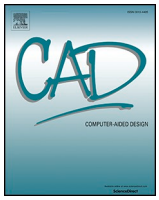
Computer-Aided Design 174 (2024) 103735

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad



Methodological paper

Synchronous integration method of mechatronic system design, geometric design, and simulation based on SysML

Chu Changyong [a,b,*], Zhang Chunjia [c], Yin Chengfang [c]

[a] School of Mechanical Engineering, Hangzhou Dianzi University Information Engineering College, Hangzhou 311305, China
[b] State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
[c] School of Mechanical Engineering, Hangzhou Dianzi University, Hangzhou 310018, China

ARTICLE INFO

Keywords:
SysML
CAD
Simulink
M2T
TTRS
Integrated design

ABSTRACT

The benefits of integrated design using the Model Based System Engineering (MBSE) approach in the design process of mechatronic systems have gradually become apparent. The automatic generation of simulation models and geometric models based on System Modeling Languages (SysML) models enables system engineers to swiftly analyze and simulate system performance, visually depict design outcomes, and expedite the product development process. Due to the current system modeling's lack of model integration and geometric design functions, this paper proposes an integrated design and simulation method for mechatronic systems that can carry out complete model synchronization and verification, rapid geometric solution generation, and visual representation. Furthermore, a corresponding model synchronization integration framework is established. This framework primarily encompasses system de-sign, system simulation, and geometric design, with its model integration method being model transformation and model synchronization. The paper concludes with an example of the design process of a quadruped robot to validate the framework and its supported methods, providing a reference for other system design and integration endeavors.

1. Introduction

Mechatronic systems (MTSs) are a multidisciplinary field of engineering that typically involves concepts from multiple fields such as electrical engineering, mechanical engineering, control engineering, and computer science [1]. Mechatronic system's complexity arises as a result of the integration of various subsystems in a single product and the involvement of technical teams from multiple disciplines [2]. When developing mechatronic systems, it is critical for engineers and analyzers to understand the design and behavior of the entire system and the details of the integration of the different subsystems involved in the product. The design of mechatronic systems also requires the effective management of the engineering process. Therefore, mechatronic systems are supposed to be designed and modeled as integrated systems from the starting stage and to advance the system design together in terms of all the domains involved. Since mechatronic systems combine technologies from different disciplines, different design options require trade-offs and analysis to find the ideal combination of technologies.

MBSE provides languages, methods, and modeling tools to integrate multidisciplinary systems by automating the interaction between the engineering disciplines involved. MBSE is gradually becoming an accepted method for designing complex systems [3,4] and provides strong support for the integrated design of mechatronic systems.

In the design of systems that employ MBSE, a facility is desired to extract and integrate key design information from various modeling languages and models included in domain-specific modeling tools [5]. It is also crucial to maintain consistency and traceability of information between different models. Multidisciplinary systems are usually implemented utilizing model integration. Zhang [6] develops a new integrated intelligent modeling and simulation language, and Zhang [7] proposed a standardized methodology and a hierarchical, modular, and generic architecture to depict comprehensive and variable industrial robot digital twin (IRDT). Hu [8] addressed the potentials of utilizing MBSE ontology integration for different models.

However, the above MBSE integration frameworks only provide macrolevel integration of multiple models involved in the design process and their corresponding interaction mechanisms with the modeling and simulation tools.They failed to provide a complete solution for the specific integration of multiple models and the specific interaction between various design tools. In this paper, a detailed integration

* Corresponding author.
E-mail address: kevin@hdu.deu.cn (C. Changyong).

https://doi.org/10.1016/j.cad.2024.103735
Received 6 February 2024; Received in revised form 21 April 2024; Accepted 24 May 2024
Available online 28 May 2024

framework for system level design, system simulation, and domain-specific design of mechatronic products are proposed.

The first part of this paper presents the back-ground related to mechatronic system design and MBSE and illustrates the work related to the integration of various design modeling languages and their tools in the MBSE framework; the second part analyzes model integration methods, model transformation methods and describes the work on the integration of SysML models with simulation and geometric models; the third part explains the construction and operation of the model transformation and synchronization functions of the model integration framework; the fourth part takes SolidWorks as an example application tool to describe the integration of SysML and geometric models in the integration framework, which includes the construction of geometric semantic extensions, mapping relationships and specific conversion methods; the fifth part implements the integrated design approach by example of the integrated design process of a quadruped robot and verifies the function of the model synchronization integration framework; the sixth part concludes the work of this paper and discusses the advantages of the integrated design framework and the content waiting for improvement.

2. Related works

2.1. Integration method for the SysML model

By the utilization of system models, especially SysML-based system models, discipline-specific models, and their tools can be managed, transformed, and analyzed through semantic data integration [9]. The SysML specification illustrates that SysML not only provides support for system-level design and the exchange of its models and data but also provides a platform for model unification and integration, which is the SysML language that emerged as one of the intentions [4]. Johnson [10] proposed that SysML semantics can support model integration, but instead of developing a direct mechanism to accomplish model integration and provide it to the user, it relies on the user to implement model integration. SysML is extremely extensible and can customize SysML semantic functionality on demand by creating custom profiles. Profiles add new concepts to the language through the conformational modeling [11] mechanism, which enables SysML to provide do-main-specific descriptions of different domains. For domain-specific modeling, engineers often create additional conformations based on their domain-specific concepts. The SysML extension mechanism allows refinement of the standard semantics of SysML in a strictly additive manner so that it cannot contradict the standard semantics [12].

The model integration framework requires the integration of different models from many different tools including system models, geometric models, and simulation models. Model transformation is the process of model integration from the input source model to the target model, and the corresponding transformation rules and tools are needed to realize model transformation. The current research on model transformation basically follows the guidelines of Model Driven Architecture (MDA) [13] proposed by OMG, and is based on the Meta Object Facility (MOF) to realize model extension and transformation.

Model transformation can be divided into model-to-model (M2M) transformation and model-to-text (M2T) transformation according to the type of model, and M2T also includes model-to-code transformation [14]. The model domain and the text domain involved in the transformation are the two most dominant model representation domains. In the model domain, semi-formal models (e.g. UML models) are mainly represented using graphical representations, which are mostly saved in XMI format in industrial production to enable the inter-action between different types of graphically de-scribed models. In the text domain, the formal models are represented in textual form. Therefore, a complete transformation is needed in the model transformation depending on the representation domain. In some cases, a syntactic transformation needs to be implemented in addition to the corresponding semantic transformation.

2.2. SysML-simulation model integration

The integration of the system model and simulation tool enables the information in the system model to be transferred to the solver of the simulation tool, and the system verification can be completed by the simulation to generate the corresponding analysis results. Therefore, much research is devoted to the establishment of a perfect integration function between SysML models and simulation tools.

Johnson [15,16] introduced a method for the dynamic modeling of continuous systems in SysML based on the mapping between SysML and Modelica, it creates equivalent SysML structures for selected Modelica elements, based on which the continuous dynamics behavior and its simulation can be per-formed in SysML tools, and the dynamic behavior model can be transformed between SysML and Modelica by the utilization of the model transformation framework VIATRA [17]. Thomas Schu-macher [18] proposed and improved the mapping of SysML and CAD elements to develop heterogeneous models, and visualized an exemplary heterogeneous model. Cao [19,20] proposed a hybrid behavioral system-level design using SysML, as well as applied Stateflow and Simscape for system-level simulation, and a profile was created based on the conformal modeling mechanism. Chao Fu [21] studied the lack of executable capability of the standard system modeling language (SysML) within MBSE, the system design-simulation model transformation method was studied. Qing Li [22] propose a com-bination algorithm of HIA based on SysML and Modelica. OMG proposed a formal SysML4Modelica profile [23] to represent the Modelica structure and specified the profile structure and the Modelica language, which can facilitate the transformation of analytical models in SysML into Modelica models. OMG also published the SysML extensions for physical interaction and signal flow simulation (SysPhS) specification [24]. This specification complements the absence of features in SysML for various physical interaction and signal flow platforms and also provides a textual language for mathematical representation based on Modelica syntax, model preprocessing instructions, model transformation rules for Modelica and Simscape, and a platform-independent component library.

In summary, the existing research and commercial software features for system model and simulation model integration are mainly for the physical components of the system model and their hybrid behavior. There is a relative lack of exploration of the ways in which the simulation validation aspects work in concert with the overall system de-sign process, and in addition, domain-specific knowledge (e.g., geometric knowledge) is not properly managed in the early stages of system design.

2.3. SysML-CAD model integration

Physical integration of mechatronic systems is a major issue in their design process, and physical integration is considered one of the important criteria in the selection and decision process of system architecture. However, due to the lack of 3D geometric data in the early design phase, the multi-physics be-havior of mechatronic systems is usually not investigated before the detailed design phase [25]. Evaluation of the 3D spatial structure under multi-physics constraints from the concept design stage can reduce the risk of costly late changes that occur in further design stages, which reduces iterations and shortens the overall design time [26]. Therefore, system engineers have to provide the same geometric information to the technical teams of all domains through the system model and have to analyze and compare this geometric information during the design process and perform some preliminary simulations. In addition, various potential 3D design architectures need to be traced back during the design process to confirm that they meet the initial geometric and physical requirements.

