한국정보전자통신기술학회논문지(jkiiect)'24-10, Vol.17 No.5
논문 24-17-05-339 http://dx.doi.org/10.17661/jkiiect.2024.17.5.339

시뮬링크 기반의 실시간 모니터링 및 로깅 도구 개발

홍윤빈*, 박민지*, 안동혁**

Simulink-based xPC Target Monitoring/Logging Tool Development

Yoonbin Hong*, Minji Park*, Donghyeok An**

요 약 현재 건설 현장 내 중장비 엔진은 실무자가 엔진 설정을 변경하며 직접 출력을 테스트하고 있다. 이에 따른 시간 비용과 안전 사고의 위험성이 꾸준히 제기되어 왔다. 이를 해결하기 위해서 Speedgoat와 Simulink API를 사용해 중장비 시뮬레이션을 수행한다. 하지만 시뮬레이션에 필요한 Speedgoat 하드웨어와 Simulink API의 버전 별 호환성이 달라 엔지니어가 다양한 Simulink API에 대한 이해가 필요하다. 중장비 구조에 대한 이해가 필수적인 엔지니어들이 API를 포함한 프로그래밍 역량까지 갖추기는 현실적으로 어렵다. 따라서 본 논문에서는 중장비 시뮬레이션을 위해 설정값을 입력하고 이에 따른 시뮬레이션의 결과값을 시각적으로 출력하고 로깅하는 도구를 제안한다. 제안하는 도구는 중장비의 엔진 등 설정값에 시뮬레이터 모델에 전달하고, 이에 따른 시뮬레이션 출력값을 모니터링 및 로깅을 할 수 있는 있는 기능을 제공한다. 제안한 도구에서 제공하는 기능들은 시나리오를 통해서 검증하였다. 개발한 도구를 사용하면 엔지니어들은 Simulink API 학습에 대한 부담을 줄일 수 있고 중장비 구조를 이해하는데 중점을 둘 수 있을 것으로 예상된다. 또한 건설 현장 내의 중장비 테스트에서 효율적이고 안전한 업무 환경을 제공할 수 있을 것으로 기대된다.

Abstract In construction sites, the engine of heavy machinery is tested by practitioners who manually adjust engine settings and directly measure the output. This process has consistently raised concerns regarding time costs and the risk of incidents. To address these issues, simulations of heavy equipment are conducted using Speedgoat and the Simulink API. However, due to the varying compatibility of different versions of Speedgoat hardware and Simulink API, engineers need to have a comprehensive understanding of various Simulink APIs. It is practically challenging for engineers, who must have a deep understanding of heavy equipment structures, to also possess programming skills including API usage. Thus, this paper proposes a tool that allows inputting configuration values for heavy equipment simulation and visually outputs and logs the simulation results. The proposed tool provides functionalities to deliver configuration values, such as engine settings of heavy equipment, to the simulator model and to monitor and log the resulting simulation outputs. These functionalities have been validated through scenarios. By using the developed tool, engineers are expected to reduce the burden of learning Simulink API and focus more on understanding the structure of heavy equipment. Additionally, it is anticipated that this tool will provide a more efficient and safer working environment for heavy equipment testing on construction sites.

Key Words : Logging, Monotoring, Simulink, Speedgoat, Visualization

1. 서론

건설 현장 내의 중장비는 엔진의 설정값에 따라서 동작한다. 설정값은 엔진의 RPM(Revolution Per Minute), 엔진의 내부 및 외부 온도나 압력 등이 될 수 있다. 엔지니어는 이러한 엔진의 설정값을 조정하고,

Following are results of a study on the "Leaders in INdustry-university Cooperation 3.0" Project, supported by the Ministry of Education and National Research Foundation of Korea
*Department of Computer Engineering, Changwon National University
**Corresponding Author: Department of Computer Engineering, Changwon National University (donghyeokan@changwon.ac.kr)
Received September 27, 2024 Revised October 13, 2024 Accepted October 16, 2024

그에 따라 출력을 테스트하며 적절한 설정값을 도출해 중장비의 성능, 내구성 및 효율을 최적화하고 안정성을 높인다. 그러나 사람이 중장비에 직접 다가가서 엔진의 설정값을 변경하고 출력을 확인하는 작업은 중장비의 가동 중 발생할 수 있는 안전사고 위험이 있고, 조정과 테스트 과정에서 많은 시간과 노력이 소요된다. 따라서 엔진 설정을 포함해 다양한 분야에서 시뮬레이터를 활용하는 것처럼, 중장비 시뮬레이터를 기반으로 설정 최적화를 위한 범위를 설정한다 [5, 6, 7, 9].

Mathworks에서 개발한 Speedgoat라는 하드웨어와 'Simulink Real-Time' 모델은 이러한 작업을 대체한다[1, 2, 10]. Speedgoat는 중장비의 설정값을 조정하고 그에 따른 출력을 시뮬레이션할 수 있는 하드웨어 장비이다. 시뮬레이션에 대한 많은 인터페이스가 지원되어 제어 시스템이나 애플리케이션을 개발하고 테스트하는 과정에서 드는 비용과 시간을 크게 단축한다. 사용자가 설정시 사용하는 Host PC는 시뮬레이션을 위해서 LAN과 TCP/IP 통신을 기반으로 Speedgoat와 연결한다. 그림 1은 시스템 개요를 나타낸다. Speedgoat 내에 삽입하는 Simulink Real-Time 모델은 개별 중장비의 설정에 해당하는 입력과 그에 따른 출력을 정의하는 시뮬레이션 모델이다. 이들을 정의하는 블록은 Signal, 파라미터, 계산 블록 등이 있다. Signal은 계산으로 도출되는 출력이며, 파라미터는 여러 계산 블록을 거쳐 Signal에 영향을 주는 값이다. Signal의 출력을 계산하는 과정을 Speedgoat가 수행한다.

