Automatic Generation of Metamorphic Relations for a Cyber-Physical System-of-Systems Using Genetic Algorithm

Esther Cho, Yong-Jun Shin, Sangwon Hyun, Hansu Kim, Doo-Hwan Bae
School of Computing
Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, Republic of Korea
{esthercho, yjshin, swhyun, hansukim, bae}@se.kaist.ac.kr

***Abstract*—A Cyber-Physical System-of-Systems (CPSoS) has innate uncertainties from operation in the physical environment and interaction among the constituent systems. These uncertainties make a CPSoS more susceptible to the oracle problem, a challenge in determining the correct behavior when testing the system. Metamorphic testing (MT) suggests a solution to addressing this challenge by utilizing metamorphic relations (MRs), relations among multiple inputs and corresponding outputs of the system. However, when applying MT on a CPSoS, generating MRs is difficult due to the continuous operation of a CPSoS in uncertain environment. In this study, we propose a method to automatically generate MRs from field operational test (FOT) data logs of a CPSoS. We define an MR template to capture the CPSoS behaviors. We then apply genetic algorithm to adapt the MR generated by the engineers, and thus improve the testing effectiveness. Our method is validated in a case study of an autonomous robot vehicle. Our results show that the automatically generated MRs capture the behaviors of a CPSoS more realistically than the manually generated MRs. With our method, engineers can obtain CPSoS MRs with minimal manual effort.**

***Index Terms*—metamorphic testing, metamorphic relations, cyber-physical system-of-systems, genetic algorithm**

I. Introduction

Cyber-physical system-of-systems (CPSoS) is an interconnected system of cyber-physical systems (CPSs) that compute and communicate in physical elements [1]–[3]. A CPSoS, therefore, has innate uncertainties from continuous operation in physical environment [4]. In addition, a CPSoS embodies emergent behaviors of a system-of-systems (SoS) from interacting constituent systems [5], [6]. These uncertainties and emergent behaviors not only hinder CPSoS goal achievements, but also restrict the testing process to improve the performance. In particular, a CPSoS is more susceptible to the oracle problem, a challenge in determining the correct behavior when testing the system [7], [8].

Metamorphic testing (MT) [9]–[13] suggests a solution to resolving this challenge by utilizing metamorphic relations (MRs), relations among multiple inputs and corresponding outputs of the system. Rather than specifying the test oracle for individual test inputs to verify the system-under-test (SUT), MT reveals faults in the system by checking for violations of MRs. Although MT alleviates the oracle problem, defining MRs to substitute the test oracles still requires domain knowledge and manual efforts. Several studies have applied MT on SoS and CPS to alleviate the oracle problem and subsequent challenges, such as defining MRs [14]–[17]. However, MR generation for MT on CPSoS, which encompass both SoS and CPS, has not been considered to the best of our knowledge. In addition, a CPSoS has continuous interactions with the environment, and thus generation of MRs should also consider the continuous data.

In this paper, we propose a search-based method to generate MRs from the initial MR and field experiment data. MR is specified in a data-driven format to leverage the field experiment data uncovering the uncertain behaviors and continuous operation of the CPSoS. Then, using engineers' basic understanding of the CPSoS, initial MR is specified. The manually generated MR is then used to automatically search for MRs using a genetic algorithm (GA). By utilizing the initial MR and GA, we limit the search space and cost of generating meaningful MRs. Finally, we evaluated our approach on a case study of a modeled autonomous vehicle [18]. The results show that the proposed approach is able to generate numerous relevant MRs. Further analysis reveals that the generated MRs better capture the real system properties, exposed by the frequency of cases that hold the MR. In summary, the main contributions of our approach are as follows:

- We propose a novel framework for MT on CPSoS.
- We propose a data-driven template of MR of a CPSoS.
- We provide a search method to generate MRs from field experiment data and initial MR.

The remainder of the paper is organized as follows. In Section II, background information on MR generation approaches and GA is introduced. In Section III, the overall approach and the modules comprising the proposed method are introduced, which are applied and assessed in a case study in Section IV. Section V discusses an additional use case of our method and future work of our study. In Section VI, threats to validity in this study are addressed. Section VII scrutinizes related work. Finally, Section VIII concludes the paper.

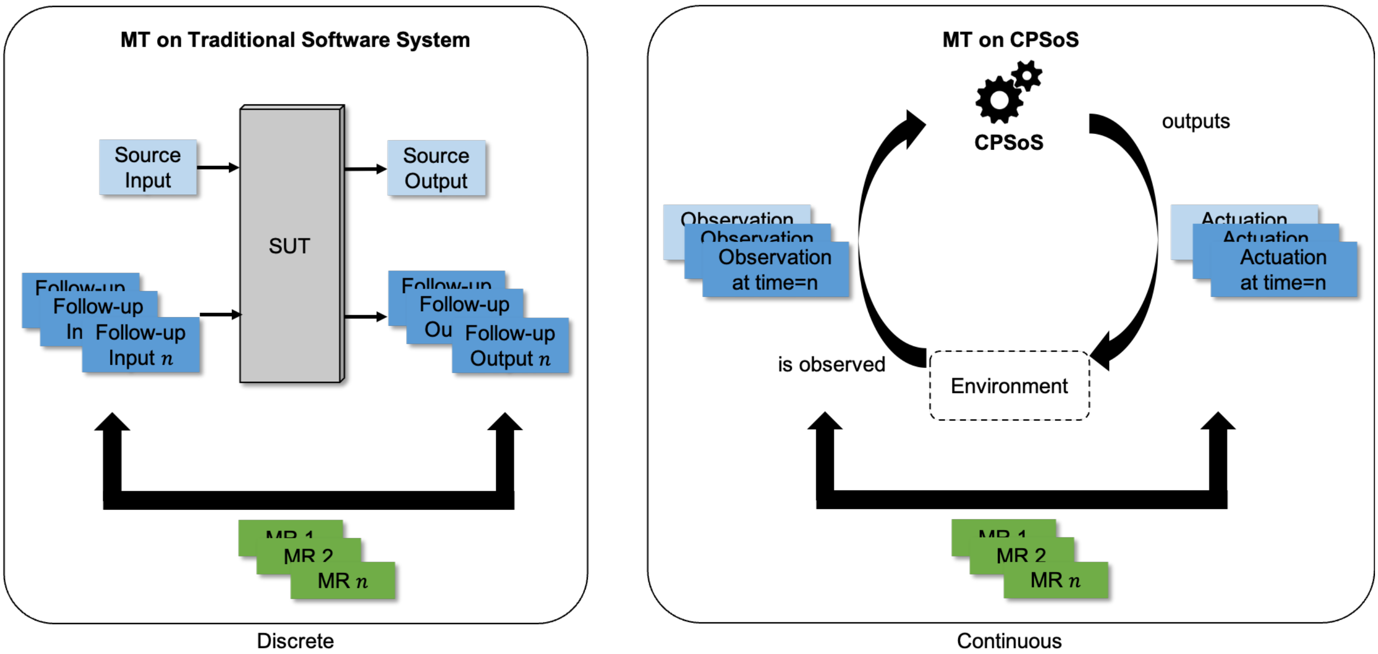

Fig. 1. Comparison of MT on Traditional System and CPSoS

II. Background

Metamorphic testing on traditional software systems vs CPSoS. Since the inception of MT [9], MT has been applied in various fields, such as software testing to reveal faults [19], quality assessment [20], and performance testing [21]. In such cases, the traditional systems utilized discrete test cases with source input and its corresponding output. Then, transformation on the source input generates follow-up inputs, which are used to execute the SUT to obtain follow-up outputs. The necessary properties across these source and follow-up test cases are used to define MRs. Violation of MRs reveals faults in SUT.