According to the MBSE methodology [27], which mainly uses SysML as the modeling language, conceptual design tasks can be integrated into system-level modeling. Several research efforts have introduced

geometric content into system-level modeling to integrate 3D visual descriptions of graphics [28]. Bohnke [29] defined a UML profile to design the 3D geometry of an aircraft, and this method aims to generate a complex geometry composed of points and then create a complete solid by association, but the method is not suitable for the conceptual design phase because this method is actually more suitable for complex detailed geometries and is not useful for conceptual designs in which the component geometry is not explicitly specified, and the method does not manage assembly constraints. Barbedienne [30,31] and Plateaux [32] proposed a geometric profile of SysML based on the theory of process and topology related surfaces (TTRS [40]) and named it Geometric Extensions Related to TTRS Reference for Unified Design (GERTRUDe), with a corresponding modeling approach and integration framework. However, TTRS theory must be mas-tered when improving such profiles, and the application of SysML to model geometric aspects also requires reference to TTRS modeling and constraint rules, which makes the corresponding modeling process tedious. In addition, the direct application of TTRS to transform the geometric information in SysML requires CAD tools with appropriate surface design capabilities.

In system design, consistency and traceability of geometric data need to be ensured throughout the design phase, especially from consumer requirements to the conceptual design phase. The geometric knowledge specified in the system-level modeling requires the ability to automatically generate the corresponding geometric model in the 3D CAD model. Qamar [33] used ATL [34] with MQL [35] to construct a transformation between SysML models and mechanical CAD models, but the capabilities in terms of model analysis and automated operation are insufficient. Graignic [36] proposed an integration framework. This framework considers the interface between components logically modeled by CATIA V6, but the geometric aspects are not clearly de-scribed. Gielingh [37] emphasized the adoption of neutral models in the integration of models to avoid the establishment of too many transformation tools from one tool or modeling language to another. Most research works consider SysML as a neutral model and in the case of usage of SysML, different profiles are required to establish correspondence [38].

3. Synchronous integration design framework

The model transformation process in this paper starts with the transformation between the logical architecture of the system model and the simulation model and synchronizes and refines the generated models as the system design process progresses. The geometric information in the design process is transformed into a CAD model for visualization, and the integration of the geometric model with the system simulation model allows for more comprehensive verification results. In this paper, Cameo System Modeler is adopted to establish the system model, and Simulink and SolidWorks are adopted as the transformation targets for model transformation and synchronization, as shown in Fig. 1.

The process of model element transformation can be summarized as follows: the model elements in the SysML model in XML format are extracted and sorted in a traversal manner by query statements; the model information is filled into the corresponding transformation template after the inter-mediate model is generated or the results are compared with the intermediate model; and the code to create or update the model is generated. The framework runs in Eclipse and generates executable code text based on the imported SysML file, and the code can be run in the system simulation tool and CAD tool to generate the simulation model and CAD model.

Fig. 2 shows the activity diagram for the process of information extraction and modeling code generation of the system model.

Based on the profiles related to system simulation in SysML and their mapping relationships, a text template for the transformation of the SysML model into a simulation model can be constructed. When Simulink is taken as the transformation target, the architecture of the SysML model is transformed into System Composer. The system requirements

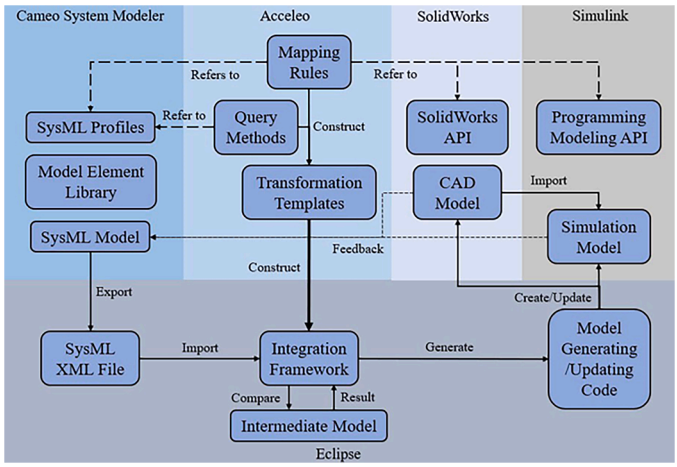


Fig. 1. Model synchronization transformation pattern.

are transformed into Simulink Requirement, the physical structure section is transformed into Simscape, the control logic sections are transformed into Stateflow, and a part of the constraints in the SysML model is transformed into Simulink mathematical operation module for auxiliary calculation.

Simscape Multibody, which belongs to Sim-scape, can also represent and analyze the geometric information of SysML models. Specifically, it expresses the geometric information of the part in terms of basic geometry combined with relative spatial coordinates through the module "Solid" and module "Rigid Transform", and expresses the constraint relationship between components through the module "Joint", which represents the kinematic side. However, Simscape Multibody has the issues of a cumbersome process, lack of intuition, and inability to express the fit relationship between parts in the actual modeling application. Therefore, instead of using Simscape Multibody directly to establish the geometric structure, this paper establishes the geometric information in the SysML model through SolidWorks and Simulink based on the integration function of SolidWorks and Simulink, and then imports it into Simulink. This method not only reduces the difficulty of geometric modeling but also introduces a more complete visual representation of the geometric structure, which makes the analysis and modification of geometric information more convenient.

4. Construction and integration of geometric information in sysml

4.1. Construction of geometry and constraint in SysML

The inclusion of geometric information in the early design phase before detailed design with the MBSE approach can improve data consistency for collaborative design between multidisciplinary teams to select conceptual architectures under geometric and physical constraints. Therefore, it has received attention from researchers to provide a modeling framework that maintains a consistent connection between the geometric data of the system model and the specific views of the "3D sketches" in the conceptual design. The construction of such a connection can ensure seamless geometric content consistency and traceability from requirements to further design steps.

The method of generating geometric elements relies on the accuracy required for geometric modeling. Since the integration work in this paper focuses on the conceptual design phase, which requires a simple and fast means to outline the spatial distribution of simplified components of a given architecture to support the selection of the best concept that can satisfy the requirements of the system model [39]. Therefore, the theoretical reference in this paper for topological modeling is the Technology and Topology Related Surface (TTRS) theory [40]. The level

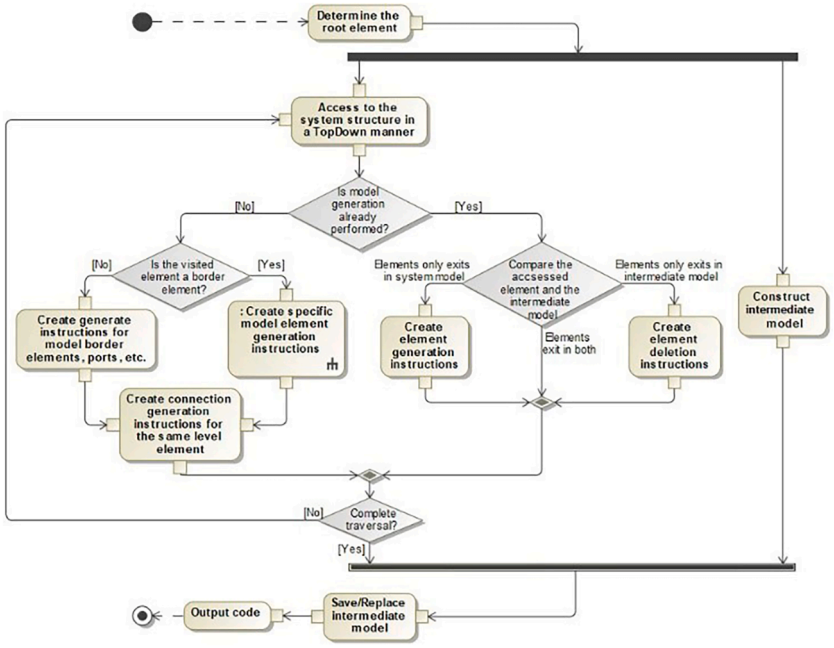


Fig. 2. Model synchronization transformation process.

of abstraction of the TTRS theory is very applicable in the conceptual design phase, where the geometry of most components can be reduced to simple geometries with kinematically invariant symmetries geometry, and kinematic invariance is valuable in 3D environments for selecting displacement directions to meet some of the geometric or physical requirements.

The TTRS theory provides support for the integration of SysML with geometric models, and several studies have developed geometric information integration capabilities for SysML. Nevertheless, the current integration methods based on this theory still have some limitations. For example, the TTRS theory must be mastered for further improvement of such geometric profiles, which im-plies an increase in learning costs; the application of SysML for modeling geometric aspects also requires reference to the modeling and constraint rules of TTRS, which renders the corresponding modeling process tedious; the direct application of TTRS for the transformation of geometric information in SysML requires CAD tools with corresponding surface design capabilities, etc.