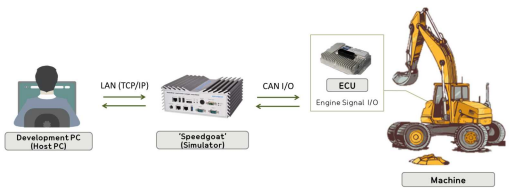

그림 1. 시스템 개요도
Fig. 1. System overview

Speedgoat와 시뮬레이션 모델을 활용해 Host PC에서 시뮬레이션 수행이 가능하지만, 연계를 위해서는 엔지니어가 Simulink API를 사용한다. 따라서 엔지니어들은 Simulink API를 포함해 프로그래밍 능력이 필요하다. Speedgoat 버전에 따라 사용 가능한 Simulink API 버전이 상이하기 때문에 엔지니어들은 다양한 Simulink API 활용 능력을 갖추어야 한다. 하지만 다양한 환경에서 중장비 설정 시뮬레이션을 위해서는 Simulink API 조작을 위한 프로그래밍 역량과 효율적인 설정을 위한 중장비 구조의 이해가 요구되지만 하지만 엔지니어들이 중장비 구조와 프로그래밍 역량을 동시에 갖추기는 현실적으로 어렵다. 엔지니어가 이미 알고 있는 Simulink API를 활용하기 위해서는 Speedgoat의 버전 교체 또는 Speedgoat 자체의 교체가 요구된다. 하지만 Speedgoat이 고가의 장비이므로 해당 장비 버전의 변화는 매우 어렵다. 따라서 따라서 시뮬레이션 모델을 인식하고 계산된 데이터를 출력 및 시각화할 수 있는 도구가 필요하다. 시뮬레이션 도구를 활용하면 엔지니어가 Simulink API 및 Matlab 기능의 사용법을 익힐 필요 없이 시뮬레이션을 진행할 수 있어 작업 효율성 증가가 예상된다.

이에 따라, 본 논문에서는 Host PC에서 시뮬레이션을 수행할 수 있는 Simulink 기반 실시간 모니터링 및 로깅 도구를 제안한다. 제안하는 도구에 시뮬레이션 모델을 입력하고, 도구를 사용해 중장비의 설정값을 변경하고, 설정 변경에 따른 출력값을 실시간으로 시각화할뿐만 아니라 로깅을 할 수 있다. 도구는 총 5개의 기능을 구현한다. 먼저 Speedgoat와 Host PC 간의 연결을 구현하는 Network 기능과 연결된 Speedgoat에 Host PC에서 빌드한 Simulink Real-Time 모델을 삽입하기 위한 파일 시스템 입출력 기능이 있다. 다음으로 삽입한 모델 내의 입력 값을 설정하는 파라미터 설정 기능과 설정한 파라미터에 따른 출력을 확인할 수 있는 시각화 기능, 마지막으로 시각화한 데이터를 로그 파일로 확인할 수 있는 Logging 기능이 있다.

도구의 다섯 가지 기능에 대한 자세한 구현 사항을 2장에서 설명한다. 이후 3장에서 기능 검증 시나리오에 따라 각 기능이 정의와 구현 의도, 기대 결과에 맞게 정확히 동작을 수행하는지 테스트한 결과를 기술하

며, 마지막으로 4장에서 결론을 기술하며 마무리한다.

2. 기능 설계

본 논문이 제안하는 도구는 중장비 출력 시뮬레이션을 위해 필요한 5개의 기능으로 구성된다. 그림 2는 사용자가 중장비의 출력을 시각화하기 위해 도구를 실행했을 때의 동작에 대해서 순서도로 나타낸 것이다. 이 장에서는 해당 순서도의 흐름에 따라 기능을 순서대로 기술한다. 순서도에서 각 상황에 따라 필요한 기능은 탭 또는 UI의 최상단에 버튼 형태로 구현하여 사용자가 원하는 기능을 탭으로 이동하여 사용할 수 있다. 각 기능의 동작을 세부 기능 순서도에 따라 정의하였고, 그에 따라 요구 사항을 도출하여 구현하였다. 본 연구에서 실시간 시뮬레이션 및 테스트에 사용된 Speedgoat는 이후 논문에서 '시뮬레이터'로 언급되며, Speedgoat에서 불러와 시뮬레이션에 사용하는 Simulink Real-Time 모델은 '시뮬레이션 모델'으로 언급된다.

구현에는 앞서 기술한 바와 같이 Speadgoat 기기의 변화를 최소화하기 위해, 구버전 Speedgoat와의 호환이 가능한 MATLAB R2018b 버전의 Simulink Real-Time API를 사용한다[3]. 이에 따라, UI 및 코드 개발에는 해당 API와의 호환성을 제공하는 Microsoft .NET Framework를 사용한다[4].

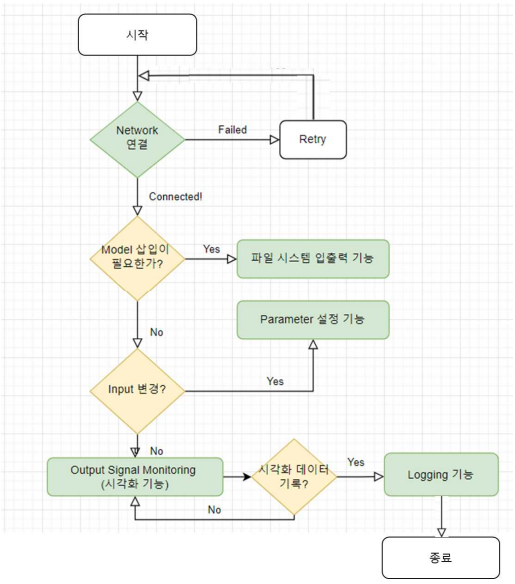

그림 2. 도구의 동작 순서도

Fig. 2. Tool operation flowchart

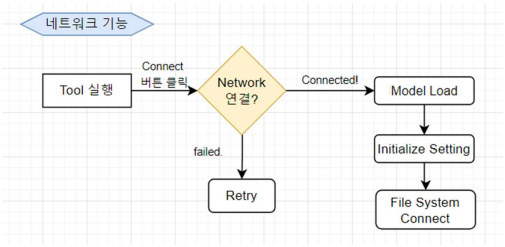

그림 3. 네트워크 기능의 요구사항

Fig. 3. Network function requirements flowchart

2.1 네트워크 기능

네트워크 기능은 Host PC와 시뮬레이터 간의 기본적인 통신을 구현한다. 그림 3은 도구의 네트워크 기능에 대한 요구 사항을 도출하기 위해 정의한 순서도이다.

