 등 탐지 방식의 확장성이 본 연구의 의의이다. 작업장 수신호의 비전 처리 관련 연구 또한 존재한다[30]. 직접 학습을 진행하고, 원하는 작업 정보를 실험적으로 도출한다.

### 3.3 모델 학습

본 연구는 작업장 작업 단위의 75%를 학습하고, 나머지 학습과 무관한 25%에 훈련된 모델을 실시간으로 적용한 결과를 사용한다. 촬영 대상은 세 개의 회전 축이 존재하며, 대상의 운동 평면의 직교 방향에 촬영하였다. 촬영된 영상을 30fps 단위로 나누었으며, 먼저 각 개별 영상마다의 25%의 크기조정(Resize)에 가장 경량화 된 블록을 사용하여 학습한다. 이미지의 크기조정은 기본 모델의 손실함수와 결합적으로 작용하며, 최적화 과정은 크기조정에 정규화 제약 조건을 적용하지 않는다[13]. 따라서, 추론의 결과를 선험적으로 통제하지 않으며, 효과적으로 학습 시간을 단축한다. Train/Valid/Test 비율은 7:2:1 이며, 별도 이미지 증강은 진행하지 않았다. Epoch는 20이며, 크게 안정화 되는 모습은 찾을 수 없다.

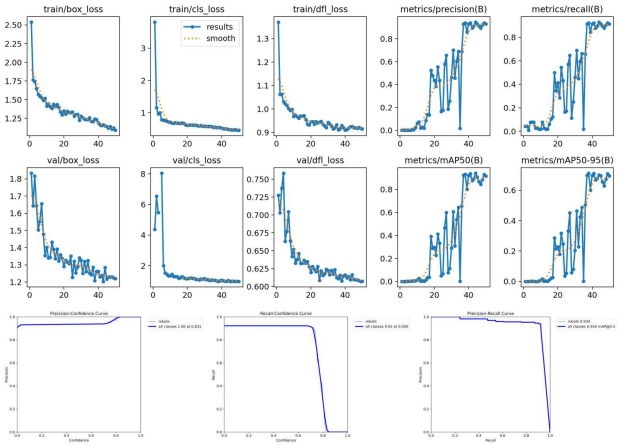



<그림2> 개별학습의 손실함수 및 평가지표 결과

본격적인 통합 데이터 학습에서는 Epoch의 개수를 30개 추가했으며, <표1>에 따라 1/2의 크기조정과, 11.2 M의 params, 28.6의 FLOPs, 1.20의 A100 TensorRT Speed 기반의 모델을 사용한다. 하드웨어 환경은 ARM64 기반에서 원격으로 GPU 백-엔드(Back-End)에 접근하여 가속화했다. 해당 아키텍처가 제시된 M1 Chip(Apple)은 데이터 전송 시간을 최소화 혹은 완전히 제거하기 위해 공유 메모리를 활용한다는 부분에서 혼성(Heterogeneous) 구조로 불리운다[36]. 이런 데이터 전송 없이 CPU와 GPU에 접근하는 알고리즘은 통신의 병목 현상을 효율적으로 제거한다. 해당 디바이스의 단정밀도(Single Precision) 병렬 연산 텐서곱(⊗)은 float 자료형(32bit)을 사용한다.

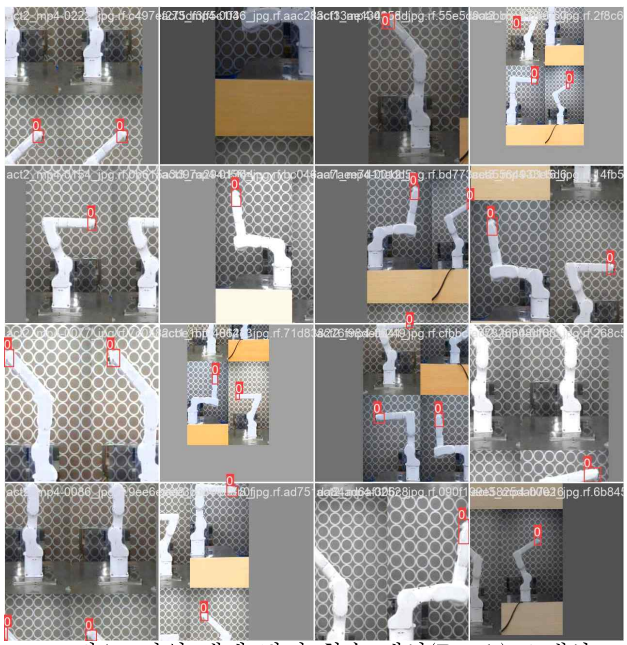
<표1> 모델 활용 지표

| Model | mAPval | CPU ONNX (ms) | A100 TensorRT (ms) | Params (M) | FLOPs (B) |
|---|---|---|---|---|---|
| Yolov8n | 37.3 | 80.4 | 0.99 | 3.2 | 8.7 |
| Yolov8s | 44.9 | 128.4 | 1.20 | 11.2 | 28.6 |