In this paper, based on the modeling methods of 3D CAD tools and the APIs they provide, adaptations are made in the extension of SysML based on TTRS theory to make it more suitable for the integration of SysML with CAD tools. In the conceptual design stage of mechatronic systems, the structure of mechatronic products can be simply represented by some basic geometries, while the refinement of the product geometry and its appearance is carried out in the detailed design stage to obtain the final structure of the mechatronic products. According to the need for geometric information in the conceptual design stage [41], the sphere, cylinder, prism, and rotation surfaces of the TTRS class are selected in this paper as references for the construction of modeling elements representing geometric shapes. In 3D CAD tools, the basic geometric shapes that include these types of surfaces are spheres, cylinders, prisms, and rotations, which can be generated by stretching and rotating features, among which cylindrical and prismatic features are generated along with cross-sectional planes. Since the TTRS class can be represented by Minimal Reference Geometrical Element (MRGE), MRGE can be further divided into Reduced Geometrical Elements (RGE), i.e., point, line, and surface. These basic geometrical elements are also the constituent elements of TTRS and basic geometric shapes.

In this paper, these basic geometric elements are defined first, with stereotypes <<point>>, <<line>> and <<plane>> corresponding to points, lines, and planes based on the metamodel <<block>>, and additional modeling elements such as <<curve>> according to the requirements of 3D modeling. Associations between these basic elements are defined, such as points for positioning in line elements, lines for determining position and direction in planes, etc. Position-related definitions such as coordinates are also included in these modeling elements that describe basic geometric elements. Based on these basic geometric elements, the basic geometries mentioned above can also be de-fined. Dimensional parameters of these geometries are identified to obtain the corresponding geometric entities, and the simplified representation and positioning capabilities of MRGE enable these geometric entities to be modeled in 3D space with certain Boolean operations for the conceptual design phase of the part.

The relationship between the basic geometric elements and the geometry is shown in Fig. 3, where the "sphere" contains the radius parameter, and its position is represented by the point indicating the center point, the "revolution" The specific shape and its position are defined by the curve that represents the base line and the line that coincides with the axis, the " cylinder" contains the base radius and height parameters, and its position is represented by the line that coincides with the axis, and the " prism " contains the height parameter, its cross-sectional shape is represented by a closed curve and its position is represented by a plane coinciding with the bottom surface, "Line" is further divided into finite line segments and infinite lines. In consideration of the association between the basic geometry, parts, and assemblies, additional conformational shapes <<prt>> and <<asm>> based on the <<block>> metamodel are defined in this paper to represent the system model boundaries corresponding to the part and assembly files. The profiles obtained by the definitions above are shown in Fig. 4.

The mechanical part of the mechatronic system is mostly for physical support and motion, which consists of several parts assembled together, and the basic geometry needs to be related by Boolean operations within the parts. Therefore, it is necessary to introduce geometric constraints in the conceptual design phase to confirm the combination between basic geometry and parts, and between parts and components.

Boolean operations between geometries are divided into three types: union, intersection, and subtraction. In this paper, stereotypes <<boolean+>>, <<boolean*>> and <<boolean->> based on the metamodel of <constraint>> are defined ac-cording to the

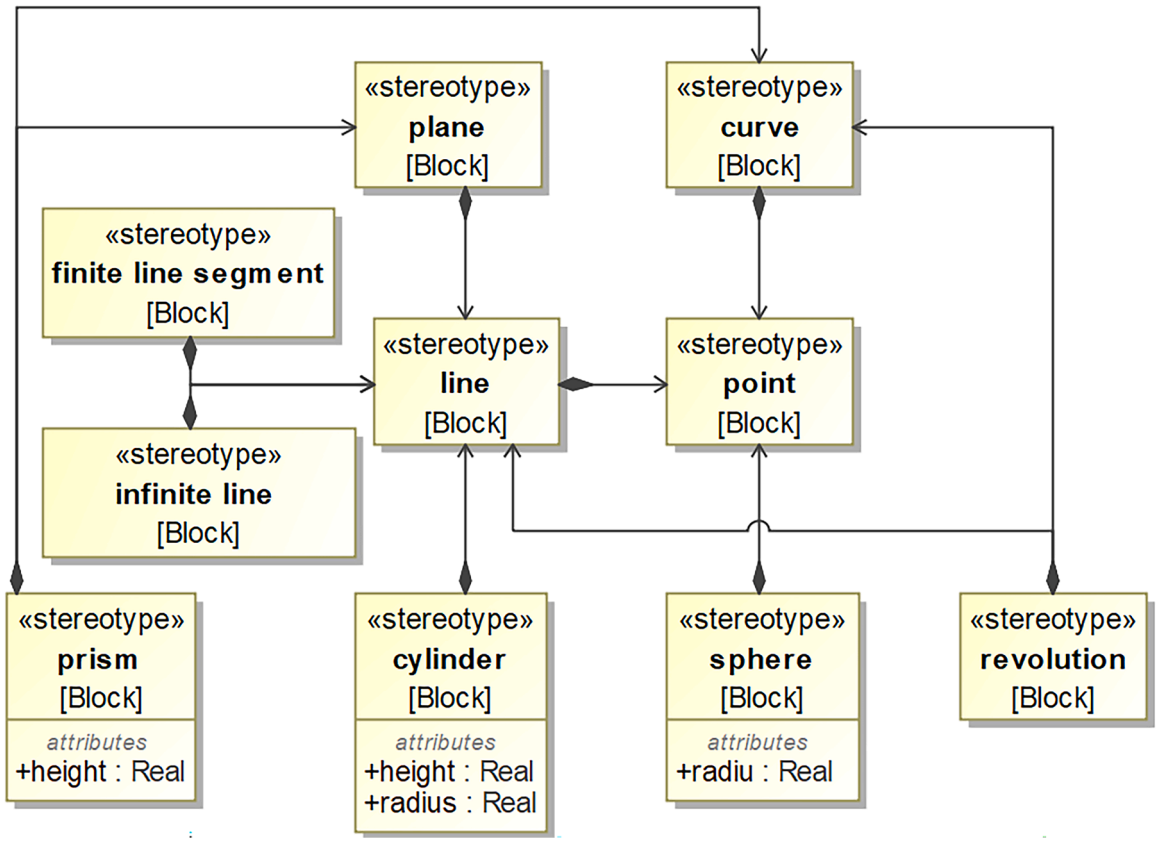


Fig. 3. Basic geometric elements, geometric shapes and their correlations.

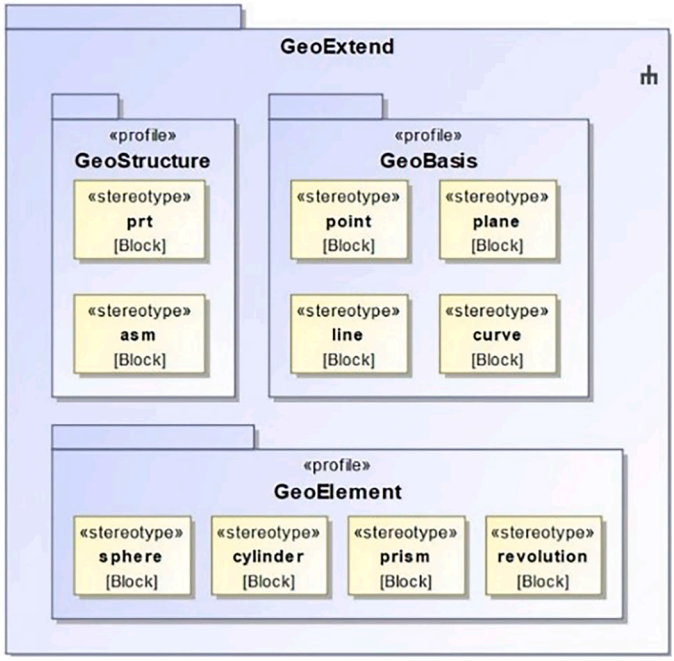


Fig. 4. Definition of geometry and structure information.

characteristics of Boolean operations, which represent Boolean summation, intersection, and subtraction, respectively, and in Boolean subtraction, the subtracted geometric entities are compared with the geometric entities as the "subtractor" by the definition of parameter properties. subtracted geometric entities and geometric entities as "subtractors" are distinguished by defining parameter attributes in Boolean differentiation. The association of the geometric entities can be determined by means of binds between the basic geometric entities and these constraints representing Boolean operations, thus confirming the geometry of the corresponding part in the conceptual design phase.

The types of constraints between MRGEs include coincidence, parallel and vertical, and the corresponding constraints are also included in the fit information within the assembly in 3D CAD tools. Based on this correspondence, this paper defines the conformational shapes <<parallel>>, <<coincidence>>, <<vertical>>, <<horizontal>> and <<distance>> that represent the assembly relationships, which represent parallel, coincidence, vertical, horizontal and distance constraints, respectively. The constraints can be applied to the geometric elements in the part for orientation, which can realize the construction of the fit relationship between parts.

The profile that contains these constraints is shown in Fig. 5.

Compared with the SysML extensions con-structed directly from the geometric information representation of 3D CAD model files (e.g., STEP), profiles in this paper are based on the TTRS theory and are more concise and suitable for conceptual design. Moreover, based on the extension mechanism of SysML, these profiles can be further modified or expanded to suit the specific modeling needs of the design process.