사용자가 도구를 실행하면 먼저 시뮬레이터와의 연결을 수립해야 한다. Host PC와 시뮬레이터는 LAN 케이블을 통한 1대1 물리적인 네트워크 연결을 사용하기 때문에 Connect() API를 활용하여 Host PC에서 시뮬레이터와의 연결을 구축할 수 있다. 시뮬레이터는 기본적으로 고정 IP 및 포트 번호를 사용한다. 하지만 사용 환경에 따라 할당된 고정 IP 또는 포트 번호를 수정하여 사용할 수 있다. 이런 확장성을 고려하여 사용자가 직접 임의의 IP 주소를 입력하거나, 고정 IP 주소를 입력해 시뮬레이터를 특정하여 연결을 맺는다. 그림 4와 같이 도구의 상단에 위치한 'Connect' 버튼을 통해 연결을 시도할 수 있다.

연결에 성공한 이후의 기능이 동작하기 위해서는 시뮬레이터의 파일 시스템과 시뮬레이터 내의 시뮬레이션 모델을 불러올 필요가 있다. 따라서, 연결한 시뮬레이터의 파일 시스템에 접근하여, 모델을 불러온다. 이때 시뮬레이터가 불러온 모델에 대해 기존에 종료되지 않은 시뮬레이션이 있을 경

그림 4. 네트워크 기능 UI
Fig. 4. UI of network function

우 도구 내에서 시뮬레이션 설정을 변경하려고 할 때 오류가 발생할 수 있으므로, 시뮬레이터가 가진 모든 시뮬레이션 설정 및 실행을 초기화한다. 설정 초기화 작업이 모두 끝나면 시뮬레이터의 파일 시스템에 연결할 수 있다.

2.2 파일 시스템 입출력 기능

파일 시스템 입출력 기능은 사용자가 시뮬레이터에 새로운 시뮬레이션 모델을 삽입할 수 있는 기능이다. 그림 5는 파일 시스템 입출력 기능의 요구 사항을 도출하기 위해 정의한 순서도이다. 본 기능은 'File System' 탭에 구현한다.

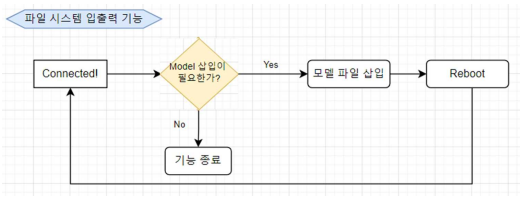

그림 5. 파일 시스템 입출력 기능 요구사항
Fig. 5. File system I/O function requirements flowchart

시뮬레이터로는 한 번에 하나의 모델만을 시뮬레이션할 수 있다. 만약 사용자가 연결한 시뮬레이터에서 현재 불러온 시뮬레이션 모델이 아닌 새로운 시뮬레이션 모델을 사용하여 시뮬레이션을 하길 원할 경우 새로운 모델을 삽입해야 한다. 따라서 Host PC에서 설계한 새로운 모델을 시뮬레이터에 삽입되어 있는 모델 위에 덮어쓸 수 있도록 구현한다. 사용자 편의를 위해 파일 복사 기능은 drag-drop 방식으로 구현하여 사용자가 직관적인 인터페이스를 통해 간단히 모델을 삽입할 수 있다. 추가로 편의성 제공을 위해 파일 관리에 사용하는 파일 복사, 삭제, 새로 고침 등 기본적인 파일 관리 기능을 구현하였다.

모델 삽입이 끝난 후에는 시뮬레이터를 재시작해야 새로운 시뮬레이션 모델을 인식할 수 있다. 따라서 도구의 상단 UI에 'Reboot' 버튼을 통해 시뮬레이터를 비동기적으로 재시작하도록 구현한다. 이 경우 Host PC와 시뮬레이터 간 재연결이 필요함을 메시지를 통해 안내하고, 비연결 상태로 모든 설정을 초기화한다.

2.3 파라미터 설정 기능

파라미터 설정 기능은 읽어 들인 시뮬레이션 모델 내의 입력에 해당하는 값들을 불러오고, 값을 변경할 수 있는 기능이다. 그림 6은 파라미터 설정 기능의 요구사항을 도출하기 위해 정의한 순서도이다. 본 기능은 'Parameter' 탭에 구현한다.

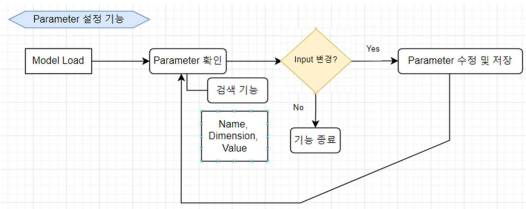

그림 6. 파라미터 설정 기능의 요구사항
Fig. 6. Parameter setting function requirements flowchart

먼저 모델에서 파라미터에 해당하는 모든 값을 불러온 파라미터의 목록을 UI의 좌측에 표시한다. 사용자가 원하는 파라미터를 찾기 쉽도록 알파벳 순으로 정렬하여 표시하며, 상단에는 사용자가 파