Unlike traditional systems, which has discrete test cases, CPSoS has continuous data. In this study, the loop of observation and actuation of CPSoS is considered the inputs and outputs of CPSoS, and thus continuous test cases. Comparison of applying MT on traditional software systems and on CPSoS is shown in Fig. 1. Based on the observation of the environment (input), the CPSoS reacts accordingly (output). On the other hand, the CPSoS action (input) may affect its observation of the environment (output). This continuous interaction between the CPSoS and its environment calls for consideration of temporal dependency of observation and actuation values at various points in time.

To apply MT, MR identification is a crucial, yet challenging step. Because MR identification often requires domain knowledge and manual effort, it is also considered the most expensive step in MT. To alleviate the manual burden, various studies [22], [23] have proposed MR identification approaches. In [13], MR identification is categorized into two approaches: 1) input driven and 2) output driven. Input-driven MR identification proposes changes to the inputs, and thus the expected output. Based on these changes, MRs are identified by pondering how the changes to the source test cases would generate the follow-up test cases. On the other hand, output-driven approach discovers the relations among the outputs and proposes changes to the inputs that result in the discovered output relations. The proposed approach in this paper can be categorized into output-driven approach as the field experiment data is used as a source of MR identification.

Genetic algorithm. GA is a meta-heuristic search method inspired by evolutionary biology. This evolutionary algorithm uses evolutionary pressure, defined by a fitness function, to search for solutions in a large search space. A solution is defined by its representation, which captures the genetic materials. A population of solution candidates, or individuals, is evolved through generations through exploitation and exploration, using genetic operators. Exploitation retains the genetic materials of “fitter individuals.” Exploration introduces diversity to the population. In this paper, GA is used to generate MRs by navigating the search space defined by the field experiment data.

Field operational test. Analysis of uncertainty in CPSoS is necessary to better comprehend and test the system operation [24]–[26]. One of the analysis methods is a field operational test (FOT) [27], which allows an empirical analysis from the system execution in a real environment. The field test result captures the real system behaviors and environmental conditions, and thus reflects the uncertain nature of the CPSoS operation. Therefore, in this study, FOT logs are utilized to capture uncertainties and reflect the environment of real system operation into the testing boundary.

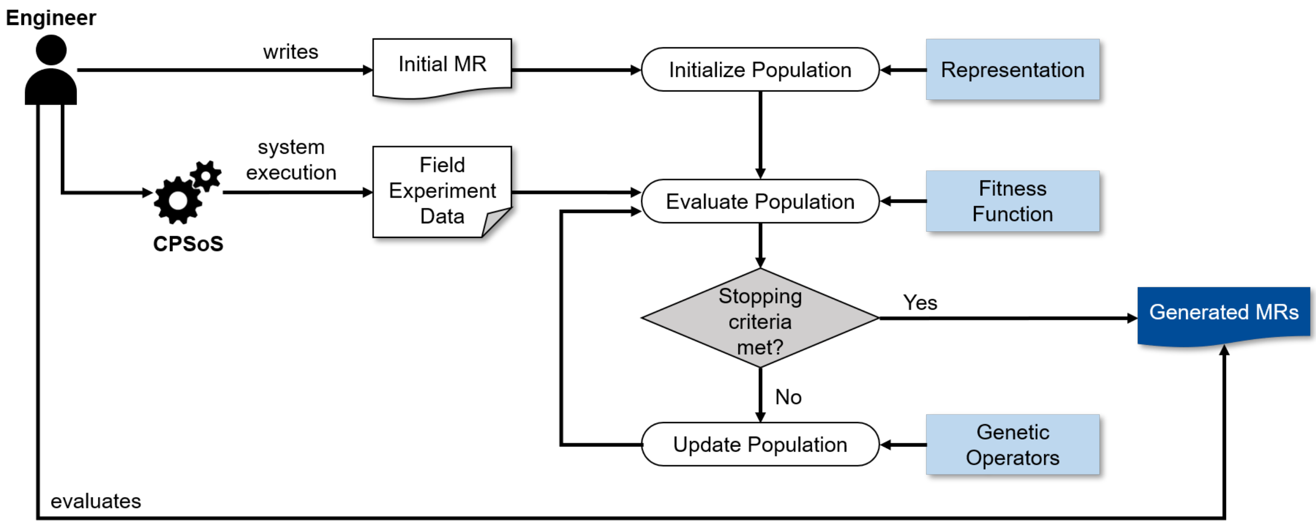

Fig. 2. Generation of Metamorphic Relations

III. Approach

The overall approach of automatic generation of metamorphic relations is demonstrated in Fig. 2. The engineers can write the initial MR based on their understanding and knowledge of the CPSoS. They can obtain the field experiment data by executing the CPSoS. First, the initial seed population is generated by using the proposed MR template as a representation of an MR and modifying the manually written initial MR. Then, the fitness function is used to evaluate the population against the field experiment data and select parents to evolve. To update the population, selected parents generate children using the genetic operators. The generation, evaluation, and selection of the offspring population is repeated until the stopping criteria is met, where the generated MRs are finally delivered. The engineers review the generated MRs to determine the relevance of the properties discovered.

The MR representation and the detailed steps of generating the seed population from the formally specified initial MR are described Section III-A. Section III-B presents the genetic operators to update the population of MRs. Lastly, the fitness function to evaluate the population is described in Section III-C.

A. Representation of Metamorphic Relations to Generate Seed Population

With domain knowledge and system understanding, engineers can write the initial MR. The manually written MRs reflect the engineer's preconceived notion of the necessary properties of the CPSoS, the environment, and their interactions. By formally defining the representation of MRs, they can be expressed in a standardized way, allowing systematic analysis against the CPSoS. An MR and its constituting components are defined in Definition 3.1. An MR is defined as a causal relationship between two propositions. Such cause-and-effect relationship illustrates the continuous operation of a CPSoS.