4.2. Construction of model mapping relationship and generation method

The definition of the geometric information in SysML enables designers to incorporate the structural information of the mechatronic system in the conceptual design at the system level. The design information can be correlated with requirements information, which provides the basis for the verification of system geometry requirements. More importantly, the system model can be integrated with the geometry model early in the design process, and the analysis, verification, and validation of the mechatronic system geometry information can be

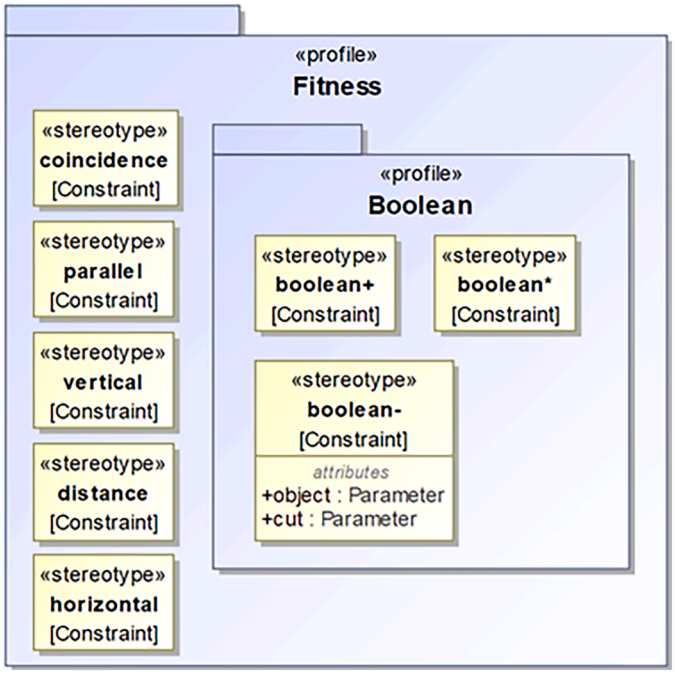


Fig. 5. Definition of geometric constraint information.

advanced from the detailed design stage to the logical and physical architecture stage. In the actual design process, the analysis of geometric information can be carried out with the help of visual representation and other functions of 3D CAD tools. To ensure the traceability of geometric information, it is necessary to establish the mapping relationship between the SysML model and the geometric in-formation in the CAD model.

In SolidWorks, spheres and rotating bodies can be represented by rotating features, and cylinders and prisms can be represented by stretching features. The dimensional parameters and positioning information required to generate the corresponding features are directly matched in the SysML model elements or obtained after analytic geometry operations. In single-part modeling, the Boolean summation is performed by default when multiple features are generated, and the entities treated as "subtractors" in the Boolean difference operation correspond to the "stretch/rotation cut" features. Although the "Add", "Delete" and "Common" options in "Combine Features" can also correspond to Boolean summation, difference, and intersection, respectively, these commands are mostly applied to the combination of parts to form new parts with more complex shapes and do not apply to the combination of simple geometry in the conceptual design stage, so it is not mapped in this paper. The constraints between geometric elements can be expressed by constraints in the assembly, which include coincident constraints, parallel constraints, and vertical constraints. The specific mapping relationships between model elements in SysML geometry profiles and SolidWorks elements can be summarized in Table 1.

Table 1
Mapping relationship between sysml geometry profiles and solidworks elements.

	SysML elements	SolidWorks elements
Shape	sphere	Rotational feature (semicircular section)
	cylinder	Stretching feature (circular section)
	revolution	Rotational feature
	prism	Stretching feature (polygon section)
Constraint	boolean cut	Cutting feature
	coincidence	Coincidence constraint
	parallel	Parallel constraint
	vertical	Vertical constraint

For other 3D CAD tools, a similar idea can be adapted to map the shape and constraint elements in the profile to the corresponding entities and constraints in the tool and to establish the geometric model in a relatively simple modeling way to meet the needs of geometric modeling and analysis in the conceptual design phase.

The integration of system models with CAD models requires the generation of geometric models in 3D CAD tools based on SysML geometric pro-files and mapping relationships. Unlike simulation models that are created using visual programming languages like SysML, Simulink, Modelica, and other system simulation models have similar modeling structures to simplify the process of transforming simulation models. On the other hand, CAD models must be generated to ensure that all interrelated geometric information within the respective part or assembly module has been captured. The CAD model generation requires confirmation that all interrelated geometric information in the corresponding part or assembly module has been traversed. Meanwhile, the simulation module representing geometric information in the system simulation model can be integrated with 3D CAD tools to represent the geometric model in the simulation model. Therefore, the transformation process of the geometric model and simulation model are integrated in this paper, and the generation process of CAD modeling code is incorporated into the action "Create specific model element generation instructions" of the simulation model transformation process, which is shown in Fig. 6. The Postorder method starts from the bottom element without child elements and then moves to the same parent element of the same level, and then visits the parent element after the traversal is completed.

The geometric information in the system model is mainly available in the physical architecture phase, and in the SysML model of the electromechanical system, in the physical structure part, such as the mechanical subsystem. After the system de-sign has entered the physical architecture phase and the geometric information has been modeled, the geometric information can be transformed to generate geometric models in 3D CAD tools. In this paper, SolidWorks is adopted as the transformation target to generate geometry modeling scripts based on SysML geometry profiles, mapping relationships, and SolidWorks API, with the SolidWorks macro tool to run the scripts to generate CAD models. Due to the large number of parameters involved in geometric modeling and the relatively large amount of text in the modeling code, the pseudocode of the main part of the corresponding modeling script is simplified and shown in Fig. 7.

In case the geometry model is integrated with the simulation model, the system model is transformed into a placeholder corresponding to the geometric module when it is transformed into the simulation model, and after the geometric elements are accessed in Top-down mode, the geometric elements are traversed in Postorder mode and the geometric model is generated, and then the simulation model is connected to the geometric model to complete the transfer of geometric information.

4.3. Synchronous updating method of the simulation model

After integrating the system logical architecture with the simulation model, the system design process shifts from relying solely on a separate system modeling platform to depending on both the system modeling platform and the simulation platform. The typical optimization approach for the logical architecture of the system model involves solving the model in the simulation environment through model transformation or co-simulation, and then incorporating the results back into the system model. Adjustments are made to the parameters and structure of the logical architecture, upon which the physical architecture is developed.

Utilizing the model transformation framework presented in this paper enables the transformed simulation model to synchronize with the system model. The complex optimization procedures are streamlined into a unified and synchronized optimization process. This reduces the back-and-forth exchange of information between these platforms,

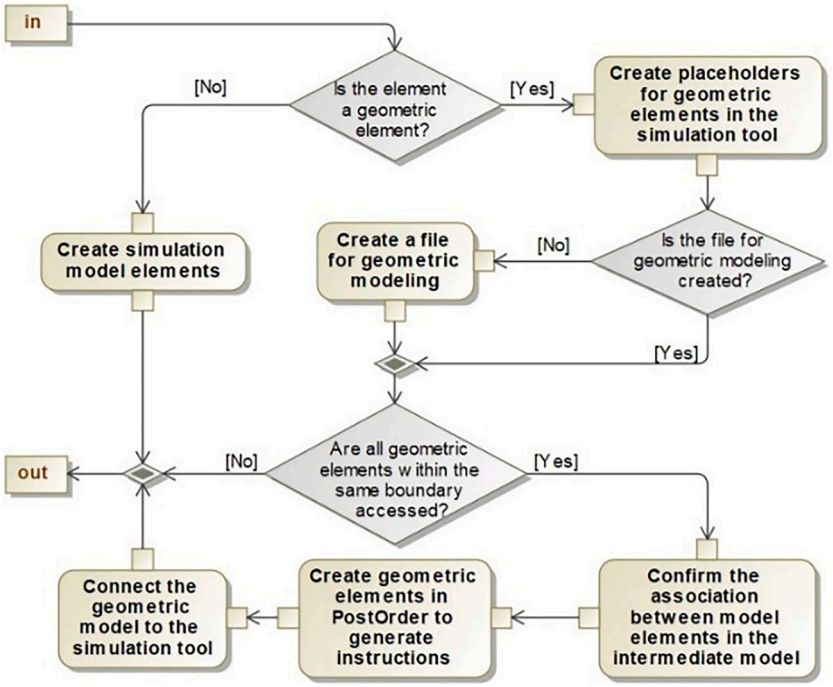


Fig. 6. Geometric model transformation process incorporated into the simulation model transformation.

enhances model reusability, and boosts design efficiency. The process of model transformation and synchronization is illustrated in Fig. 8.

For the exported system model, the transformation framework performs a model comparison to validate altered sections and generate an update script. This script includes command statements for adding, deleting, and modifying model elements and their relationships in the simulation model based on the revised content. The intermediate model is likewise refreshed after code generation.