<그림3> 통합학습의 손실함수 및 평가지표 결과

단일 객체 탐지에서는 모델의 전체 탐지 성능은 평균 AP(Average Precision)를 통해 평가한다. AP는 IoU 임계 값에서 모든 객체의 예측 정확도를 평가한다. <그림3>에서 AP는 이 곡선 아래의 면적을 계산하여 구할 수 있다. 탐지된 객체가 1개 이상의 경우는 AP는 탐지된 부분군, 즉 배치(Batch) 사이의 균일성을 반영하지 못한다. 이때 mAP(mean Average Precision)을 사용하여, 배치 사이 이질적인 변동을 반영한다[31]. 통합학습의 평가 결과로, 개별학습보다 안정된 결과를 얻었다.


<그림4> 단일 객체 탐지 학습 배치(Batch) 스냅샷

# 4. 실험 과정

## 4.1 영상 처리

모델 학습과 동일한 조건으로, 실시간 영상을 30fps, 즉 0.033 sec 마다 하나의 영상 프레임을 감지한다. <식2>(2)의 조건에 따라 학습된 β(x, y, w, h)는 최적의 신뢰 점수(Confidence Score)에 따라 추론되며[24], 사용자의 중단 조건이 입력되거나 제한된 영상 길이에 도달하면 종료된다. 이때, 타점 되는 정보는 탐지된 객체의 중앙 C(x, y)를 기준으로, 탐지된 객체의 신뢰 구간 까지의 범위를 넓이와 높이로 지정한다. Yolov8에서는 이를 모두 비율척도로 제공한다. 본 연구에서는 가로 720p, 세로 1280p 실시간 영상을 처리하였고, 곱 연산을 통해 모델의 추론 좌표와 실제 처리 영상의 좌표 간 가역성을 확보하였다.

<표2> 객체 탐지 실행 알고리즘 (OpenCV)

**Algorithm 1** YOLO Object Detection and Data Extraction

**Input:** Video stream from **cap**, YOLO **model**, frame dimensions (720, 1280)
**Output:** DataFrame **df** with bounding box information and frame annotations

Initialize lists: **x_coords**, **y_coords**, **widths**, **heights**, **frame_numbers**
Initialize frame counter **frame_count** to 0
**while** **cap.isOpened()** **do**
  **success, frame** ← **cap.read()**
  **if** success **then**
    **results** ← **model(frame)**
    **if** len(results) > 0 **and** len(results[0].boxes.xyxy) > 0 **then**
      **obj** ← **results[0]**
      **box** ← **obj.boxes.xyxy[0]**
      **x** ← $\frac{(box[0]+box[2])}{2}$
      **y** ← $\frac{(box[1]+box[3])}{2}$
      **w** ← **box[2]** - **box[0]**
      **h** ← **box[3]** - **box[1]**
      Append **x**, **y**, **w**, **h**, and **frame** to respective lists
append({'X': x/720, 'Y': y/1280, 'H': h/1280, 'W': w/720})
      **annotated_frame** ← **obj.plot()**
    **else**
      Append NaN data to **df**:
append({'X': NaN, 'Y': NaN, 'H': NaN, 'W': NaN})
    **end if**
    **frame_count** ← **frame_count** + 1
    **if** **waitKey(1)** & 0xFF == ord("q") **then**
      **break**
    **end if**
  **else**
    **break**
  **end if**
**end while**

<표2>에서 먼저, OpenCV에서 고안되어 있는 속성 값의 인덱스와, <식2>(2)에서의 β(x, y, w, h)가 동일하지 않아, 이를 일치시키는 작업을 진행한다. 객체가 검출되면, 신뢰 점수(Confidence Score)에 따라 추정된 박스의 네 꼭짓점을, 좌측 상단부터 시계 방향으로 총 네 개의 인덱스를 지닌다.

객체의 2차원 평면 상 좌표는 물론, 객체의 너비, 높이 등의 치수는 객체의 운동 방향과는 무관한 변수이지만, 탐지된 객체의 실제 값과의 비교 수단으로 사용되며 추정 결과를 정교하게 가꾼다. <표 2>에서 탐지 결과(annotated frame)의 대시보드(Dashboard)는 객체가 온전히 탐지 되었는지, 혹은 아무것도 탐지 못하여 결측 값이 발생하는지 확인한다. 이때, 결측 값이 발생한 프레임 마다 지정하고 모두 정렬하여, 최종적으로 한 주기의 결과 값을 저장한다.