Definition 3.1 (Metamorphic relations):

$$
\begin{array}{rrcl}
\text{index} & i, j & \in & \mathbb{Z} \\
\text{time} & t & \in & \mathbb{Z} \\
\text{Variable of index } i \text{ at time } t & v_i^t & \in & \mathbb{V} \\
\text{Constant of index } j & c_j & \in & \mathbb{V} \\
\text{Value} & v & \in & \mathbb{V} \\
\text{Proposition} & p & ::= & (v < v) \\
 & & | & (v \leq v) \\
 & & | & (v > v) \\
 & & | & (v \geq v) \\
 & & | & (v == v) \\
 & & | & (v \neq v) \\
 & & | & p \cup p \\
 & & | & p \cap p \\
 & & | & !(p) \\
\text{Metamorphic Relations} & mr & ::= & (p, p)
\end{array}
$$

Variables, v_i^t, embody time-series data, such as the sensor readings obtained from monitoring the environment. Consequently, the relative time tick, t, is used to compare the change in value between two points in time. The index, i, identifies which data is being called from the collected data. Constants, c_j, are invariant values used in the system, such as the threshold values. Since the system may contain numerous constants, index j is used to identify which one is being called. Unlike the variables, a constant value is timeless. Therefore, the time component is not included in the definition of a constant. A value, v, is an element from a set of variables and constants defined in the system. Therefore, the variables, constants, and values are in the value domain $\mathbb{V}$.

A proposition, p, is defined as a relationship between the values or the propositions. The relationship between the values

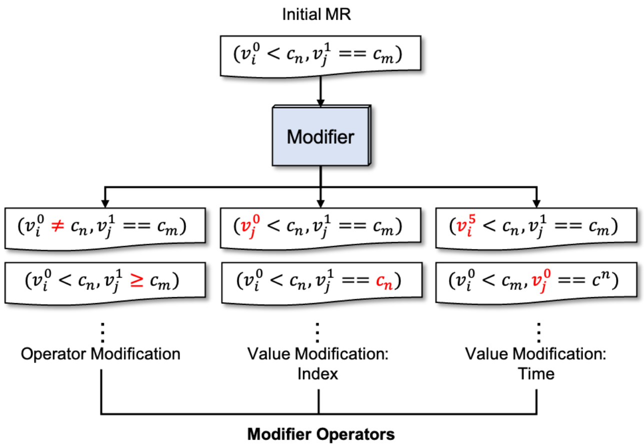

Fig. 3. Example Modification of initial MR to Generate Seed Population

is restricted to the following operators: $<$, $\leq$, $>$, $\geq$, $==$, and $\neq$. A proposition consisting of two values and a relationship operator is defined as a unit proposition. A proposition consisting of propositions is defined as a composite proposition. The relationship between the propositions is restricted to the negation (!), AND ($\cup$), and OR ($\cap$). An MR, mr, is defined as two propositions separated by a comma (,), which indicates a causal relationship between the two propositions, such as "if" and "then" statements. For example, an initial MR may be defined as $mr = (v_0^0 < c_0, v_1^1 == c_2)$ in the formal notation, which indicates that if variable 0 at time 0 is less than constant 0, then variable 1 at the next time tick is equal to constant 2. The values referenced in an MR are specified in the data.

Using the proposed MR representation, engineers can express their knowledge on the necessary properties of the CPSoS operation as initial MR. The seed population of MRs is generated by slightly modifying the initial MR. The modifications are categorized as 1) operator modification and 2) value modification as shown in Fig. 3. The value modification is further categorized into index modification and time modification.

The operator modification refers to modifying the operators of the proposition p. The unit propositions can be modified by changing the unit proposition operators. For example, proposition $(v < v)$ can be modified to $(v \leq v)$, $(v > v)$, $(v \geq v)$, $(v == v)$, or $(v \neq v)$. The composite propositions can be modified by changing the composite proposition operators. For example, a proposition $p \cap p$ can be modified to $p \cup p$ or $!(p \cap p)$.

The value modification refers to changing the values, v, and can be classified into the modification of the value index and the modification of the time. In a unit proposition, the value v can either be a variable v_i^t or a constant c_j. Consequently, a variable can be modified by changing either the index or time. On the other hand, a constant can be modified by changing its index since a constant is a timeless value.

The expected number of initial seed population can be calculated by Equation (1).

$$\begin{aligned} Seed\ Pop\ Size \ = & \ (\#\ of\ Op.\ Mod. - 1) * 2 + \\ & (Time\ Range - 1) * V_{num} + \\ & (Variable\ Range - 1) * V_{num} + \\ & (Constant\ Range - 1) * C_{num} \end{aligned} \tag{1}$$

V_{num} and C_{num} are the number of variables and constants specified in the initial MR. The time range is the total number of ticks available in the data. It is multiplied by the number of variables as the constants do not have a time component. The sum of the number of MRs generated from the variable range and constant range is the number of MRs for "Value Modification: Index", whereas the number of MRs from the time range is "Value Modification: Time."

B. Genetic Operators for Updating Metamorphic Relations

The step to update the population aims to exploit and explore the search space of metamorphic relations using crossover and mutation, respectively. A metamorphic relation is a causal relationship of two propositions, consisting of values and their relationship as defined above. Consequently, we designed the genetic operators to update the population of metamorphic relations, taking their hierarchy into account.

The crossover operator exploits the searched space by swapping the genetic materials between two survived parent to populate two offspring. For the exchange, the operator can either 1) swap at the proposition-level or 2) at the value-operator level. Proposition-level crossover has two options, an option to switch the "if" proposition or the "then" proposition. On the other hand, the value-operator-level crossover can exchange at the component-level of a proposition by swapping either the values or the operator. The level at which the crossover occurs and the occurrence of the crossover are decided by the probabilities the engineer specify.

The mutation operator explores the search space of metamorphic relations by introducing (potentially) new genetic materials using random generation. Similar to the crossover operator, the mutation can occur at 1) the proposition-level or 2) the value-operator level. Proposition-level mutation generates an entirely new proposition and replaces the existing proposition in a metamorphic relation. Value-operator-level mutation creates a new value or operator to replace the corresponding component inside a proposition. Engineers can specify the probabilities of the level at which the mutation occurs and the occurrence of the mutation.