5. Case study

5.1. SysML modeling for mechatronic system

In the system-level design of mechatronic products, the integration and verification of the entire model at the early stage enable the design, integration, and analysis of the system model to be more intuitive and convenient. In this chapter, a quadruped robot design process is selected as an example to illustrate the integrated design method. The system model of the quadruped robot is constructed by CSM, and the corresponding design information is transformed into Simulink and SolidWorks synchronously by the integration framework. The effect and feasibility of the model transformation function in the integration framework are verified through the transformation and analysis of the system model. Meanwhile, the simulation model would be integrated with the geometric model for a more comprehensive system simulation, and this process can verify the model synchronization function in the model transformation framework.

It is assumed that the application scenario of the quadruped robot is flexible and high-speed movement in a small complex terrain, and its movement posture is similar to the trotting gait of a horse. In this way, the design requirements of the quadruped robot are confirmed, and the main design requirements can be initially organized into the requirements diagram shown in Fig. 9. In the system modeling process, the requirements are further refined, and associated relationships are established with the structure, behavior, and other elements, such as the relationship "satisfaction" and "verification".

Based on these requirements, the quadruped robot is required to have corresponding motion functions, and the motion functions can be further decomposed into motion execution functions, energy supply functions, control functions, feedback functions, and physical support functions, which represent executing specific motions, provision of energy for motions, control of moving parts, acquisition of motion data and provision of structural support, respectively. In general, the components of the mechatronic system structure include the support part, energy part, control part, sensing part, and execution part. In this paper, with reference to this structural division, the functions of the quadruped robot are refined and reorganized to establish the corresponding logical architecture, which contains five subsystems: mechanical structure subsystem, energy subsystem, drive subsystem, control subsystem, and sensing subsystem. The mechanical structure subsystem contains the physical structure that provides support and actuates motion, the energy subsystem provides the energy required for system operation and performs initial regulation and distribution, the drive subsystem provides torque and speed to the actuator, the control subsystem provides control signals according to the system motion requirements and outputs, and the sensing subsystem is responsible for collecting data generated during the motion. The corresponding block definition diagram of the quadruped robot system model is shown in Fig. 10.

The quadruped robot needs to move in complex terrain, so its actuator is chosen to be a limb with a knee joint; due to the small range of motion and flexibility of the quadruped robot, its energy source is electric, powered by a high-power battery and driven by joint motors. In the corresponding system model, the drive module that contains the motors is connected to the power module, the control module, and the actuators in the mechanical structure module, whereas the sensing module is connected to both the actuators and the control module. With the advancement of the design, the requirements and functions are refined further and the system model is further expanded, and the physical architecture is established. At the completion of the physical architecture, the parameters of the physical components such as moving parts, power supply, motor, and control computer are determined, which provide the basis for the subsequent domain-specific design and device selection in the detailed design.

```
[begin/]
Dim the default variable; /* Declare default variables
Sub main()
Create the SolidWorks application process; /*Start SolidWorks process
[traverse block by composite relation in Postorder /*Traversing blocks by Postorder
  {if the block is a part property of a new <<prt>> block
    {/] Create and open a new part document; /* Create new part file and open
     Sort the block in <<prt>> block by boolean operation
    [}/] /*Sort the geometry within the part by Boolean operation type, with the elements bound
    to the same Boolean difference constraint arranged centrally and the subtracted geometry first
   InsertCoordinateSystembyCoordinate([coordinate /]);
   /*Insert the reference coordinate system according to the geometry position
   [if the block applied <<sphere>>
     {/]SelectXZplane and InsertSketch;
      /*Insert the sketch in the XZ plane of the reference coordinate system
      CreateSectionSemicircle ([origin_coordinate /], [radiu /], [datum_axis /]);
      /*Draw semicircular section with reference axis
      [if the block bind the parameter 'cut'
        {/]CreateFeatureRevolutionCut ( [datum_axis /]);[} /*Rotational excision to form a ball
       else
        {/]CreateFeatureRevolution ( [datum_axis /]);[} /*Rotation to form a ball
     }
   if the block applied <<revolution>>
     {/]SelectXZplane and InsertSketch; /*Insert the sketch in the XZ plane of the reference coordinate system
      CreateSectionSemicircle ([generatrix /], [datum_axis /]); /*Plot busbar and datum axis
      [if the block bind the parameter 'cut'
        {/]CreateFeatureRevolutionCut ( [datum_axis /]);[} /*Rotational excision to form a rotary body
       else
        {/]CreateFeatureRevolution ( [datum_axis /]);[} /*Rotation to form a rotary body
     }
   if the block applied <<cylinder>>
     {/]SelectXYplane and InsertSketch; /*Insert the sketch in the XY plane of the reference coordinate system
      CreateSectionCircle ([origin_coordinate /], [radiu /]); /*Draw a circular section
      [if the block bind the parameter 'cut'
        {/] CreateFeatureExtrusionCut ( [height /], [direction_coordinate /]);[}
         /*Stretch and cut to form a cylinder
       else
        {/] CreateFeatureExtrusion ( [height /], [direction_coordinate /]);[} /*Stretch to form a cylinder
     }
   if the block applied <<prism>>
     {/]SelectXYplane and InsertSketch; /*Insert the sketch in the XY plane of the reference coordinate system
      CreateSectionPolygon ([vertex_coordinate /]); /*Draw the polygonal section
      [if the block bind the parameter 'cut'
        {/] CreateFeatureExtrusionCut ( [height /], [direction_coordinate /]);[}
         /*Stretch and cut to form a prism
       else
        {/]CreateFeatureExtrusion ( [height /], [direction_coordinate /]);[} /*Stretch to form a prism
     }
   if the block is a part property of a new <<asm>> block
     {/]Create and open a new assembly document; /*Create a new assembly file and open it
      [for each <<prt>>block
        {/]AddComponent([part document/]);[} /* Add components
      [for each constraint
        {/]AddMate([related geometry element of component/], [constraint_name/]); [}/*Add fits
     }
  }/]
Save Model /*Save the model file
 [exit/]
```

Fig. 7. A part of pseudo modeling code.

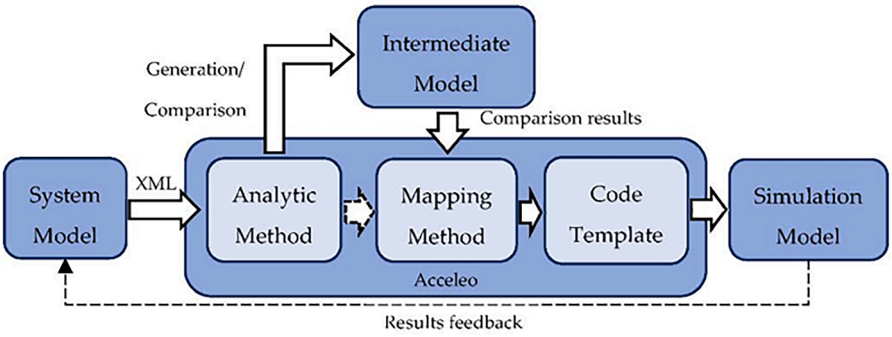


Fig. 8. Model transformation and synchronization mode.

5.2. Synchronous transformation and analysis

During the logical architecture phase, initial design parameters and constraints are established. It is imperative to validate whether these parameters meet the constraints or can be optimized. This verification process helps uncover potential issues in the early design phase, reducing the workload in subsequent stages and enhancing overall design efficiency. Simulation tools are predominantly used to analyze

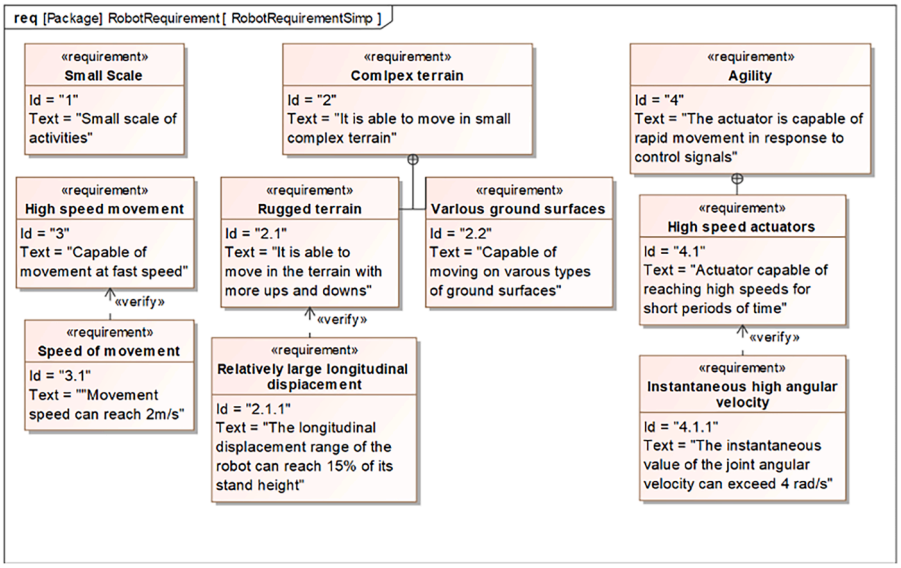


Fig. 9. Requirement diagram for the quadruped robot.