## 4.2 매개변수 곡선

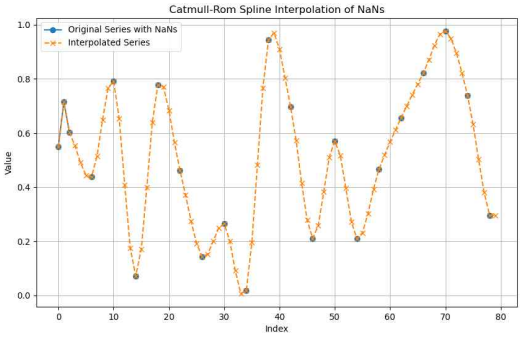
매개변수 방정식을 활용하여, 2차원 평면 내에서의 타점 된 로봇의 동작 정보를 시간의 흐름에 따라 사상(Mapping)시키면, 물체의 궤적을 매개 변수로 시간에 따른 흐름과 배향을 방정식으로 나타낸다. 일반적으로 극좌표계에서 직교 좌표계로의 변환은 고유한 단일 값을 상정하지 않는다. 이를 해결하기 위해 시간을 매개변수로써 활용하고, 본 연구에서 시간에 흐름에 따라 생성된 배향의 차이를 고유하게 정의한다. 베지에 곡선(Bézier Curve)은 컴퓨터 그래픽스에서 가장 일반적인 매개변수 곡선이다[25].

$$B(t)=\sum_{i=0}^{n}\binom{n}{i}(1-t)^{n-i}t^{i}P_i,\quad 0\le t\le 1$$

<식3> 베지에 곡선(Bézier Curve)

<식3>은 번스타인(Berstein) 다항식의 급수이다. 탐지된 객체의 제어점 집합을 연속적인 곡선의 형태로 변환시키며, 다른 관점에서는 임의의 불연속점 보간에 활용될 수 있다[18]. 본질적으로 곡선은 수학적 표현이 없거나 표현이 알려지지 않았거나 너무 복잡한 실제 모양을 근사하기 위한 것이라는 해석학적 관점을 활용한다[27]. <식3>에서와 같이 급수로 정의된 함수를 통해 매개화 한다. <식3>의 n값을 설정함에 따라, 총 제어점의 수를 정의하여, 이에 따라 주체가 노출을 결정을 할 최대 차수를 정한다[19]. 이를 '보간'의 관점에서 바라본다면, 허용할 변동의 최대의 범위를 n값을 통해 지정할 수 있다. 본 연구에서는 보간 매개변수 함수 족(Family) 중, 캣멀-롬 보간 기법을 사용한다.

## 4.3 결측 값 보간 기법(Catmull-Rom Spline)




<그림5> 결측 값 보간 난수화 테스트

매개변수로의 사상(Mapping)이라는 아이디어를 파생시켜 최적의 기법을 실험적으로 도출한다. 캣멀-롬 보간 기법은 '탄력'이라는 개념이 베지에 곡선과 비교된다. 탄력도(τ) 설정을 통해 변동 정도를 조율한다[27]. 탄력도 낮아질수록, 결과값 곡선의 구부러지는 정도는 낮아진다. 또한, <식4>에서 τ=1로 설정되면, 제어점들 사이의 진행상태는 베지에 곡선에서의 n=1의 진행상태, 즉 직선의 진행상태를 가진다. 하지만, 제어점의 개수(차원) 등 형식 자체는 베지에 곡선에서의 n=3의 형식을 따른다. 이 기법의 가장 큰 장점은, 보간 전 원점 집합들이 보간 후 새로운 집합에서 제어점으로 기여되며, 보간의 왜곡 정도를 최소화 시키는 것이다.

$$CatmullRom(t)=\frac{1}{2}\cdot\begin{bmatrix}1 & t & t^2 & t^3\end{bmatrix}\cdot\begin{bmatrix}0 & 2 & 0 & 0\\ -\tau & 0 & \tau & 0\\ 2\tau & \tau-6 & -2(\tau-3) & -\tau\\ -\tau & 4-\tau & \tau-4 & \tau\end{bmatrix}\cdot\begin{bmatrix}P_0\\ P_1\\ P_2\\ P_3\end{bmatrix}$$

<식4> Catmull-Rom Spline 함수, τ=1(탄력도)

임의의 불연속점에 대해, 양쪽 끝에 두개의 추가점을 구성하고, 캣멀-롬 보간 기법을 실행하여 자체 교차점(Self-Intersection) 루프를 형성한다. 탄력도 조절에 따라 닫힌 형태(Closed-Form)의 연산을 실행하여 비교적 낮은 공간 복잡도를 지닌다. 특히, 보간을 통해 생성된 곡선의 접선이 여러 분할에 걸쳐 연속적임을 보장하기 때문에 높은 안정도가 있다[30].