C. Evaluating Metamorphic Relations Using Fitness Function

The population of MRs is evaluated against the field experiment data, which provides objective information on how the MRs hold in real operation. The result of evaluation guides the search for MRs that are more prominent by selecting the next generation that achieves better fitness, according to the fitness function as shown in Equation (2).

$$score = \frac{tp}{tp + tn} * Pen_{time} * Pen_{cohesion} \tag{2}$$

Here, tp or true-positive is defined as the number of cases where both the first and second propositions are true, indicating that the MR is satisfied. On the other hand, tn or true-negative is defined as the number of cases where the first proposition is true, but the second proposition is false, implying the cases where the property of MR-of-interest was not held. The ratio, $\frac{tp}{tp+tn}$, is a quantitative measure of MR property satisfaction. The ratio that is closer to a 100% implies the generated MR is more observed, and thus relevant to the real operation. The false positive, fp, and false negative, fn, are the number of cases where the first proposition is false. Since we are interested in the cases where the first proposition is true, thus satisfying the "if" property, the fp and fn cases are not considered in the score.

To further guide the search for meaningful MRs, time and cohesion penalties are introduced. Time penalty, Pen_{time}, penalizes MRs that exceed a threshold time range, since the causal relationship found may be uncorroborated due to the significant time gap. Equation (3) shows how Pen_{time} is computed:

$$Pen_{time} = \begin{cases} 1 - p_t, & \text{if time greater than max threshold} \\ 1, & \text{otherwise,} \end{cases} \tag{3}$$

where p_t is a time penalty coefficient, which is a value between 0 and 1.

Cohesion penalty, $Pen_{cohesion}$, penalizes MRs that contain proposition(s) with different value types. Since a proposition is a relationship between the values or the propositions, the comparison of different value types is meaningless. Propositions containing different value types are thus penalized. Equation (4) shows how $Pen_{cohesion}$ is computed:

$$Pen_{cohesion} = \begin{cases} 1 - (p_{ch} * i), & \text{if different value types} \\ 1, & \text{otherwise,} \end{cases} \tag{4}$$

where p_{ch} is a cohesion penalty step size, which is a value between 0 and 1. i is the number of search iterations, which is multiplied to p_{ch} to retain diversity in the population by preserving different value types early on in the search process.

Once the MRs are generated, their scores are compared to the score of the initial MR. MRs that yield equal or higher scores than the initial MR are further evaluated by the domain experts to reveal meaningful properties of the CPSoS that account for uncertainties.

IV. Evaluation

To evaluate the proposed approach, we implemented the proposed approach, which can be found in our repository [1]. Research questions and experiment setup are described in the following sections. The results from the experiment are analyzed.

[1]https://github.com/est-cho/MRGenerator

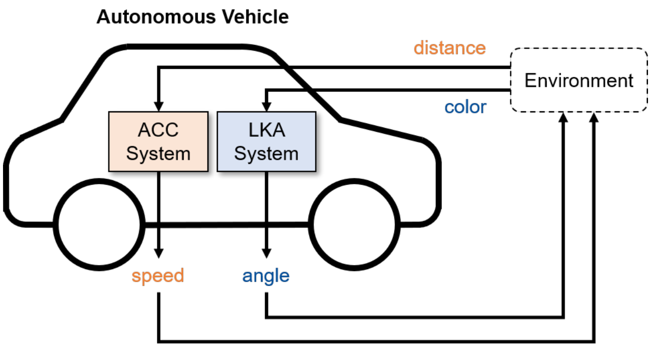

Fig. 4. Case Study: Modeled Autonomous Robot Vehicle

A. Research Questions

We designed the experiment with aims to answer the three research questions:

RQ1 What is the cost of generating MR?
RQ2 Is the proposed approach effective at generating MRs?
RQ3 How do generated MRs compare with other methods?

The cost of generating MR (RQ1) is assessed by the ratio of meaningful MRs from a list of MRs. The effectiveness of generating MRs (RQ2) analyzes the number of cases the metamorphic relations hold, such as the tp and tn numbers. RQ3 evaluates how the proposed approach perform compare to other methods, such as the baseline of manual generation of metamorphic relations.

B. Case Study Subject: Autonomous Robot Vehicle

We implement an autonomous robot vehicle equipped with both a lane-keeping assistant (LKA) system and an adaptive cruise control (ACC) system, as shown in Fig. 4, extending an open experimental environment [18], [28], to answer the research questions. We run the ego vehicle on a three-meter lane with an external robot vehicle in front of the ego vehicle driving at a constant speed as shown in Fig. 5. The ego vehicle's two autonomous driving assistance systems operate simultaneously and automatically. The LKA system observes how much the vehicle deviated from the lane center using a color sensor facing the lane and computes the steering wheel angle to keep the vehicle following the lane center. The ACC system observes the distance to the front vehicle and calculates the driving speed to keep a safe distance. During the operation, two systems record the environmental observation from the sensors (i.e., the color and distance sensor values) and the actuation values (i.e., the steering wheel angle and driving speed) at 20 Hz. For collecting an FOT log, the two vehicles were one meter apart at the beginning, and the driving trace log was recorded until they arrived at the end of the experimental lane. We collect 50 FOT logs to evaluate our approach. Example of the obtained FOT log is shown in Fig. 6.

C. Initial MR Generation

Initial MR is written by the engineers with their knowledge on the CPSoS and its environment. For this experiment, four

TABLE I
INITIAL METAMORPHIC RELATIONS

	Initial MR	Description	Value Types
MR1	$(v_0^0 > c_0, v_0^1 < c_0)$	If the color value is **greater** than the color goal, then the color value will be **less** than the color goal after 1 tick.	v_0^t: Color, c_0: Color goal
MR2	$(v_2^0 > c_1, v_2^1 < v_2^0)$	If the distance value is **greater** than the distance goal, then the distance value will **decrease** after 1 tick.	v_2^t: Distance, c_1: Distance goal
MR3	$(v_0^0 > c_0, v_1^1 < v_1^0)$	If the color value is **greater** than the color goal, then the angle after 1 tick will **decrease** after 1 tick.	v_0^t: Color, v_1^t: Angle
MR4	$(v_2^0 < v_2^1, v_3^0 < v_3^1)$	If the distance value **increased** after 1 tick, then the speed will **increase** after 1 tick.	v_2^t: Distance, v_3^t: Speed