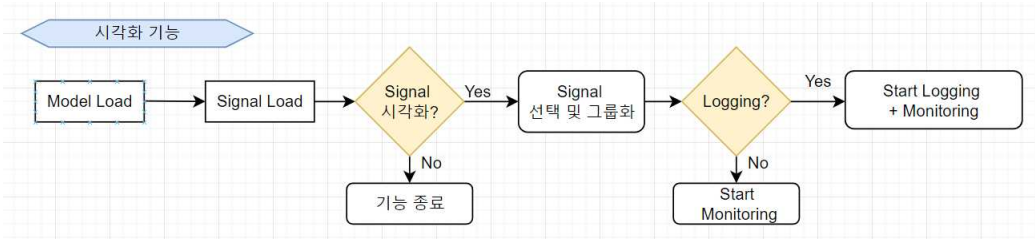

그림 7. 시각화 기능의 요구사항

Fig. 7. Visualiztion function requirements flowchart

라미터를 이름 기반으로 검색할 수 있도록 구현한다. 파라미터는 실제 중장비의 동작과 성능에 영향을 미치는 센서 데이터나 제어 명령값, 압력, 위치, 힘, 무게 등의 다양한 종류가 존재한다. 이에 따라 파라미터의 Dimension 값이 각각 다르다는 점을 유의하여 UI에 파라미터의 이름과 차원, 그에 따른 값을 명확하게 표시할 수 있도록 구현한다. 모델 내의 파라미터를 인식하고 이의 속성인 Dimension 정보를 문자열로 변환하여 이름과 함께 목록에 표시한다. 사용자가 파라미터 목록에서 파라미터를 클릭할 경우, UI의 우측에 해당 파라미터의 Dimension 정보에 따른 테이블을 생성하고 이에 따른 값을 불러와 표시한다.사용자가 확인한 파라미터의 값의 변경을 원할 경우 값을 클릭하여 즉시 원하는 값으로 변경할 수 있다. 사용자가 임의로 변경할 파라미터의 새 값이 해당 파라미터의 자료형이나 원칙에 어긋나지 않을 경우 반영된다. 변경이 완료된 파라미터 값은 즉시 시뮬레이터에 전달되어 모델에 반영되고, 이는 시각화하는 Signal 값에 바로 적용될 수 있다.

2.4 시각화 기능

시각화 기능은 앞서 변경한 입력 파라미터에 따른 출력을 실시간으로 모니터링하는 기능이다. 그림 7은 시각화 기능의 요구 사항을 도출하기 위해 정의한 순서도이다. 본 기능은 'Graph' 탭에 구현한다.

해당 기능에서 사용자는 출력할 하나 또는 다수의 Signal을 선택하여 시각화할 수 있다. 시각화 직전에 Logging을 활성화/비활성화할 수 있는데, 이는 로깅 기능이며 2.5장에서 설명한다. 차트 하나에 그릴 Signal의 묶음을 'Scope'라고 정의한다. 사용자는 Signal의 전체 리스트에서 Signal을 선택하고, 시각화할 Signal의 개수가 많을 경우 Scope 단위로 그룹화하여 Scope 설정에 따라 차트화된 데이터를 모니터링할 수 있다. Scope는 최대 8개까지 정의될 수 있어 즉, 한 번의 모니터링에 최대 64개의 Signal이 8개의 차트에 표현될 수 있도록 구현한다.

그림 8은 시각화 기능 탭의 UI 전체 모습이며, 프로그램 실행 화면을 캡쳐한 것이다. UI의 좌측에서 Scope를 설정하고, 우측에서는 설정한 Scope에 따라 차트를 표현한다. 좌측 UI에서는 모델에서 불러온 모든 Signal을 목록에 표시하며, 사용자는 Scope 설정 박스에 Signal을 Drag & Drop을 통해 추가할 수 있다. Scope 설정 박스는 앞서 기술한 바와 같이 8개까지 추가될 수 있으며, 버튼을 통해 삭제될 수 있다. Scope 설정이 끝나면 하단의 'Start Graphing' 버튼을 통해 모니터링을 시작한다. 만약 사용자에 의해 모니터링을 진행 중인 Signal에 영향을 주는 파라미터의 값이 변경될 경우, 이를 실시간으로 반영한다. 추가로, Scope의 개수가 4개 이하일 경우 표시할 차트의 수에 따라 동적으로 크기를 조정하도록 구현하고, Scope 설정 UI는 열고닫을 수 있도록 구현하여 모니터링 UI의 가시성이 높다.

Scope 설정에 따른 데이터의 시각화의 구현은 다음과 같이 이루어진다. 그림 9는 모니터링 시작 시점부터 데이터의 차트 시각화까지의 Host PC

와 시뮬레이터 간의 상호작용을 나타낸다. ‘Start Graphing’ 버튼이 눌리면 먼저 시뮬레이터의 Scope 설정을 초기화한다. 만약 모니터링을 새롭게 시작할 때 직전의 모니터링 설정(Scope의 개수 또는 Scope 안에 들어있는 Signal의 개수와 종류, Scope와 Signal의 라벨링)이 하나라도 남아있을 경우 주고받는 데이터에 오류가 매우 쉽게 발생한다. 따라서 모니터링 시작 시 모든 Scope에 관한 설정을 초기화하여 오류를 피한다. 이후 Host PC에서 사용자가 정의한 Scope 설정 전체를 시뮬레이터로 전송한다. UI의 Scope 박스 라벨에 따라 Scope를 생성하고, 박스 내에 담긴 Signal을 Scope에 추가하는 형식으로 구현한다.

Scope에 Signal을 추가하기 위한 Signal Access API는 Deprecation 문제가 발생하며, 이를 해결하기 위한 새로운 방식이 필요하다. Signal ID로 Signal 이름을 반환하는 API 내재 함수가 동작하지 않는 경우가 존재한다. 이는 개발 단계에서 시뮬레이터와의 호환성을 만족하기 위해 낮은 버전의 MATLAB 기반 API를 채택하면서 발생하는 문제이다. Signal의 ID는 이름과 1대1 대응으로 모델에 정의된다. 그러나 Signal 정보 접근 함수가 복잡한 블록 구조를 가진 Signal에는 사용이 불가하다. 따라서 해당 API를 사용하지 않고 Signal의 구조에 직접 접근하여 Signal에 대한 정보를 클래스 형태로 구조화하는 로직을 새롭게 구현하고, 해당 클래스를 이용해 Signal 정보를 저장 및 접근하도록 구현한다.