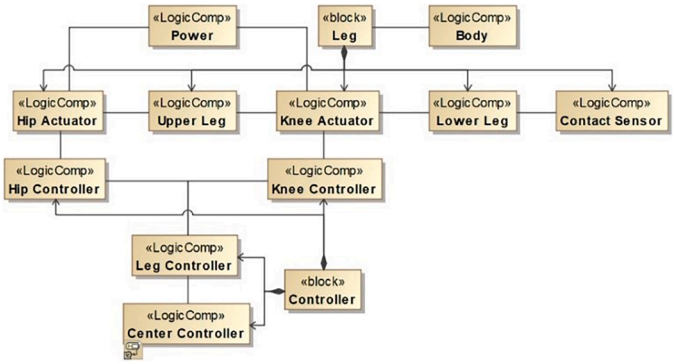


Fig. 10. Block definition diagram of the quadruped robot.

and validate these parameters. The transformation framework outlined in this paper can convert the entire logical architecture of the system into a simulation model, aiding in the analysis of parameters within the architecture.

Upon integrating the content of the logical architecture into a general block, the model is exported as an XML file. The corresponding modeling script is then generated using the transformation framework. By executing the script file in MATLAB, the corresponding simulation model is generated. As an illustration, the transformation process of the control subsystem, alongside the related model and snippets of code, is depicted in Fig. 11. The content of the logical architecture in the system model is delineated by the System Composer architecture model, with parameters encapsulated within stereotypes of the architecture model.

The system model can be transformed and analyzed in its entirety within the synchronous integration framework in the logical architecture phase, and after the logical architecture of the quadruped robot is verified and optimized, the physical architecture can be further designed on this basis. This process involves modification, expansion, and validation of the system model and synchronous update of the corresponding simulation model and geometric model. After the physical architecture of the system model is constructed, the physical simulation model of the quadruped robot can be obtained through synchronization.

In the process of synchronous transformation, the geometric information in the physical architecture is transformed into the CAD model through the transformation framework, and the adapted SysML profiles and mapping relationships are more concise in the transformation. The transformed CAD model can be visually represented in SolidWorks and analyzed by interference checking and other functions of SolidWorks. In addition, the CAD model can be imported into Simulink and connected to the Sim-scape Multibody module to transfer geometric information, which eliminates the need to remodel with complex modeling semantics. The geometric information in the physical architecture of the quadruped robot, the transformed CAD model, the Simscape Multibody model, and their transformation relationships are shown in Fig. 12.

From the design requirement of flexible motion of the quadruped robot, the motor is needed to provide high torque and angular velocity to the hip joint in a short period. After the import of the CAD model into Simulink, the variation of the angular velocity of the hip joint of the quadruped robot can be obtained by the simulation operation (Fig. 13), which shows that the instantaneous value of the angular velocity can be close to 6 rad/s, therefore the requirement of flexible motion is satisfied.

According to the requirement of the quadruped robot to adapt to the rugged terrain, the vertical displacement range of the body needs to reach 20% of the standing height when the quadruped robot is running stably. Meanwhile, refinement of the requirement for high-speed movement of the quadruped robot leads to a movement speed close to 2 m/s. Reliance on the detection module of the body position of the simulation model, the displacement and velocity information of the body of the simulation model in operation can be obtained. The longitudinal displacement and the velocity change in the horizontal and numerical directions are shown in Fig. 14. The longitudinal displacement range of the robot is greater than 20% of the standing height, and the horizontal movement speed is close to 2 m/s, therefore requirements of adaptation to rugged terrain and high-speed movement are satisfied.

Based on the results of such simulations, modifications, and adjustments can be carried out to various aspects of the system model parameters to complete the verification and optimization. The smooth implementation of the model generation and synchronization process above validates the role of the applied integrated design methodology and the synchronization integration framework.

6. Conclusion

Based on the problems of insufficient parallelism between the simulation of the entire system and the system modeling of the mechatronic system, and the relative complexity of geometric modeling and its integration functions, an integrated design method for the mechatronic system is studied in this paper, and a model integration

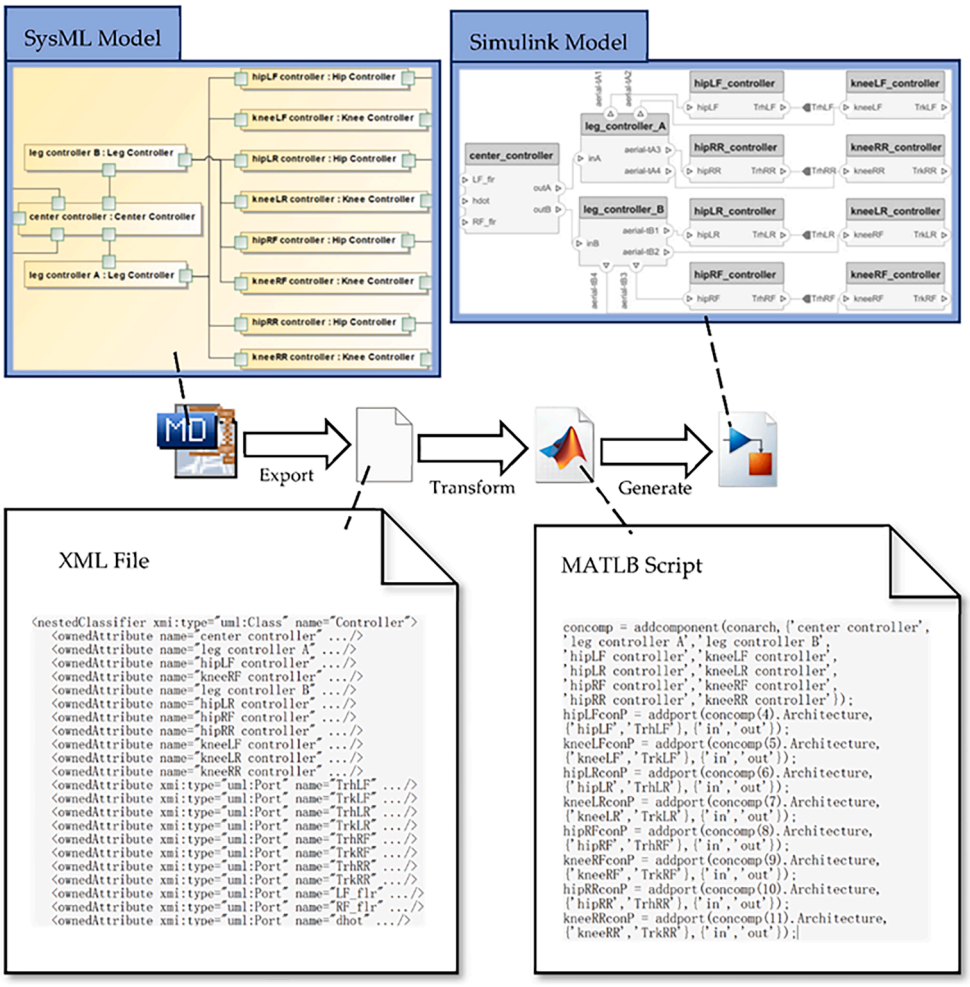


Fig. 11. Transformation process for the logical architecture of the quadruped robot based on the synchronous transformation framework.

framework for the support of the integrated design is constructed. These works provide effective support for integrated design and simulation methods, and the consistency and traceability of information in the design process are further ensured, and the number of design iterations is effectively reduced and the efficiency is improved.

Specifically, a synchronous model integration framework that covers the SysML model, the simulation model, and the geometric model is constructed by Acceleo [42], and the functionality of the integration framework and the feasibility of the integrated design method are verified by an example of a quadruped robot design process. The framework extracts the required system model information by mapping rules, generates intermediate models for comparison, and modeling or updating codes, which enables the simulation verification part of the design process in parallel with the system architecture design part. For geometric information in the system model, SysML profiles that express geometric semantics are constructed based on TTRS theory, combined with the geometric design requirements in system modeling, and SolidWorks is selected as the integrated object for the mapping relationships. These profiles and mapping relation-ships reduce the difficulty of geometric modeling and geometric information integration in the conceptual design stage, enable the geometric information to be expressed visually in CAD tools, and enable the import of system simulation models from CAD tools to reduce redundant modeling and enrich simulation models.

However, even if the MBSE method is applied for integrated design, there are still several aspects that need to be further investigated. For example, the scope of the construction and integration of mapping relationships in this paper is minor, the interaction model in the system simulation model is not involved, and the simulation of domain-specific models is not integrated, so the further expansion of SysML semantics and mapping relationships, and the direct or indirect integration between system simulation and domain-specific simulation can be considered; with the integration of multiple models, SysML profiles become more and more redundant and cause difficulties in the development and operation of the integration tools, so it is necessary to further streamline the extension semantics, such as the construction of a common format with simulation models, geometric models or unified meta-model semantic extensions, optimization and specification of the algorithm for the specific process of model conversion, etc. This is a planar quadruped motion robot that did not consider rolling during design. The introduction of external conditions, analysis of motion stability, and improvement of corresponding parameters will be further improved in the future.

Funding

This work was supported by Open Fund of State Key Laboratory of Digital Manufacturing Equipment and Technology of China (Grant number: DMETKF2022015).