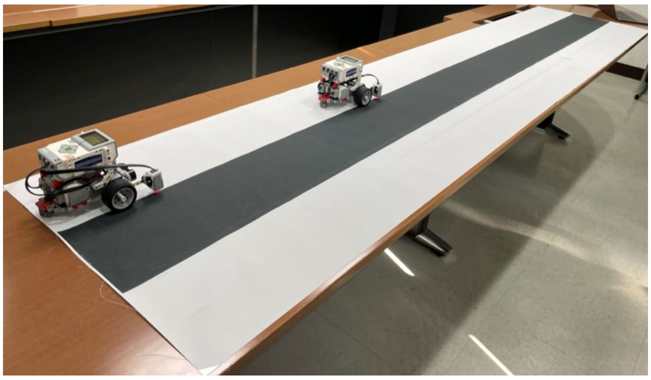

Fig. 5. Case Study: Physical Implementation of Autonomous Robot Vehicle

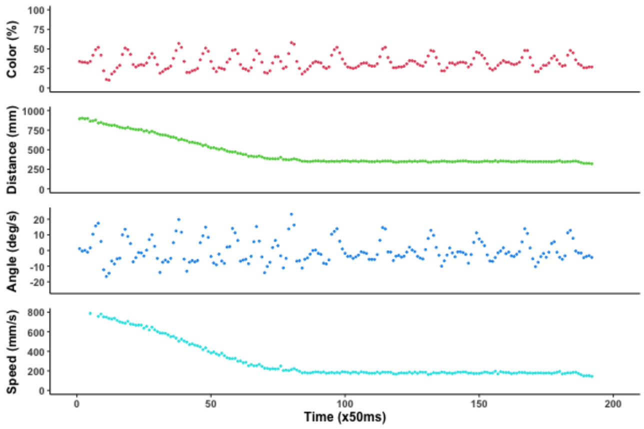

Fig. 6. Field Experiment Data of a Single FOT Log

initial MRs were manually written as shown in Table I. These MRs demonstrate the relationships of both the observation values, such as the color and distance, as well as the actuation values, such as the angle and speed. Since these values are time-series values, time relationship between the variables is specified. In addition, the color goal is set to 33% and the distance goal set to 200 mm to be utilized in our experiment as the constant values.

Initial MRs 1 and 3 aim to assess the LKA system by observing the color (observation) and angle (actuation) values. The goal of the LKA system is to guide the autonomous vehicle to follow the lane center, and thus when the color value is greater than the color goal, the system is expected to correct its angle to converge to the color goal. Initial MR1 describes the relationship of the color observation at time 0 and time 1. At time 1, the color observation is expected to observe the effect of the correct system reaction to observing the color value greater than the color goal at time 0. Initial MR3 describes the relationship of the color observation at time 0 and the angle value at time 1. To correct the autonomous vehicle to follow the lane center, when the color value at time 0 is greater than the color goal, the angle value at time 1 is expected to decrease compared to the angle value at time 0. Based on the understanding of the LKA system and its goal, initial MRs 1 and 3 were written.

Initial MRs 2 and 4 aim to assess the ACC system by observing the distance (observation) and speed (actuation) values. The goal of the ACC system is to maximize driving speed while keeping a safe distance to the front vehicle. Consequently, when the distance value is greater than the distance goal at time 0 (MR2), the distance observation at the next time tick (time 1) is expected to decrease as the system should have increased its speed in attempt to converge to the distance goal. Initial MR4 aims to assess a situation when the system observed an increase in the distance value from time 0 to time 1. Such increase in the distance value indicates that the system can output higher speed while maintaining a safe distance to the front vehicle. Subsequently, the system is expected to output higher speed at time 1 than the previous speed at time 0. The experiment was performed for each initial MR, which was used to generate the initial seed population.

TABLE II
EXPERIMENT PARAMETERS

	Parameter	Value
Genetic Algorithm	Population Size	100
	Crossover Rate	0.6
	Crossover Proposition Rate	1.0
	Mutation Rate	0.2
	Mutation Proposition Rate	1.0
	Budget	10
Fitness Function	Maximum Time Difference	30
	Time Penalty Coefficient	0.1

D. Experiment Parameters

The proposed approach utilizes genetic algorithm to evolve MRs. The genetic operators are applied according to certain probabilities, such as the crossover rate and mutation rate. Additionally, the proposed approach supports the hierarchy of MR. Consequently, genetic operation may occur at both the proposition level and value-operator level of MRs. The experimental parameters are shown in Table II, which were selected based on preliminary experiments.

The population size of 100 configures the total number of generated MRs at each iteration of the search. The total number of the search iterations is specified by the budget, which is also the stopping criteria. The budget specified in this experiment is 10, and thus once the search was executed 10 times, 100 MRs and their relevant information, such as the total number of tp, tn cases, and scores are saved.

Because the initial seed population is generated by modifying small parts of every element in the initial MR, the crossover rate is set to 0.6 or 60% probability, while the mutation rate is set to 0.2 or 20% probability. The crossover rate is the rate which the genetic materials of the survived individuals are exchanged. The mutation rate is the rate at which new genetic material is introduced to the population. By setting the genetic operator rates as such, the genotype of the initial MR is retained at a high rate, while still exploring the search space.

The crossover proposition rate (CPR) and the mutation proposition rate (MPR) configure at which level each genetic operation occurs. Due to the small budget, applying the genetic operators at low-level components have a high chance of introducing cohesion penalty. Therefore, CPR and MPR are set to 1.0 or 100% probability, indicating all genetic operation in this experiment occurs at the proposition level.

Time and cohesion penalties are included in the fitness function to guide the search for meaningful MRs. Because the two systems of the autonomous robot vehicle recorded their observation and actuation values at 20Hz, each time tick is about 50ms. The maximum time difference allowed is set to 30 ticks, which is equivalent to 1.5s. This time difference is approximately one-sixth of the entire operation, which is around 200 ticks or 10s. The time penalty coefficient is 0.1. To warrant minimal number of MRs survived in the finally generated MRs that are penalized by cohesion penalty, the cohesion penalty step size p_{ch} is simply set to $\frac{1}{budget}$.

E. Results

Generated MRs were evaluated according to the RQs defined above. Table III shows the results for RQs 1 and 2. RQ1 assesses the cost of generating MRs, which is evaluated by the ratio of meaningful MRs from a list of MRs. The meaningfulness of MRs are determined by their scores, as they imply the ratio that satisfies the cause-and-effect property. The results show that the proposed approach generated 62 MRs with improved scores on average. This allows the engineers to obtain 62 realistic MRs when specifying a single MR, and thus reducing the significant manual effort required by the engineers in defining and evaluating MRs.

RQ 2 aims to look at the effectiveness of generating MRs, which is analyzed by the sum of tp cases in the FOT logs. The increase in tp cases indicates that the CPSoS satisfies the MR more frequently. On average, 61 MRs were generated with higher number of tp cases, which indicate that the proposed approach is effective at generating MRs that reflect the real CPSoS operations.

TABLE III
RESULTS FOR RQ1 & RQ2 (OUT OF 100 GENERATED MRS)

	MR1	**MR2**	**MR3**	**MR4**	**Avg.**
# of Improved MRs	100	100	46	2	62
# of MRs with $> tp$	94	100	40	11	61

The results for RQ3 is shown in Table IV, which shows the comparisons of the proposed approach and the baseline method of manually generating MRs. For all four initial MRs, the generated MR achieved higher tp cases and score, while decreasing tn cases. For MR1, the generated MR improved the initial MR by 337.3% in the number of tp cases and 312.5% for scores, while decreasing the tn cases by 67.4%. These quantities reveal that the proposed approach performs well compared to the baseline method of manually generating MRs.