<표3> 결측 값 보간 알고리즘

**Algorithm 2** Catmull-Rom Spline Interpolation for NaN Values

**Input:** Series **series**
**Output:** Series with NaN values interpolated

```
function INTERPOLATE_NANS_CATMULL_ROM(series)
    nans ← series.isna()
    not_nans ← not nans
    indices ← arange(len(series))
    if sum(not_nans) < 4 then
        return series
    end if
    interp_values ← series.copy()
    idxs ← indices[not_nans]
    for i in indices[nans] do
        pos ← searchsorted(idxs, i)
        if pos = 0 then
            interp_values[i] ← series[idxs[0]]
        else if pos ≥ len(idxs) then
            interp_values[i] ← series[idxs[-1]]
        else
            before ← idxs[pos - 1]
            after ← idxs[pos]
            before_2 ← idxs[pos - 2] if pos - 2 ≥ 0 else before
            after_2 ← idxs[pos + 1] if pos+1 < len(idxs) else after
            t ← (i − before)/(after − before)
            interp_values[i] ← catmull_rom_spline(series[before_2], series[before], series[after], series[after_2], t)
        end if
    end for
    return interp_values
end function
```

$$m_n = \frac{f(n+1)-f(n-1)}{2} = \frac{p_{n+1}-p_{n-1}}{2}$$

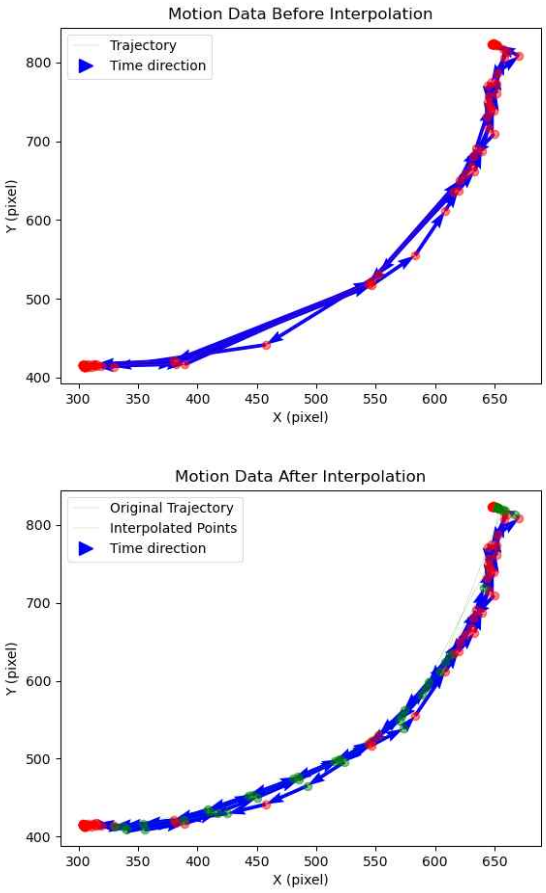
<식5> 이진 탐색 목적 함수

영상 처리 과정을 통해 타점 된 객체의 동작 정보는 보간 알고리즘에 입력되고, 이진 탐색을 통해 결측 값 위치가 선언 및 배열된다. <표3>의 알고리즘에 따라, 결측 값과 결측 값이 아닌 인덱스를 식별한다. 결측 값이 아닌 위치가 4개 미만인 경우, 보간 기법을 적용할 수 없으므로 원본을 그대로 반환한다. 결측 값을 처리하기 위해, 배열을 순차적으로 순회하면서, 결측 값의 경우, 해당 위치를 기준으로 가장 가까운 주변의 네 개의 제어점을 찾는다. 찾은 네 개의 제어점 중 두 개는 <식5>에 따라 결측 값 이전의 값, 두 개는 결측 값 이후의 값이 된다. 이후, 현재 결측 값의 위치를 네 개의 제어점 사이에 정규화 된 비율로 계산하여, 캣멀-롬 보간 연산을 통해 값을 출력한다. 최종적으로, 보간 전 원형과 비교하여 보간 후 변동의 정도를 확인한다.