Scope 설정에 대한 모든 송수신이 완료되면 설정에서 추가된 모든 신호에 대해 시뮬레이션을 시작한다. 시뮬레이터는 Signal 데이터를 연산하여 출력 데이터를 Host PC에 전송한다. 이때, 실시간 데이터를 송수신하기 위해서는 기존 API의 데이터 송수신 로직이 아닌 새로운 로직이 필요하다. 기존 로직에서는 시뮬레이터에서 계산 완료한 데이터를 10초 단위로 묶어서 Host PC에 전송하는 방식을 채택하기 때문에, Real-Time 데이터 시각화에는 한계가 있다. Host PC에서 10초에 한 번 데이터를 수신해 모니터링 차트를 갱신하는 것은 실시간으로 데이터의 변화를 시각적으로 확인하기에 적합하지 않기 때문이다. 따라서 시뮬레이터에 실시간 반복 접근하여, 접근한 시점까지 쌓인 데이터를 가져오도록 구현한다. 이는 그림 9에서 표현한 바와 같이, Host PC에서 시뮬레이션에 반복적으로 개입하는 방식이다. 시뮬레이션을 시작시키고 0.01초 뒤 바로 중단시킴으로써, 시뮬레이터는 시뮬레이션을 진행하는 것보다 중단시킨 시점까지의 데이터를 송신하는 것을 우선으로 하게 된다. Host PC에서 데이터를 수신하면 다시 시뮬레이션을 시작시킨다. 새롭게 구현한 데이터 접근 방식으로 시각화 데이터는 0.01초 간격으로 쌓이며, Host PC에서는 10초

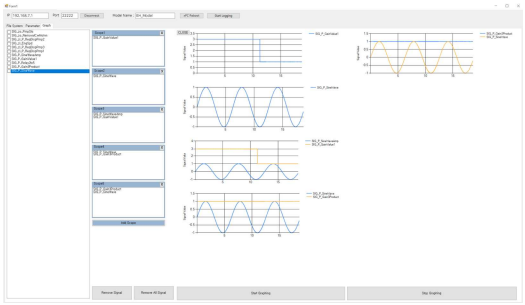

그림 8. 시각화 기능 UI
Fig. 8. UI of visualization function

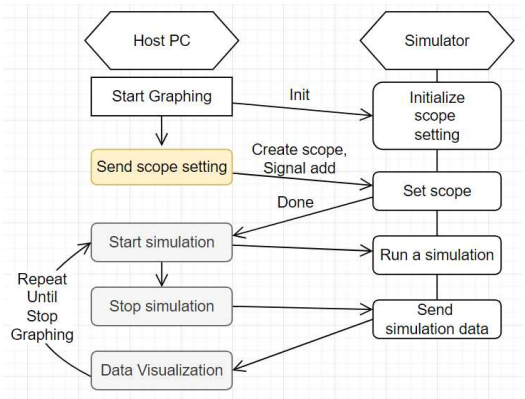

그림 9. Host PC와 시뮬레이터 간 송수신
Fig. 9. Transmission and Reception between Host PC and Simulator

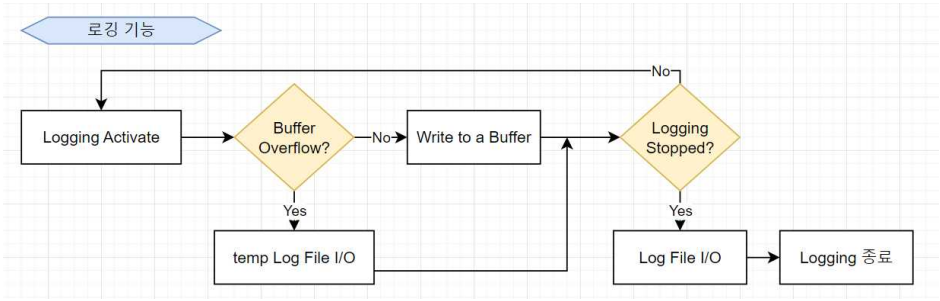

그림 10. 로깅 기능 구현 사항
Fig. 10. Logging function requirements flowchart

간격이 아닌 반복문을 통한 매 접근마다 시뮬레이터에 쌓인 데이터를 가져와 차트에 갱신할 수 있다.

2.5 로깅 기능

로깅 기능은 시각화하는 데이터를 로그 파일로도 저장할 수 있도록 구현한 기능이다. 2.4의 시각화 기능에서 Signal 데이터 시각화를 하는 시점에 Logging을 활성화 또는 비활성화할 수 있다. 사용자가 로깅을 활성화한 경우 시각화하는 모든 데이터를 로그 파일에 기록하는 것이다. 그림 9의 Host PC의 'Data Visualization' 작업에서 데이터의 로깅을 함께 진행한다. 데이터를 시각화하여 모니터링하는 것은 실시간으로 데이터의 변화를 확인하기에 유용하며, 로깅 기능을 통해 저장된 로그 파일을 통해 정확한 데이터 수치를 확인할 수 있다. 로그 파일은 Host PC에 엑셀 파일(.csv) 형태로 저장되며, 도구 상단의 버튼을 통해 로깅 기능을 활성화/비활성화할 수 있다.

그림 10은 로깅 기능의 요구사항을 도출하기 위해 정의한 순서도이다. 로깅 기능이 활성화된 시점부터 시각화하는 모든 데이터를 파일에 기록한다. 이때, 파일 읽기 및 쓰기 작업은 컴퓨터에서 속도가 매우 느린 작업에 속한다. 실시간으로 Host PC와 시뮬레이터 간 데이터를 송수신하고 시각화하는 도중 해당 작업까지 병행할 시 모니터링 기능에 속도 저하를 일으킬 수 있다. 따라서 임시 저장 버퍼를 활용한다. 로그 데이터를 시각화 도중에는 버퍼에 입력하고, 로깅이 끝나면 파일에 버퍼에 저장한 모든 데이터를 입력한다. 만약 모니터링 도중 임시 저장 버퍼가 가득 찰 경우, Host PC의 바탕화면에 임시 파일을 생성하여 데이터를 쓰고 저장함으로써 버퍼의 overflow 문제를 막을 수 있다. 이후 로깅을 멈추면 사용자에게 로그 파일을 저장할 경로와 파일 이름을 지정하도록 하고, 임시 파일에 저장한 데이터와 현재 버퍼에 있는 데이터를 합쳐서 사용자가 지정하는 경로 및 파일 이름으로 최종 로그 파일을 생성한다.

그림 11은 저장된 로그 파일의 모습이다. 로깅이 끝난 시점에 설정한 프로젝트명과 날짜 등의 정보와 함께 Signal의 ID, 이름, 출력 시점에 따른 데이터가 기록된다.