Further analysis of the generated MRs is needed to gather useful information about the MRs. The score of initial MR1 is 18.14%, which indicates that the MR1 property occurred less than one-in-five cases in the real operation of the CPSoS. The generated MR1 improves the score by 312.5% by modifying the operator in the if proposition from $>$ to $<$. This reveals that the property initially defined and understood by the domain expert was opposite of what actually happens in the real operation. The engineers can gain understanding that the environment rarely changes so quickly to converge to the specified goal.

Generated MRs for MR2 and MR3 both revealed change in the time component to improve the initial MRs. The initial score of MR2 was 50.44%, indicating that the system observed the effect of the system reaction of decrease in distance value after one-tick delay only half the time. On the other hand, when the distance value after 29-tick delay is compared, the system observed the effect of the system reaction 74.88% of the time, a 48.4% increase. Similarly, the system reaction to the observation of color value in initial MR3 satisfied the property 67.07% when the actuation of angle value was compared after one-tick delay. However, the angle value after 12-tick delay achieved the score of 82.42%, indicating that such delay need to be accounted for when evaluating the system behavior. By utilizing the scores of generated MRs with different time component, engineers may also find the time distribution that best satisfies the necessary properties that define the MRs, and thus account for temporal uncertainty that may arise in a CPSoS operation.

Interestingly, the improvement made in MR4 is very slim with 0.2% increase in score, which can be explained by the fact that the initial MR captures the real operation well with a score of 98.0%. This may also explain the analysis of RQ1 and RQ2 for MR4. The proposed approach only generated 2 MRs with improved score and 11 MRs with higher tp cases for MR4, while the numbers of improved MRs and MRs with

TABLE IV
RESULTS FOR RQ3

		MR	Description	tp		tn		Score	
MR1	Initial	$(v_0^0 > c_0, v_0^1 < c_0)$	If color t+0 $>$ color_goal, then color t+1 $<$ color_goal	765		3453		18.14	
	Generated	$(v_0^0 \boldsymbol{<} c_0, v_0^1 < c_0)$	If color t+0 $<$ color_goal, then color t+1 $<$ color_goal	3345	**337.3%**	1126	**-67.4%**	74.82	**312.5%**
MR2	Initial	$(v_2^0 > c_1, v_2^1 < v_2^0)$	If distance t+0 $>$ distance_goal, then distance t+1 $<$ distance t+0	4845		4759		50.44	
	Generated	$(v_2^0 > c_1, v_2^{\mathbf{29}} < v_2^0)$	If distance t+0 $>$ distance_goal, then distance t+29 $<$ distance t+0	6155	**27.0%**	2065	**-56.6%**	74.88	**48.4%**
MR3	Initial	$(v_0^0 > c_0, v_1^1 < v_1^0)$	If color t+0 $>$ color_goal, then angle t+1 $<$ angle t+0	2829		1389		67.07	
	Generated	$(v_0^0 > c_0, v_1^{\mathbf{12}} < v_1^0)$	If color t+0 $>$ color_goal, then angle t+12 $<$ angle t+0	3273	**15.7%**	698	**-49.7%**	82.42	**22.9%**
MR4	Initial	$(v_2^0 < v_2^1, v_3^0 < v_3^1)$	If distance t+0 $<$ distance t+1, then speed t+0 $<$ speed t+1	2947		60		98.00	
	Generated	$(v_2^0 < v_2^1, v_3^0 \boldsymbol{\leq} v_3^1)$	If distance t+0 $<$ distance t+1, then speed t+0 $\leq$ speed t+1	2954	**0.2%**	53	**-11.7%**	98.24	**0.2%**

higher tp cases were about 100 for initial MR1 and MR2. By changing the operator on the then proposition in MR4 from $<$ to $\leq$, MR4 is improved to achieve a score of 98.24%. However, such improvement is trivial, and thus we can conclude that the initial MR4 defined by the domain expert correctly captured the CPSoS property.

Based on this case study on an autonomous robot vehicle, we validated that our approach can reduce cost while generating effective MRs that capture realistic properties of CPSoS. We believe that our novel framework of applying MT on CPSoS will promote further research into MR generation and MT on CPSoS. In addition, the proposed approach will allow for various CPSoS engineering activities based on objective field data, which includes the uncertain environment within the testing boundary.

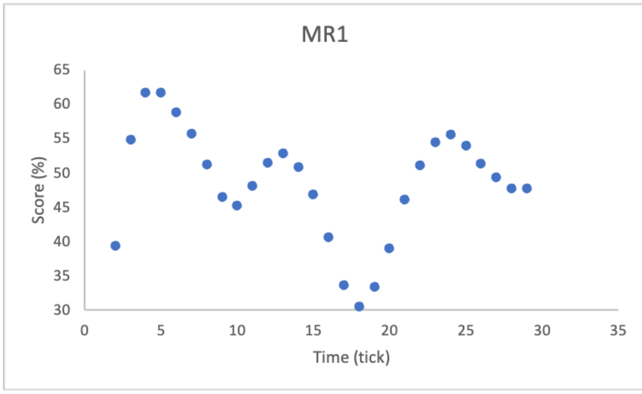

Fig. 7. MR1 Time Distribution vs Scores

V. DISCUSSION

Sensitivity analysis of experiment parameters. GA relies on parameters to guide the search in the search space. Depending on the configurations, the generated MRs may differ. For example, when the mutation rate is high, more new genetic materials are introduced to the population from random generation. With higher diversity in the population, the search may lead to varying MRs. The current approach only considered one GA configuration. The generated MRs discussed in Table IV show MRs from the initial seed population, indicated by a modification of a single element to the initial MRs. Therefore, sensitivity analysis of GA parameters may reveal other MRs that investigate the relations outside the initial seed population.

In addition to the GA parameters, time penalty used to evaluate the MRs may also be analyzed. In the proposed approach, the time penalty is a step-wise function. Time penalty is only applied when the maximum time difference between the variables in an MR is outside the threshold. Instead of using the step-wise function, the time penalty may be modified to a linear function to penalize MRs with greater time difference at a higher rate. Consequently, MRs that capture immediate cause-and-effect relations are favored to survive the evolutionary pressure. Therefore, analysis on the fitness function parameters may also reveal other MRs that were not generated in this study.

Case studies on various CPSoS configurations to reveal relevance to software regression testing The proposed approach was evaluated in a case study of an autonomous robot vehicle consisting of LKA and ACC systems. The user-configurable parameters allow updates to the system by varying the system behavior. Although we used FOT logs of one configuration, it may also be interesting to investigate other configurations supported by the robot vehicle. One MR generated from one configuration may not apply well on a different configuration. We expect that such differences in MR evaluations will reveal the variations among different system versions.