# 5. 실험 결과

## 5.1 보간 결과

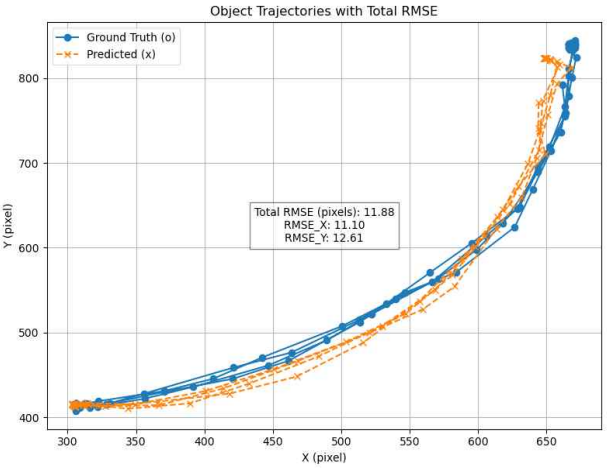
영상 처리 이후 캣멀-롬 보간 기법 적용 결과를 보간 전/후로 비교한다. <그림6>에서 빨간 점은 원본의 측정 값, 파란 화살표는 시간 흐름에 따른 움직임의 방향, 우측 그래프에서 초록 점은 보간 후 정보를 의미한다. 보간을 통한 결과 분석의 정성적 평가 지표에는 데이터 연속성, 궤적의 부드러움, 시간 방향의 일관성 등이 있다. 보간 기법을 사용하지 않은 자료에서는 결측 값으로 인해 포인트 사이의 간격이 불규칙하고 그래프의 궤적이 불완전하고, 시간의 흐름에 따른 움직임 방향이 명확하게 드러나지 않고 궤적의 연속성이 부족하다. 이에 반해 보간 후 결과에서 보간 된 동작 정보가 원본 측정 값의 흐름을 따라 궤적을 부드럽게 잇는다. 시간의 흐름에 따른 움직임 방향이 명확히 드러나며 궤적의 연속성이 향상되었다.




<그림6> 보간 전/후 동작 정보 예시

## 5.2 실측 자료 대비 결과

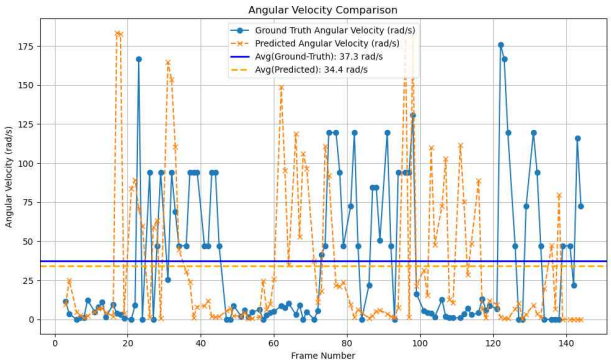
최종적으로, Yolov8의 객체 탐지 성능과 결측 값 보간 기법의 정확도를 측정하기 위해, 사람이 직접 관측한 디지털 정보를 실측 자료(Ground-Truth)로 정의하고, 각각의 오차를 계산했다. 수작업으로 진행되던 계측 작업 형식에서 발전하여, 학습된 비전 인식 모델의 탐지 결과를 바탕으로 모델의 결측 값 보간 결과까지의 자동화 결과를 제시한다. 관측 좌표계는 pixel 단위로 측정한다. 로봇 동작의 90° 방향에서의 촬영 환경을 전제한다. 좌표계의 실측 길이로의 변환은 영상 심도(Depth)의 추정 문제로 귀결되어, 광학카메라의 구성 조건을 제외하고 관측을 진행한다.




<그림7> 추정 값의 누적 오차율(pixels) 예시

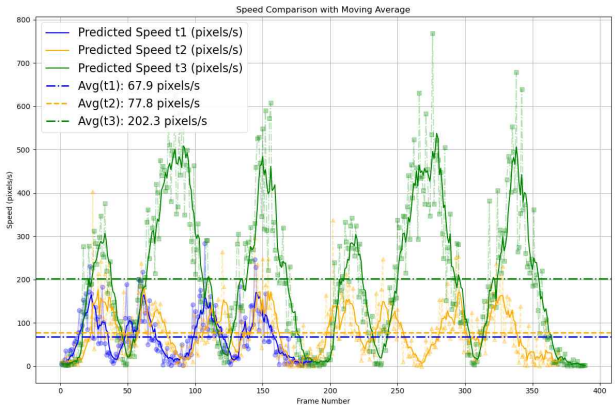
모델의 편의 추정 값에서 평균 제곱근 오차는 분산의 제곱근, 즉 표준 오차가 된다. 평균 제곱근 오차(RMSE)는 각 프레임 단위를 시간의 흐름으로, 잔차 제곱을 누적하여 모델의 정밀도를 표현하는데 적합하다. 제곱근의 성질로 인해 이상치에 덜 민감하다. 본 연구에서는 실험 결과, 다양한 매개변수 함수 족(Family) 중 캣멀-롬 보간 기법이 11.88의 가장 낮은 오차 지표를 가졌다. <그림7>의 예시 동작 영상에 경우, 주기의 37/145 및 82/145 지점 전후로 가장 큰 편의를 보인다.

각 부분마다 편의를 자세하게 확인하기 위해 각속도를 측정한다. 좌표계의 실측 단위 추정을 제외한 상태로 각 로봇 관절 회전의 선형 결합으로 발생한 각 운동 정보는 단위 변환 없이 측정 가능하다.