3. 검증

본 논문에서 제안하는 실시간 모니터링 및 로깅 도구의 각 기능과 UI가 구현 의도에 따라 예상된 동작을 수행하는지 확인하기 위하여 각 기능에 검증 시나리오를 세우고 테스트하였다. 검증 시나리오는 각 기능에서 가장 중요한 기능 및 UI를 중심으로 선정하였다. 이 장에서는 각 시나리오를 진행하여 검증되는 기술 및 검증 결과에 대해서 순서대로 기술한다. 로깅 기능의 수행 결과물인 로그 파일에 대해서는 2.5의 그림 11을 통해 확인할 수 있어 해당 기능의 검증 시나리오는

생략한다. 테스트는 Speedgoat와의 물리적 LAN 1:1 연결이 완료된 상태의 Window 10 운영체제 데스크톱 환경에서 수행하였다.

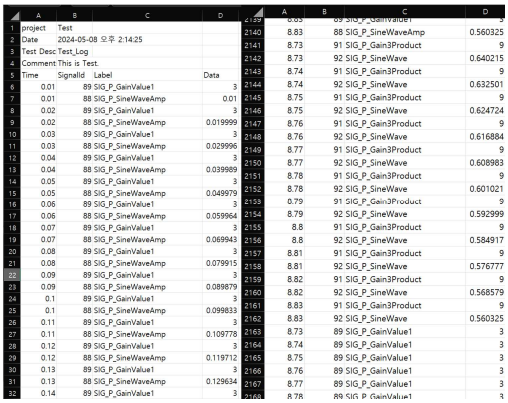

그림 11. 저장된 로그 파일
Fig. 11. Log File

3.1 네트워크 기능 검증

먼저, 네트워크 기능을 검증하기 위해 'Connect' 버튼을 누르는 시나리오 1을 수행하였다. 해당 시나리오를 통해 시뮬레이터와 Host PC 간 연결 기능이 검증될 수 있다. 해당 기능이 성공적으로 수행되면 'Connect' 버튼은 'Disconnect' 버튼이 된다. 그림 12는 프로그램 실행 화면을 캡쳐한 것이다. 버튼을 누르자 그림 12와 같이 예상한 결과대로 버튼이 올바르게 변경되었으며, 시뮬레이터에 내제된 모델의 이름을 읽어들여 UI에 표시하는 것을 확인하였다.

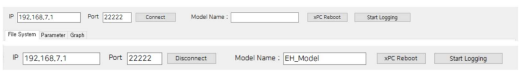

그림 12. 시나리오 1 검증 결과
Fig. 12. Scenario 1 Verification Results

3.2 파일 시스템 입출력 기능 검증

파일 시스템 입출력 기능 검증을 위한 두 가지 시나리오를 수행하였고, 검증 결과를 확인하였다. 시나리오 2-1은 네트워크 연결이 완료된 후 파일 시스템 탭에 접근했을 때 Host PC와 시뮬레이터의 C 드라이브를 UI에 올바르게 띄우는지 확인한다. 이는 시뮬레이터에 모델 파일을 복사하기 위해 필요한 핵심 UI에 대한 검증이다. 해당 시나리오에 따라 파일 시스템 탭에 접근한 결과 그림 13과 같이 좌측에 Host PC의 C 드라이브, 우측에 시뮬레이터의 C 드라이브가 UI에 올바르게 나타나는 것을 확인하였다.

시나리오 2-2는 Drag & Drop을 통해 Host PC에서 시뮬레이터로 모델 파일을 복사하는 해당 탭의 핵심 기능을 검증한다. Host PC의 파일 시스템에 있는 모델 파일인 'EH_Model.mldatx'를 도구 내의 시뮬레이터 파일 시스템 UI에 Drag & Drop을 하여 결과를 확인한다. 그림 14는 시나리오 2-2에 대한 결과이다. 파일이 올바르게 복사되었으며 시뮬레이터 파일 시스템 UI에서 복사된 모델 파일을 확인하였다.

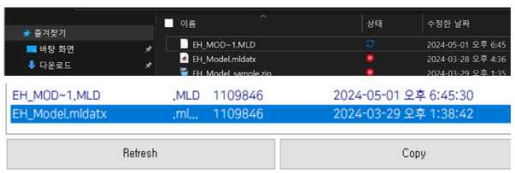

그림 14. 시나리오 2-2 검증 결과
Fig. 14. Scenario 2-2 Verification Results

3.3 파라미터 설정 기능 검증

파라미터 설정 기능 검증을 위한 두 가지 시나리오가 있다. 첫 번째 시나리오는 파라미터 설정 탭에 접근했을 때 전체 UI와 해당 UI와 사용자 간 상호작용을 확인한다. 사용자가 해당 탭에 접근했을 때 모델 내의 모든 파라미터가 파라미터 목록 UI에 로드되고, 파라미터를 클릭할 시 파라미터의 세부 정보가 나타나야 한다. 또한 상단에 위치한 파라미터 검색 기능 또한 올바르게 동작해야 한다. 시나리오 3-1은 파라미터 탭에 접근하고, 검색 UI에 'int'를 입력하여 나타나는 파라

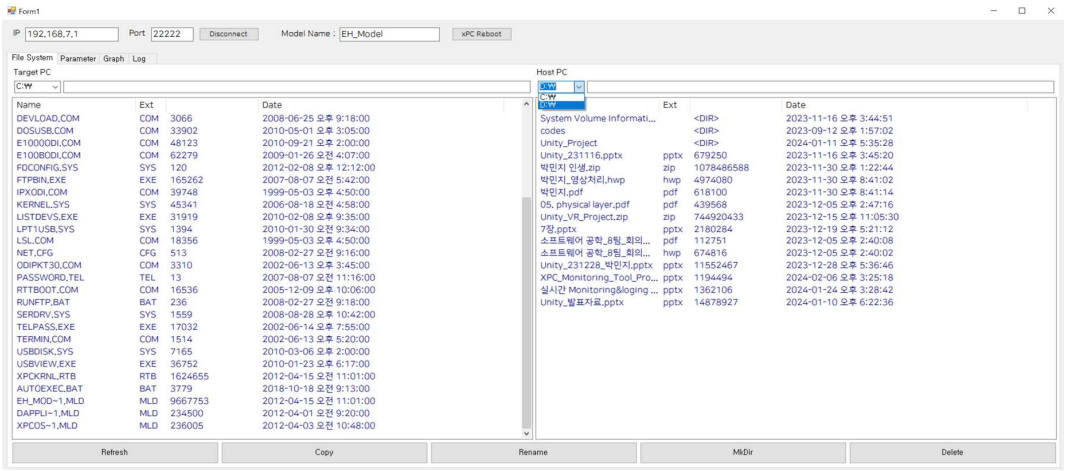

그림 13. 시나리오 2-1 검증 결과
Fig. 13. Scenario 2-1 Verification Results

미터 중 하나를 선택하여 세부 정보를 확인한다. 그림 15는 해당 시나리오를 통한 파라미터 설정 UI 검증 결과이다. 이름에 'int'를 포함한 파라미터를 검색하였고 이를 클릭해 우측에서 해당 파라미터가 가진 Dimension 1x3에 맞는 테이블과 값을 확인하였다.