Additional use case of our method to evaluate the quality attribute of CPSoS. Our approach focused on generating MRs given initial MR and field experiment data. The generated MRs were then evaluated based on their scores to reveal whether such property was satisfied or not, and to suggest more realistic MR. In analyzing the evaluation result for RQ1, we found that a series of time modifications on variables returned improved scores. When we plotted the result based on the time tick versus the scores as shown in Fig. 7, we observed that the graph is periodic. Therefore, we expect that our approach can also be used to find a reliability distribution among the generated MRs.

VI. Threats to Validity

Specifying an initial MR and selecting the generated MRs for analysis may introduce bias, and thus threaten our study. To overcome this threat, authors derived the initial MRs based on their respective understanding of the case study subject. In addition, all of the generated MRs selected for analysis are from a pool of MRs with improved scores compared to the initial MR.

Our simplified case study on the automotive domain threatens the generality of the evaluation results. However, we utilized the experiment environment of a real physical CPSoS with two subsystems (LKA and ACC systems) [18] using robot vehicles. From this experiment, 50 FOT logs were used to search for MRs in the evaluation and capture the uncertainties innate to the physical experiment setting. In addition, this paper proposes and utilizes the MR template that reshapes the field experiment data as variables specified by indices and time. In doing so, the proposed approach can be applied to domains other than autonomous vehicles that generate data of time-series nature.

The proposed approach employs GA, which presents stochasticity. The stochastic nature of the underlying technique used in this paper may challenge the validity of the generated MRs. To overcome this threat, the proposed approach also utilizes the initial MR defined by the engineer to direct the starting point in the search space. In addition, the entire time space is explored to evaluate the fitness score of each MR. Low fitness scores can be assigned to MRs that detect system faults when such faults occur only a few times during the operation. Granular analysis of MR per time progression remains a future work to identify such MRs that may reveal faults.

This paper focuses on MR generation and analyzes the generated MRs, rather than applying MT on CPSoS to discover real faults in the system. Possible application of the proposed method in assistance of MT on CPSoS may stem from investigating both the initial MR and generated MRs. The low fitness score of initial MR may indicate 1) there is a fault in the system, or 2) initial MR can be incorrect, thus challenging the discovery of faults in the system. The first case assumes that the initial MR is true, which is simply conducting MT on the system. In the second case, generated MRs with higher scores than the initial MR could indicate more accurate depiction of the system behaviors, which is difficult to pinpoint due to the varying results of CPSoS stemming from innate uncertainties. For future, generated MRs and data logs may be used to determine the pass or fail criteria of a CPSoS.

VII. Related Work

Chua et al. [29] applied a top-down approach in defining MRs to apply MT on SoS. They defined metamorphic relation patterns and metamorphic relation input patterns [16], an abstraction of MRs, based on SoS dimensions, such as emergent behavior and dynamic reconfiguration, defined by Nielsen et al. [5]. However, they manually defined the MRs based on SoS dimensions. In addition, their target system is on SoS, and thus their approach is limited when analyzing the continuous operation and uncertainties from the physical environment of a CPSoS.

Lindvall et al. [14] generated models to apply MT on autonomous drones. The models create test cases and explore the SUT paths. They also identified MRs using the equivalence properties when the input test case is geometrically transformed. Their case study used autonomous drone simulation and identified safety-critical corner cases, such as failing to detect landing pads due to shadows. In addition, they record the path of the drone generated from the continuous flight from point A to point B. Although they applied MT on an example case of a CPS, their MRs are manually identified by the equivalence properties.

Ayerdi et al. [17] applied MT on an industrial case study on elevator systems. They proposed a search-based method to generate MRs by defining the initial metamorphic relation input patterns. Although the elevator system is a CPS, the continuous data generated from continuous operation of CPS is not considered. In addition, there is no support for specifying metamorphic relation input patterns for practitioners to test using their own MRs.

The studies scrutinized here either considered SoS and CPS as their SUT to generate MRs and apply MT. On the other hand, our approach aims to generate MRs for CPSoS with a focus in addressing uncertainties stemming from CPSoS operation. In addition, the extensibility of the proposed approaches and case studies is unclear as they all required expert understanding of the SUT or the characteristics of the SUT, such as the dimensions of SoS. Although our approach is also initialized with the MR specified by the engineers, we automatically search for better MRs, and thus do not require expert knowledge or huge effort to generate the initial MR. Lastly, the case studies conducted here utilized simulations. In contrast, our case study was conducted on a real physical implementation of an autonomous robot vehicle, which implements two subsystems.

VIII. Conclusion

A CPSoS is characterized by uncertainties from its continuous operation in a physical environment and emergent behaviors of constituent systems. These uncertainties challenge the testing process to verify and improve the CPSoS, especially when the correct behavior of a CPSoS in a given situation is unknown, so-called oracle problem. To alleviate the oracle problem, MT has been applied to reveal faults in various systems. MT defines MRs from the relationships among multiple inputs and outputs of system execution and checks for violations of the identified MRs. However, identifying MRs requires domain knowledge and manual effort. In addition, MR generation has not been scrutinized on CPSoS to the best of our knowledge.

In this study, we proposed a GA-based method to generate MRs of a CPSoS. To achieve this, we first proposed a novel framework for MT on CPSoS and compared it to MT on traditional systems. We then proposed a data-driven template of MR to specify initial MR and utilize field experiment data. To generate MRs using GA, we provide the design of genetic operators and a fitness function to guide the search. In a case study on a real autonomous robot vehicle, we validated our approach of generating MRs considering the continuous operation and environment interactions of a CPSoS. The experiment was conducted on four initial MRs defined, and further analysis revealed that the proposed approach was able to generate more realistic MRs that better capture the CPSoS operation. Although our approach showed promising results, further analysis on experiment parameters and case studies on more CPSoSs is needed to validate the generality of the proposed approach.

ACKNOWLEDGMENT

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2022-2020-0-01795) and (No. 2015-0-00250, (SW Star Lab) Software R&D for Model-based Analysis and Verification of Higher-order Large Complex System) supervised by the IITP(Institute of Information & Communications Technology Planning & Evaluation).

REFERENCES

[1] R. Baheti and H. Gill, "Cyber-physical systems," *The impact of control technology*, vol. 12, no. 1, pp. 161–166, 2011.

[2] J. Shi, J. Wan, H. Yan, and H. Suo, "A survey of cyber-physical systems," in *2011 International Conference on Wireless Communications and Signal Processing (WCSP)*, 2011, pp. 1–6.

[3] S. Engell, R. Paulen, M. A. Reniers, C. Sonntag, and H. Thompson, "Core research and innovation areas in cyber-physical systems of systems," in *International Workshop on Design, Modeling, and Evaluation of Cyber Physical Systems*. Springer, 2015, pp. 40–55.