<그림8> 각속도 비교 분석 예시

동작 정보의 차분 값을 이용해 탄젠트(tan)를 구하고, 역 탄젠트(arctan)연산을 통해 각위치(rad)의 변위를 프레임 당 0.033 sec의 단위로 각속도(rad/s)를 계산한다. <그림8>에서 제시된 프레임별 실측 자료(Ground-Truth)와 모델의 평균 각속도의 오차는 2.9 rad/s이며, 직교 좌표에서 편의가 발생한 부분에서 동일하게 지연 시간(Lag)이 발생한다. 결측 값 보간 횟수가 누적될수록 시간 지연(Lag)의 크기가 증가한다.




<그림9> 동일 속도 제어, 개별($t_i$) 동작 속도 비교

<그림9>에서 추가적으로, 새로운 세 개의 동작 샘플에 대해 속도 비교를 통한 모델의 성능 검증을 진행했다. 각속도 측정은 3개 샘플 평균 24.87%의 오차율을 기록했다. 오차의 원인으로는 촬영 각도의 흔들림, 상이한 영상 처리 규격, 촬영 심도 미측정 등으로 진단되며, 이 중 촬영 심도에 관한 문제는 속도 비교에서 더욱 분명하다. 동일한 속도 제어를 바탕으로, 개별 동작의 단위(pixel) 속도의 거시적인 움직임에서는 동일한 패턴을 보였다. 심도 측정 및 탐지 객체의 실측 길이를 활용한 보정을 통해 정밀한 속도 계측이 가능하다.

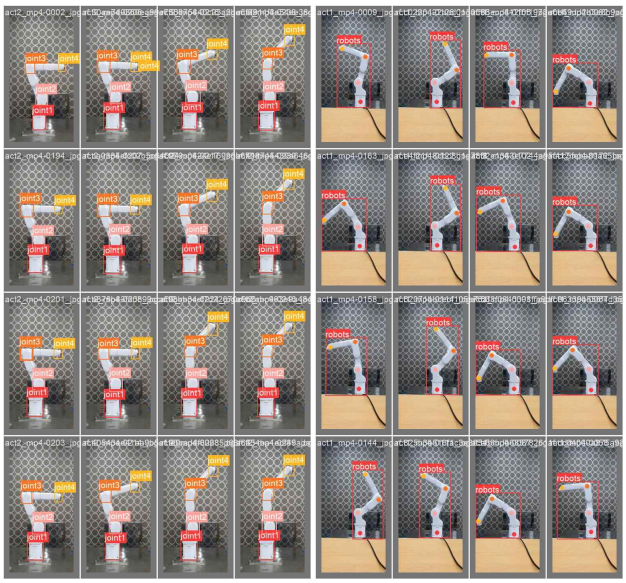
## 6. 결 론

### 6.1 타당성

Yolov8 모델을 기반, 실시간 객체 탐지를 통해 동작 정보 추출하고, 캣멀-롬 보간 기법을 통해 불규칙한 간격이 현저히 개선되었다. 하지만, 더욱 미시적인 조건에서 관찰하면 시간 지연(Lag) 등 왜곡 요인을 쉽게 발견할 수 있다. 본질적으로 결측값이 생기는 원인은 학습 입력 값의 양(Quantity)적인 문제로 진단된다[29]. 표준화 된 환경에서 대량의 학습을 전제로, 부득이하게 발생하는 결측 값의 효율적인 처리 방식에 대해 제안한다. 초-매개변수(Hyper-Paramter)를 조율하여 움직임의 탄력을 반영하여 왜곡의 정도를 관리한다[25]. 객체 탐지 알고리즘의 가장 큰 강점은 실시간 영상의 동작 정보가 프레임 조정을 통해 시간 단위로의 변환이다. 특정 프레임의 어떤 부분에 변동 요인이 생겼는지 부수적으로 파악이 가능한 점을 토대로, 저비용-최소 기능의 자율적인 개발을 지향한다. 본 연구에서 선택된 방안은 동작 정보의 최소 요인을 바탕으로 최적의 연산을 수행한다.

### 6.2 연구 확장

최소 기능에서 확장하여, 광학카메라 등 촬영 심도(Depth) 등의 요인을 추가한다면, 대표적으로 모델 트랜스포머(Transformer) 기반의 6D 자세 추정 모델 등 관련 연구 등을 참조할 수 있다[20]. 영상의 맥락을 구성하기 위해 촬영의 심도(Depth) 등을 통계적 일관성과 같은 평가 지표를 통해 모델을 학습한다. <그림9>에서의 문제상황에서 발전하여, 촬영 심도, 각도 제시를 통해 개선하고, 객체의 동작 정보는 단일 객체의 움직임이라는 제한점에 벗어나, 먼저 <그림10> 다중 객체 학습을 제안한다. Yolov8은 하나 이상에 객체에 대해서도 높은 성능을 보였다. 객체마다 각위치 변위를 추출하여, 각 관절의 각운동의 선형 결합이 올바르게 반영되었는지 확인한다면 동작 정보 추출의 타당성을 제고시킬 것으로 기대된다자세 추정(Pose Estimation) 및 관절 회귀(Joint Regression) 문제 또한 연구 발전 방안으로 제시된다. 촬영 각도가 동작의 90° 방향에서 벗어난다면, 제기될 문제점이지만, 3D 상황에서의 관련 보간 기법 또한 존재한다[19].촬영 심도 추정을 통해 최종적으로 동작 정보의 단위 좌표계까지 단계적으로 추정한다.