CRediT authorship contribution statement

Chu Changyong: Writing – original draft, Software, Project administration, Methodology, Data curation, Conceptualization. **Zhang**

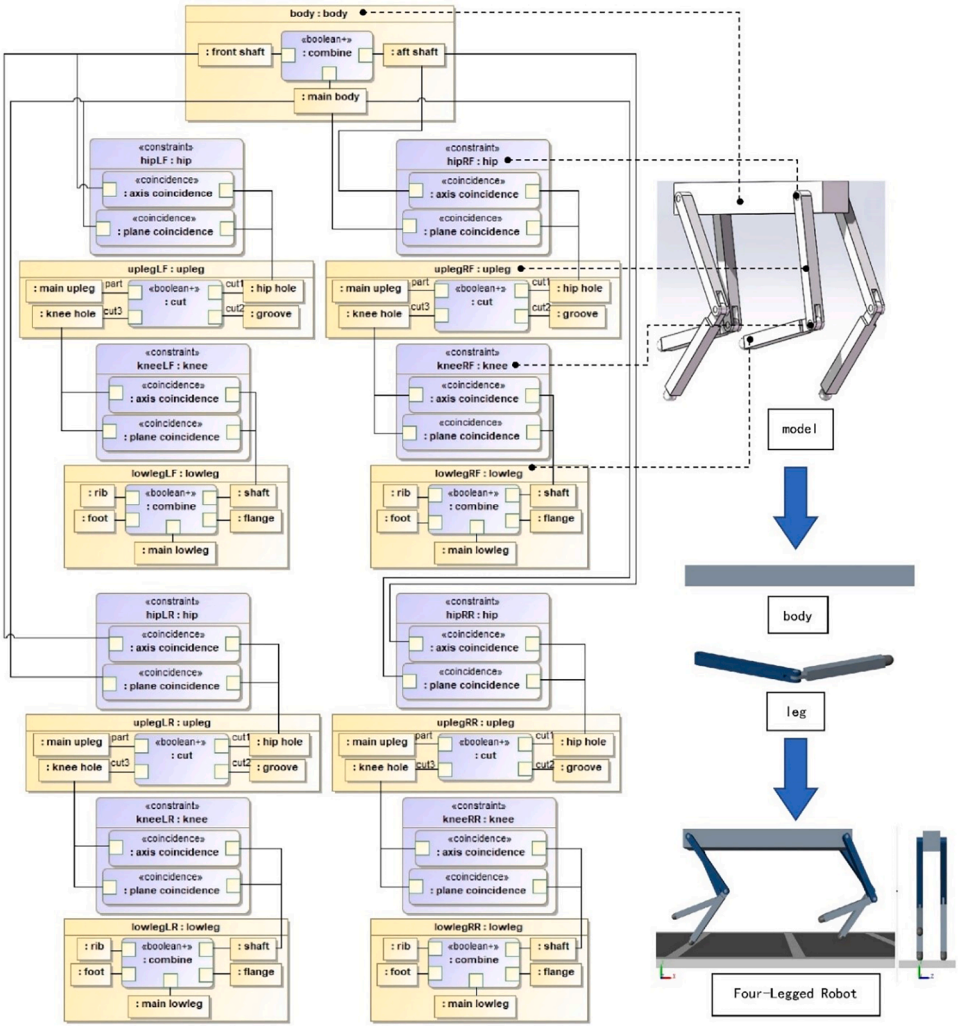


Fig. 12. Transformation process of geometric model.

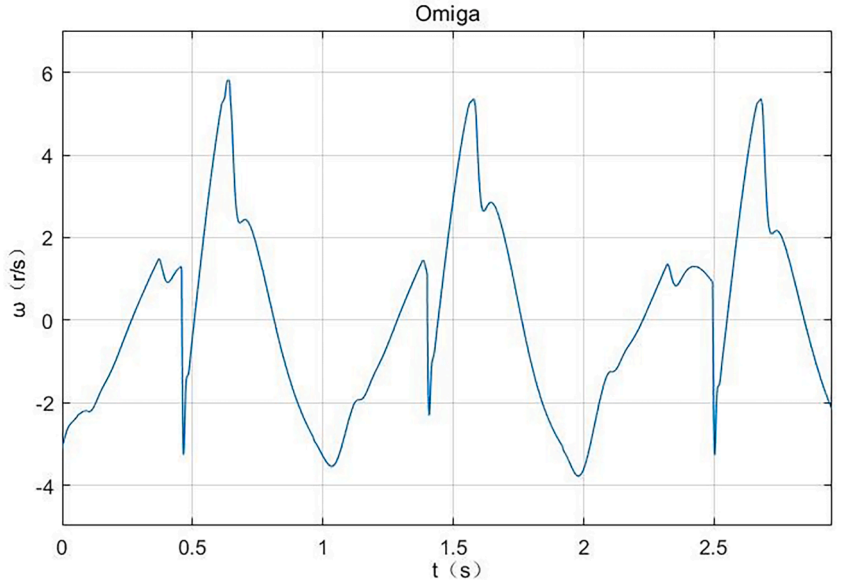


Fig. 13. Variation of angular velocity of hip joint.

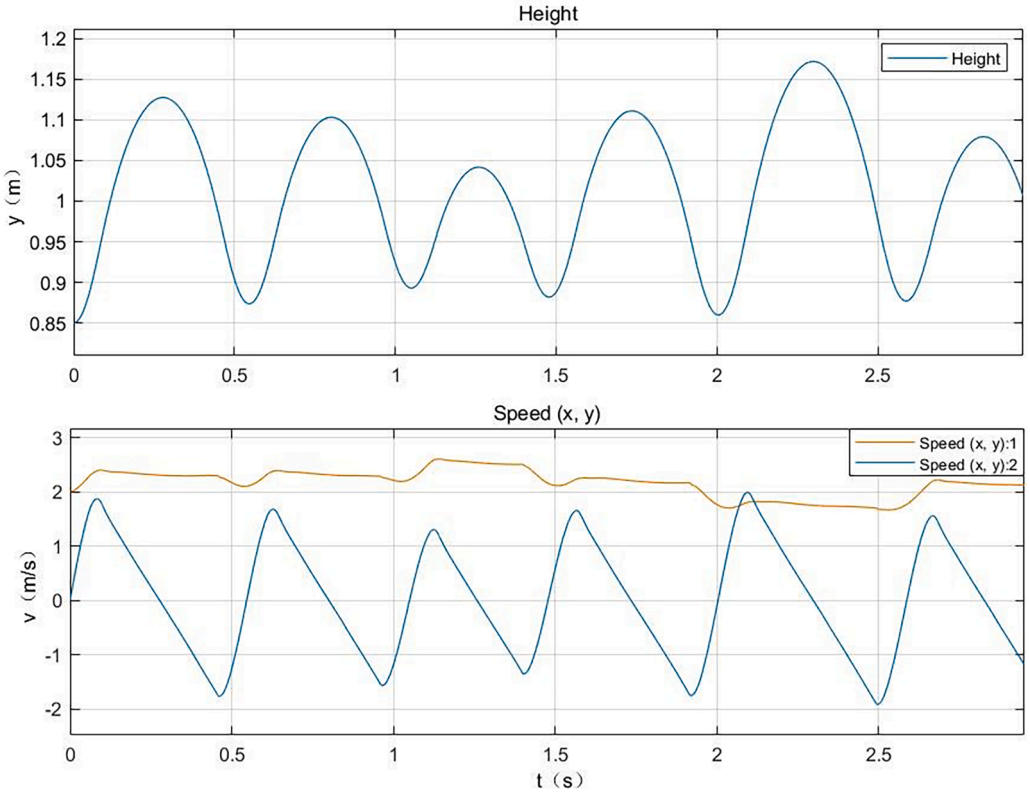


Fig. 14. Longitudinal displacement of the robot body and its lateral and longitudinal velocities.

Chunjia: Writing – review & editing, Supervision, Software, Methodology, Formal analysis, Data curation. **Yin Chengfang:** Writing – review & editing, Validation, Project administration, Methodology, Formal analysis.

Declaration of competing interest

The author(s) declared no potential conficts of interest with respect to the research, author- ship, and/or publication of this article.

Data availability

The authors do not have permission to share data.