[4] Y.-J. Shin, J.-Y. Bae, and D.-H. Bae, "Concepts and models of environment of self-adaptive systems: A systematic literature review," in *2021 28th Asia-Pacific Software Engineering Conference (APSEC)*, 2021, pp. 296–305.

[5] C. B. Nielsen, P. G. Larsen, J. Fitzgerald, J. Woodcock, and J. Peleska, "Systems of systems engineering: Basic concepts, model-based techniques, and research directions," *ACM Comput. Surv.*, vol. 48, no. 2, sep 2015. [Online]. Available: https://doi.org/10.1145/2794381

[6] I. Sommerville, D. Cliff, R. Calinescu, J. Keen, T. Kelly, M. Kwiatkowska, J. Mcdermid, and R. Paige, "Large-scale complex it systems," *Communications of the ACM*, vol. 55, no. 7, pp. 71–77, 2012.

[7] A. Kane, T. Fuhrman, and P. Koopman, "Monitor based oracles for cyber-physical system testing: Practical experience report," in *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2014, pp. 148–155.

[8] E. T. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo, "The oracle problem in software testing: A survey," *IEEE Transactions on Software Engineering*, vol. 41, no. 5, pp. 507–525, 2015.

[9] T. Y. Chen, S. C. Cheung, and S. M. Yiu, "Metamorphic testing: a new approach for generating next test cases," Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, Tech. Rep. HKUST-CS98-01, 1998.

[10] S. Segura, G. Fraser, A. B. Sanchez, and A. Ruiz-Cortés, "A survey on metamorphic testing," *IEEE Transactions on Software Engineering*, vol. 42, no. 9, pp. 805–824, 2016.

[11] T. Y. Chen, F.-C. Kuo, H. Liu, P.-L. Poon, D. Towey, T. H. Tse, and Z. Q. Zhou, "Metamorphic testing: A review of challenges and opportunities," *ACM Comput. Surv.*, vol. 51, no. 1, jan 2018. [Online]. Available: https://doi.org/10.1145/3143561

[12] S. Segura and Z. Q. Zhou, "Metamorphic testing 20 years later: A hands-on introduction," in *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*, ser. ICSE '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 538–539. [Online]. Available: https://doi.org/10.1145/3183440.3183468

[13] S. Segura, D. Towey, Z. Q. Zhou, and T. Y. Chen, "Metamorphic testing: Testing the untestable," *IEEE Software*, vol. 37, no. 3, pp. 46–53, 2020.

[14] M. Lindvall, A. Porter, G. Magnusson, and C. Schulze, "Metamorphic model-based testing of autonomous systems," in *2017 IEEE/ACM 2nd International Workshop on Metamorphic Testing (MET)*, 2017, pp. 35–41.

[15] Z. Q. Zhou and L. Sun, "Metamorphic testing of driverless cars," *Communications of the ACM*, vol. 62, no. 3, pp. 61–67, 2019.

[16] Z. Q. Zhou, L. Sun, T. Y. Chen, and D. Towey, "Metamorphic relations for enhancing system understanding and use," *IEEE Transactions on Software Engineering*, vol. 46, no. 10, pp. 1120–1154, 2020.

[17] J. Ayerdi, V. Terragni, A. Arrieta, P. Tonella, G. Sagardui, and M. Arratibel, "Generating metamorphic relations for cyber-physical systems with genetic programming: An industrial case study," in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ser. ESEC/FSE 2021. New York, NY, USA: Association for Computing Machinery, 2021, p. 1264–1274. [Online]. Available: https://doi.org/10.1145/3468264.3473920

[18] Y.-J. Shin, E. Cho, H. Kim, and D.-H. Bae, "Hands-on field operational test dataset of a multi-controller cps: A modeled case study on autonomous driving," in *2022 17th Annual System of Systems Engineering Conference (SOSE)*, 2022, pp. 205–212.

[19] V. Le, M. Afshari, and Z. Su, "Compiler validation via equivalence modulo inputs," *SIGPLAN Not.*, vol. 49, no. 6, p. 216–226, jun 2014. [Online]. Available: https://doi.org/10.1145/2666356.2594334

[20] Z. Q. Zhou, S. Xiang, and T. Y. Chen, "Metamorphic testing for software quality assessment: A study of search engines," *IEEE Transactions on Software Engineering*, vol. 42, no. 3, pp. 264–284, 2016.

[21] S. Segura, J. Troya, A. Durán, and A. Ruiz-Cortés, "Performance metamorphic testing: Motivation and challenges," in *2017 IEEE/ACM 39th International Conference on Software Engineering: New Ideas and Emerging Technologies Results Track (ICSE-NIER)*, 2017, pp. 7–10.

[22] T. Y. Chen, P.-L. Poon, and X. Xie, "Metric: Metamorphic relation identification based on the category-choice framework," *Journal of Systems and Software*, vol. 116, pp. 177–190, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121215001624

[23] J. Zhang, J. Chen, D. Hao, Y. Xiong, B. Xie, L. Zhang, and H. Mei, "Search-based inference of polynomial metamorphic relations," in *Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering*, ser. ASE '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 701–712. [Online]. Available: https://doi.org/10.1145/2642937.2642994

[24] M. Zhang, B. Selic, S. Ali, T. Yue, O. Okariz, and R. Norgren, "Understanding uncertainty in cyber-physical systems: A conceptual model," in *Modelling Foundations and Applications*, A. Wasowski and H. Lönn, Eds. Cham: Springer International Publishing, 2016, pp. 247–264.

[25] F. Oquendo, "Coping with uncertainty in systems-of-systems architecture modeling on the iot with sosadl," in *2019 14th Annual Conference System of Systems Engineering (SoSE)*, 2019, pp. 131–136.

[26] M. Zhang, S. Ali, T. Yue, R. Norgren, and O. Okariz, "Uncertainty-wise cyber-physical system test modeling," *Software & Systems Modeling*, vol. 18, no. 2, pp. 1379–1418, 2019. [Online]. Available: https://doi.org/10.1007/s10270-017-0609-6

[27] Y. Barnard and O. Carsten, "Field operational tests: challenges and methods," in *Proceedings of European Conference on Human Centred Design for Intelligent Transport Systems, Eds edn. HUMANIST publications, Lyon*, 2010, pp. 323–332.

[28] Y.-J. Shin, L. Liu, S. Hyun, and D.-H. Bae, "Platooning legos: An open physical exemplar for engineering self-adaptive cyber-physical systems-of-systems," in *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2021, pp. 231–237.

[29] K. K. A. Chua, D.-H. Bae, and E. Jee, "Metamorphic testing for reliability in system of systems," in *2021 28th Asia-Pacific Software Engineering Conference (APSEC)*, 2021, pp. 390–400.