<그림10> 다중 객체 탐지(좌) 및 관절 구조 추정(우)
학습 배치(Batch) 스냅샷

<그림10> 관절 구조 추정은 기본적으로 다중 객체 탐지를 기반한다. 자세 추정의 주 목적은 관절의 움직임을 확인하고, 추출된 정보를 바탕으로, 보다 자연스러운 움직임을 개선하는 데에서 객체 탐지와 큰 차이점이 있다[9]. YOLO 모델은 지속적으로 사용 환경이 개선되는 대표적인 오픈소스 상용화 비전 모델이다. 객체 탐지는 컴퓨터 비전의 가장 기초적 수단인 동시에 그 활용 가능성은 다양했다. Yolov8은 객체 탐지에서 효과적인 범용성과 적응성을 제공했다.

## 7. 참고문헌


[1] Lee, T., Tremblay, J., To, T., Cheng, J., Mosier, T., Kroemer, O., Fox, D. and Birchfield, S. (2020) Camera-to-robot pose estimation from a single image, arXiv.org. Available at: https://arxiv.org/abs/1911.09231.

[2] Chen, X., Wang, X. and Xuan, J. (2018) Tracking multiple moving objects using unscented Kalman filtering techniques, arXiv.org. Available at: https://arxiv.org/abs/1802.01235.

[3] Kalita, D. and Lyakhov, P. (2022) Moving object detection based on a combination of Kalman filter and median filtering, big data & cognitive computing, 6(4). doi:10.20944/preprints202209.0109.v1.

[4] Wang, Y.-T., Sun, C.-H. and Chiou, M.-J. (2012) Detection of moving objects in image plane for robot navigation using monocular vision, EURASIP Journal on Advances in Signal Processing, 2012(1). doi:10.1186/1687-6180-2012-29.

[5] Zhao, Y., Wang, Y. and Cui, Q. (2014) Calculating intrinsic and extrinsic camera parameters based on the PNP problem, Journal of Engineering and Technological Sciences, 46(3), pp. 258 - 270. doi:10.5614/j.eng.technol.sci.2014.46.3.2.

[6] Tremblay, J., To, T., Sundaralingam, B., Xiang, Y., Fox, D., Birchfiled. (2018) Deep object pose estimation for semantic robotic grasping of household objects, arXiv.org. Available at: https://arxiv.org/abs/1809.10790.

[7] Le, N., Rathour, V., Yamazaki, K., Luu, K. and Savvides, M. (2021) Deep Reinforcement Learning in computer vision: A comprehensive survey, arXiv.org. Available at: https://arxiv.org/abs/2108.11510.

[8] James, S., Wohlhart, P., Kalakrishnan, M., Kalashnikov, D., Irpan, A., Ibarz, J., Levine, S., Hadsell, R. and Bousmail, K. (2019) Sim-to-real via sim-to-SIM: Data-efficient robotic grasping via randomized-to-canonical adaptation networks, arXiv.org. Available at: https://arxiv.org/abs/1812.07252.

[9] Ratliff, N.D., Issac, J. and Kappler, D. (2018) Riemannian motion policies, arXiv.org. Available at: https://arxiv.org/abs/1801.02854.

[10] Girshick, R., Donahue, J., Darrel, T. and Malik, J. (2014) Rich feature hierarchies for accurate object detection and semantic segmentation, arXiv.org. Available at: https://arxiv.org/abs/1311.2524.

[11] Girshick, R. (2015) Fast R-CNN, arXiv.org. Available at: https://arxiv.org/abs/1504.08083.

[12] He, K., Zhang, X., Ren, S. and Sun, J. (2015) Deep residual learning for image recognition, arXiv.org. Available at: https://arxiv.org/abs/1512.03385.

[13] Talebi, H. and Milanfar, P. (2021) Learning to resize images for computer vision tasks, arXiv.org. Available at: https://arxiv.org/abs/2103.09950.

[14] Xie, X., Cheng, G., Wang, J., Yao, X. and Han, J. (2021) Oriented R-CNN for object detection, 2021 IEEE/CVF International Conference on Computer Vision (ICCV) [Preprint]. doi:10.1109/iccv48922.2021.00350.