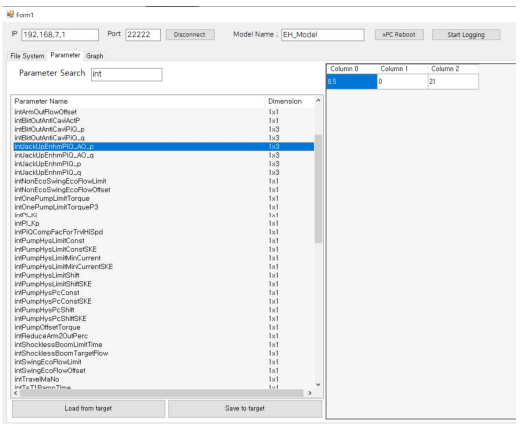

그림 15. 시나리오 3-1 검증 결과
Fig. 15. Scenario 3-1 Verification Results

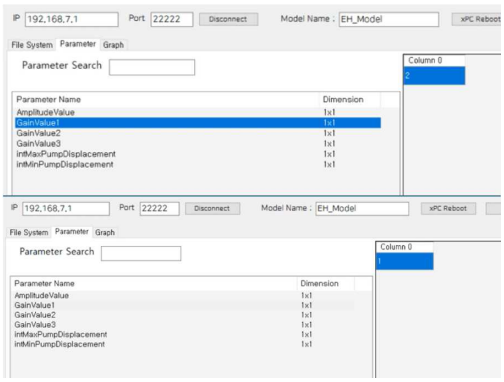

그림 16. 시나리오 3-2 : 파라미터 값 변경
Fig. 16. Scenario 3-2 : parameter setting

시나리오 3-2는 파라미터 설정 탭에서 우측의 세부 정보에서 확인한 파라미터의 값을 임의로 변경하고, 모델에 올바르게 반영되는지를 확인한다. 이는 파라미터 변경 기능이 올바르게 동작하는지를 검증하기 위한 시나리오이다. 시나리오에 사용한 Signal 'Sig_P_GainValue1'은 파라미터 'GainValue1'의 값을 그대로 출력한다. 따라서 그림 16과 같이 'GainValue1'의 값을 2에서 1로 변경하여 'Sig_P_GainValue1'의 값을 시뮬레이션하여 결과를 확인하였다. 시뮬레이션 실행 결과는 그림 17에서 나타낸다. 그림 17과 같이, 해당 Signal은 파라미터값 '2'에 따라 '2'를 출력하다가 파라미터값의 변경이 적용된 후 '1'이라는

출력값을 나타내는 것을 확인하였다. 시나리오 3-2를 통해 파라미터 변경 기능과 모니터링 도중 실시간 파라미터 변경 반영 기능을 검증 완료하였다.

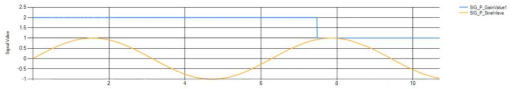

그림 17. 시나리오 3-2 검증 결과
Fig. 17. Scenario 3-2 Verification Results

3.4 시각화 기능 검증

다음으로, 시각화 기능 검증을 위하여 두 가지 시나리오를 수행하였다. 두 가지 시나리오를 통해 Signal을 그룹화하여 시각화하는 시각화 기능 전체에 대한 기능과 시각화 Chart의 동적 생성 기능을 검증한다.

첫 번째 시나리오는 Signal을 5개의 그룹에 임의로 Drag & Drop을 통해 그룹화하고, 'Start Graphing' 버튼을 통해 모니터링을 시작한다. 해당 시나리오를 통해 Host PC와 시뮬레이터 간 실시간 Scope 그룹 정보와 시각화 데이터 송수신 기능, Host PC에서의 실시간 데이터 처리 및 시각화 기능을 검증할 수 있다. 그림 18은 해당 시나리오를 수행한 결과이며, 프로그램 실행 화면을 캡쳐한 것이다. 'Scope 3(세 번째 그룹)'에 묶인 Signal 두 가지는 세 번째 차트에 함께 시각화되는 것을 확인하였다.

다음으로, 시나리오 4-2에서는 시나리오 4-1보다 적은 개수의 Chart를 생성함으로써 Chart의 동적 크기 조절 기능에 대해 검증한다. Scope 그룹을 3개로 설정하여 시각화할 Chart의 개수를 줄여 결과를 확인한다. 그림 19는 시나리오 4-2의 수행 결과이다. 그림 18에서는 2열으로 5개의 Chart를 생성하였지만 해당 시나리오에서는 차트를 1열으로 생성하자, 차트의 크기를 동적으로 조절하는 것을 확인하였다.

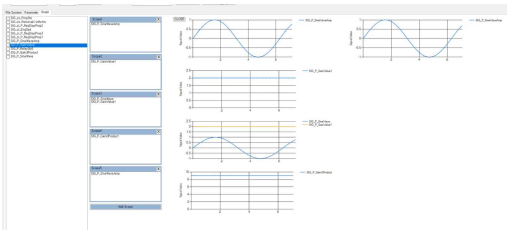

그림 18. 시나리오 4-1 검증 결과
Fig. 18. Scenario 4-1 Verification Results

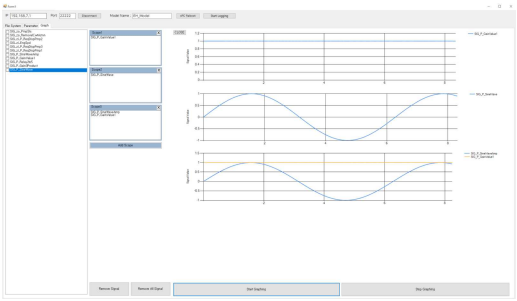

그림 19. 시나리오 4-2 검증 결과
Fig. 19. Scenario 4-2 Verification Results

마지막으로 시나리오 4-3에서는 결과값의 출력 간격이 동일하지 않은 chart를 생성하였다. 그림 20은 수행 결과를 나타낸다. 결과에서 확인할 수 있듯이 x축 간격이 동일하지 않은 것을 확인하였다. 이는 각 Chart 객체는 개별 Thread에 의해서 생성되고 제어되기 때문에 x축 간격이 미세하게 다른 것을 확인하였다. 이를 해결하기 위해서는 Thread 간 동기화가 필요하지만 이로 인해 실시간 시각화 성능이 제한될 수 있다.

4. 결론

본 연구는 Simulink Real-Time API를 이용해 Speedgoat와의 통신을 기반으로 Host PC에서 Simulink Real-Time 모델의 파라미터를 설정하고, 그에 따른 출력 Signal을 모니터링/로깅 할 수 있는 도구를 제안한다. 제안하는 도구의 기능들은 Speedgoat와의 연결을 지원하는 네트워크 기능, 시뮬레이션 모델을 삽입할 수 있는 파일 시스템 입출력 기능, 시각화 데이

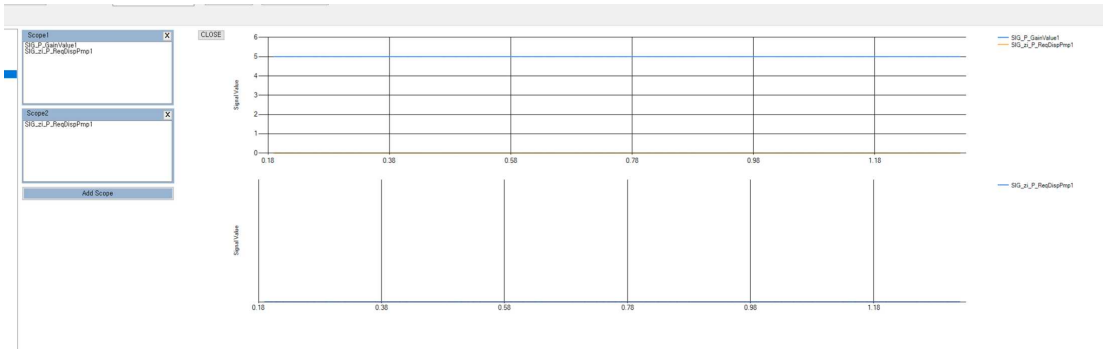

그림 20. 시나리오 4-3 검증 결과
Fig. 20. Scenario 4-3 Verification Results

터에 영향을 주는 파라미터를 확인 및 변경하는 파라미터 기능, 출력값을 모니터링하는 시각화 기능, 시각화 데이터를 파일에 저장하는 로깅 기능이 있다. 개별적인 기능 검증 시나리오를 기반으로 본 도구에서 제안하는 기능들의 동작을 검증하였다. 각 검증 결과 제안하는 도구의 기능들이 오류 없이 동작함을 확인하였다.

REFERENCES

[1] Speedgoat, https://www.speedgoat.com
[2] Simulink Real-Time, https://kr.mathworks.c om/products/simulink-real-time.html
[3] https://kr.mathworks.com/help/releases/R201 9a/pdf_doc/xpc/xpc_target_api_ugref.pdf
[4] https://dotnet.microsoft.com/ko-kr/
[5] CHO WOOSUNG, Cha Jihyoung, Sangho Ko, “Development of MATLAB/Simulink Modular Simulation Toolbox for Space Shuttle Main E ngine”, Journal of the Korean Society of Pro pulsion Engineers, August, 2019.
[6] In-Woo Lee, Soo-Yeong Yi, “Simulation of S ensorless BLDC Motors and a Two-axis Gimb al Using MATLAB/Simulink”, Journal of Instit ute of Control, Robotics and Systems, July, 2 024.
[7] Seung-Chan Yang, Tae-Seong Chae, Ju-Seon g Nam, Sun-Yong Kim, “Construction of Wh eel Loader Simulation Environment Using Si mulink”, Journal of Korea Academia-Industria l cooperation Society, December, 2023.
[8] Min-Woo Ham, Insu Paek, “Design and contr ol performance validation of HILS system bas ed on MATLAB/Simulink”, Journal of Wind E nergy, March, 2024.
[9] Bohwa Lee, Poomin Park, Chuntaek Kim, Su ngyug Kim, Sooseok Yang, “A Electric Power Source Modeling and Simulation for Electric Propulsion Systems of a Fuel Cell Powered S mall UAV”, Journal of The Korean Society fo r Aeronautical and Space Sciences, October, 2011.
[10] Carlos Villegas, “Hardware-in-the-Loop Test ing of Control Algorithms for Modular Multi-Level Converters”, ICPE 2019-ECCE Asia, Ma y, 2019.

저자약력

박 민 지 (Minji Park) **[학생회원]**

- 2021년 3월 ~ 현재: 창원대학교 컴퓨터공학과

〈관심분야〉 네트워크 기술, 안드로이드, 소프트웨어

홍 윤 빈 (Yoobin Hong) **[학생회원]**

- 2021년 3월 ~ 현재: 창원대학교 컴퓨터공학과

〈관심분야〉 네트워크 기술, 안드로이드, 소프트웨어

안 동 혁 (Donghyeok An) **[정회원]**

- 2006년 2월: 한동대학교 전산전자공학부 학사
- 2013년 2월: KAIST 전산학과 박사
- 2015년 3월 ~ 2017년 8월: 계명대학교 컴퓨터공학과
- 2017년 9월~현재: 창원대학교 컴퓨터공학과

〈관심분야〉 5G 및 6G, 저지연 통신, 딥러닝