[15] Thakur, A., Chauhan, H. and Gupta, N. (2023) Efficient ResNets: Residual Network Design, arXiv.org. Available at: https://arxiv.org/abs/2108.05699.

[16] Katsman, I., Chen, E., Holalkere, S., Asch, A., Lou, A., Lim, S. and Sa, C. (2023) Riemannian Residual Neural Network, arXic.org. Available at: https://arxiv.org/abs/2310.10013.

[17] Wang, C., Yeh, I. and Liao, H. (2024) YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information, arXiv.org. Available at: https://arxiv.org/abs/2402.13616.

[18] Xiaolin Luo, Pavel V. Shevchenko2 (2014) Fast and Simple Method for Pricing Exotic Options using Gauss-Hermite Quadrature on a Cubic Spline, arXiv.org. Available at: https://arxiv.org/abs/1408.6938.

[19] Francesc Arandiga, Antonio Baeza, Dionisio F. Yanez (2021) Monotone cubic spline interpolation for functions with a strong gradient, arXiv.org. Available at: https://arxiv.org/abs/2102.11564.

[20] Amini, A., Periyasamy, A. and Behnke, S. (2022) YOLOPose: Transformer-based Multi-Object 6D Pose Estimation using Keypoint Regression, arXiv.org. Available at: https://arxiv.org/abs/2205.02536.

[21] Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X. and Wei, X. (2022) YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications, arXiv.org. Available at:

https://arxiv.org/abs/2209.02976.
[22] Schneidereit, S., Yarahmadi, A., Schneidereit, T., Breuß, M. and Gebauer, M. (2023) YOLO-based Object Detection in Industry 4.0 Fischertechnik Model Environment, arXiv.org. Available at: https://arxiv.org/abs/2301.12827.
[23] Terven, J., Esparza, D. and Gonzalez, J. (2023) A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS, machine learning & knowledge extraction, 5(4), pp. 1680-1716, Available at: https://doi.org/10.3390/make5040083.
[24] Reis, D., Kupec, Z., Hong, J. and Daoudi, A. (2023) Real-Time Flying Object Detection with YOLOv8, arXiv.org. Available at: https://arxiv.org/abs/2305.09972.
[25] V.A. Baturin, W. Däppen, A.V. Oreshina, S.V. Ayukov, A.B. Gorshkov (2019) Interpolation of equation-of-state data, arXiv.org. Available at: https://arxiv.org/abs/1905.08303.
[26] Gao, P., Ji, C., Yu, T. and Yuan, R. (2024) YOLO-TLA: An Efficient and Lightweight Small Object Detection Model based on YOLOv5, arXiv.org. Available at: https://arxiv.org/abs/2402.14309.
[27] Soroosh Tayebi Arasteh, Adam Kalisz (2021) Conversion Between Cubic Bezier Curves and Catmull – Rom Splines, arXiv.org. Available at: https://arxiv.org/abs/2011.08232.
[28] Zhu, Y., Yang, Q. and Xu, Li. (2024) Active Learning Enabled Low-cost Cell Image Segmentation Using Bounding Box Annotation, arXiv.org. Available at: https://arxiv.org/abs/2405.01701.
[29] Masum, M., Sarwat, A., Riggs, H. and Boymelgreen, A. (2024) YOLOv5 vs. YOLOv8 in Marine Fisheries: Balancing Class Detection and Instance Count, arXiv.org. Available at: https://arxiv.org/abs/2405.02312.
[30] Antonio Baeza, Zehuan Yu, Huan Yin and Shaojie Shen (2023) Online Monocular Lane Mapping Using Catmull-Rom Spline, arXiv.org. Available at: https://arxiv.org/html/2307.11653.
[31] Zhijian Qiao, M. and Govindaraju, K. (2024) Unveiling the Advancements: YOLOv7 vs YOLOv8 in Pulmonary Carcinoma Detection, Journal of Robotics Control, 5(2). pp. 459-470. doi: 10.18196/jrc.v5i2.20900.
[32] Jonathan Ho, Ajay Jain, Pieter Abbeel (2020) Denoising Diffusion Probabilistic Models, arXiv.org. Available at: https://arxiv.org/abs/2006.11239.
[33] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren (2020) Distance-iou loss: Faster and better learning for bounding box regression, in Proceedings of the AAAI conference on artificial intelligence, vol. 34, pp. 12993 – 13000.
[34] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, and J. Yang (2020) Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection, Advances in Neural Information Processing Systems, vol. 33, pp. 21002 – 21012
[35] Stefan Elfwing, Eiji Uchibe, Kenji Doya (2017) Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning, arXiv.org. Available at: https://arxiv.org/abs/1702.03118.
[36] Connor Kenyon, Collin Capano (2022) Apple Silicon Performance in Scientific Computing, arXiv.org. Available at: https://arxiv.org/abs/2211.00720.