References

[1] Harashima F, Tomizuka M, Mechatronics Fukuda T. What is it, why, and how?" an editorial[J]. IEEE/ASME Trans Mechatron 1996;1(1):1–4. https://doi.org/10.1109/TMECH.1996.7827930.
[2] Tomiyama T, Amelio VD, Urbanic J, et al. Complexity of multi-disciplinary design [J]. CIRP Annals - Manuf Technol 2007;56(1):185–8. https://doi.org/10.1016/j.cirp.2007.05.044.
[3] Rahman MAA, Mizukawa M. Model-Based Development and Simulation for Robotic Systems with SysML, Simulink and Simscape Profiles[J]. Int J Adv Robot Syst 2012;10(2):112–3. https://doi.org/10.5772/55533.
[4] Object Management Group. OMG Systems Modeling Language Specification. [EB/OL]. [2017-05-01]. https://www.omg.org/spec/SysML/1.5/PDF.
[5] Paredis C, Johnson T. Using OMG' s SysML to support simulation[C]. In: Simulation Conference. IEEE; 2008.
[6] Zhang L, Ye F, **e K, et al. An integrated intelligent modeling and simulation language for model-based systems engineering[J]. J Ind Inf Integr 2022;28:100347.
[7] Zhang X, Wu B, Zhang X, et al. An effective MBSE approach for constructing industrial robot digital twin system[J]. Robot Comput Integr Manuf 2023;80:102455.
[8] Hu X, Arista R, Lentes J, et al. Ontology-centric industrial requirements validation for aircraft assembly system design[J]. IFAC-PapersOnLine 2022;55(10):3016–21.
[9] Moser T, Biffl S. Semantic Integration of Software and Systems Engineering Environments[J]. IEEE Trans Systems Man Cybernetics Part C 2011;42(1):38–50. https://doi.org/10.1109/TSMCC.2011.2136377.
[10] Johnson TA. Integrating models and simulations of continuous dynamic system behavior into SysML[J]. Traffic Safety & Nuisance Research Institute Rpt 2008. DOI: TJohnson_MS_Thesis_08-04-24 (psu.edu).
[11] Berkenköter K, Bisanz S, Hannemann U, et al. The HybridUML profile for UML 2.0 [J]. International Journal on Software Tools for Technology Transfer (STTT) 2006;8(2):167–76. https://doi.org/10.1007/s10009-005-0211-z.
[12] Friedenthal S, Moore A. A Practical Guide to SysML: the Systems Modeling Language[M]. A Practical Guide to SysML:The Systems Modeling Language | Guide books (acm.org). Waltham: Morgan Kaufmann; 2012. p. 15–29.
[13] Object Management Group. MDA Guide[EB/OL]. [2014-06-01]. https://www.omg.org/cgi-bin/doc?ormsc/14-06-01.pdf.
[14] Nikolaidou M, Kapos GD, Dalakas V, et al. Basic guidelines for simulating SysML models: an experience report[C]. In: International Conference on System of Systems Engineering(SoSE). IEEE; 2012. p. 95–100. https://doi.org/10.1109/SYSoSE.2012.6384172.
[15] Johnson TA, Jobe JM, Paredis C, et al. Modeling Continuous System Dynamics in SysML[C]. In: ASME International Mechanical Engineering Congress & Exposition; 2007. https://doi.org/10.1115/IMECE2007-42754.
[16] Johnson T, Kerzhner A, Paredis C, et al. Integrating Models and Simulations of Continuous Dynamics Into SysML[J]. Journal of Computing & Information Science in Engineering 2012;38(1):122–9. https://doi.org/10.1115/1.4005452.
[17] Csertán G, Huszerl G, Majzik I, et al. VIATRA - visual automated transformations for formal verification and validation of UML models[C]. In: IEEE International Conference on Automated Software Engineering. IEEE; 2002. https://doi.org/10.1109/ASE.2002.1115027.
[18] Schumacher T, Inkermann D. Heterogeneous models to support interdisciplinary engineering-map** model elements of SysML and CAD[J]. Procedia CIRP 2022;109:653–8.
[19] Cao Y, Liu Y, Paredis C. Integration of System-Level Design and Analysis Models of Mechatronic System Behavior Based on SysML and Simscape[C]. In: ASME International Design Engineering Technical Conferences & Computers & Information in Engineering Conference. DOI; 2010. https://doi.org/10.1115/DETC2010-28213.
[20] Cao Y, Liu Y, Paredis C. System-level model integration of design and simulation for mechatronic systems based on SysML[J]. Mechatronics 2011;21(6):1063–75. https://doi.org/10.1016/j.mechatronics.2011.05.003.
[21] Fu C, Liu J, Yu HY, et al. A Visual transformation method of SysML model to Modelica model[C]. //. In: Journal of Physics: Conference Series. IOP Publishing. 1684; 2020, 012058.
[22] Li Q, Cao Z, Huang T. Modeling hybrid systems based on combination of SysML and Modelica[C]. //. In: International Conference on Computer Network Security and Software Engineering (CNSSE 2022). SPIE. 12290; 2022. p. 151–60.

[23] SysML-Modelica integration working group. SysML Modelica transformation specification [EB/OL]. [2011-08-22]. https://www.omgwiki.org/OMGSysML/lib/exe/fetch.php?media=sysml-modelica:sysml-modelica_transformation_ftf_report_-_ptc2011-08-13_-_09-12-2011.pdf.
[24] Object Management Group. SysML Extension for Physical Interaction and Signal Flow Simulation[EB/OL]. [2021-05-03]. https://www.omg.org/spec/SysPhS/1.1/PDF.
[25] Habib T, Komoto H. Comparative analysis of design concepts of mechatronics systems with a CAD tool for system architecting[J]. Mechatronics 2014;24(7):788–804. https://doi.org/10.1016/j.mechatronics.2014.03.003.
[26] Barbedienne R, Penas O, Choley JY, et al. TheReSE: sysML extension for thermal modeling[C]. In: 9th Annual IEEE International, Systems Conference (SysCon); 2015. p. 301–8. https://doi.org/10.1109/SYSCON.2015.7116768. DOI.
[27] Estefan J.A. INCOSE MBSE Initiative Survey of Model-Based Systems Engineering (MBSE) Methodologies[J]. 2011, 34(12): 78–93. Microsoft Word - MBSE_Methodology_Survey_RevA.doc (psu.edu).
[28] Shah JJ, Rogers, et al. Functional requirements and conceptual design of the feature-based modelling system[J]. Computer-Aided Engineering Journal 1988;5(1):9–15. https://doi.org/10.1049/cae.1988.0004.
[29] BöHnke D, Reichwein A, Rudolph S. Design Language for Airplane Geometries Using the Unified Modeling Language[C]. In: ASME International Design Engineering Technical Conferences & Computers & Information in Engineering Conference; 2010. https://doi.org/10.1115/DETC2009-87368. DOI.
[30] Barbedienne R, Penas O, Choley JY, et al. Introduction of geometrical contraints modeling in SysML for mechatronic design[C]. In: Mecatronics. IEEE; 2015. p. 145–50. https://doi.org/10.1109/MECATRONICS.2014.7018580.
[31] Barbedienne R, Penas O, Choley J, et al. Modeling Framework for a Consistent Integration of Geometry Knowledge During Conceptual Design[J]. J Comput Inf Sci Eng 2019;19(2):021009. https://doi.org/10.1115/1.4042551.
[32] Plateaux Régis, Penas O, Barbedienne R, et al. Use of technologically and topologically related surfaces (TTRS) geometrical theory for mechatronic design ontology[J]. Comput Aided Des Appl 2017;14(5):595–609. https://doi.org/10.1080/16864360.2017.1280270.
[33] Qamar A, Wikander J, During C. Managing dependencies in mechatronic design: a case study on dependency management between mechanical design and system design[J]. Eng Comput 2015;31(3):631–46. https://doi.org/10.1007/s00366-014-0366-x.
[34] Eclipse Foundation. ATL – a model transformation technology[EB/OL]. [2021-12-10]. https://wiki.eclipse.org/MMT/ATL_Transformation_Language_(ATL).
[35] Sdius. Cameo Workbench. [CP/DK]. [2016-8-3]. https://www.slideshare.net/sodius7789/cameo-workbench.
[36] Graignic P, Vosgien T, Jankovic M, et al. Complex System Simulation: proposition of a MBSE Framework for Design-Analysis Integration[J]. Procedia Comput Sci 2013;16(1):59–68. https://doi.org/10.1016/j.procs.2013.01.007.
[37] Gielingh W. An assessment of the current state of product data technologies[J]. Computer-Aided Design 2008;40(7):750–9. https://doi.org/10.1016/j.cad.2008.06.003.
[38] Vallecillo A. On the Combination of Domain Specific Modeling Languages[C]. In: European Conference on Modelling Foundations and Applications. Berlin: Springer; 2010. p. 305–20. https://doi.org/10.1007/978-3-642-13595-8_24.
[39] Li W. An Internet-enabled integrated system for co-design and concurrent engineering[J]. Comput Ind 2004;55(1):87–103. https://doi.org/10.1016/j.compind.2003.10.010.
[40] Qin SF, Harrison R, West AA, et al. A framework of web-based conceptual design [J]. Comput Ind 2003;50(2):153–64. https://doi.org/10.1016/S0166-3615(02)00117-3.
[41] Yang Z, Du H, Liu Y, et al. Use the Harmony-SE Approach to Extend the Advantages of MBSE[C]. In: 16th Conference on Industrial Electronics and Applications (ICIEA). IEEE; 2021. https://doi.org/10.1109/ICIEA51954.2021.9516269.
[42] Eclipse Foundation. Acceleo - generate anything from any EMF model[CP/DK]. [2022-12-25]. https://wiki.eclipse.org/Acceleo.

Chu Changyong received the PhD degree from Nanyang Technological University in Singapore. He was a visiting scholar at Concordia University in Canada from July 2017 to August 2018. Now, he is an associate professor in the School of Mechanical Engineering, Hangzhou Dianzi University.

His-research focuses on model-based systems engineering MBSE and product lifecycle management PLM/PDM.

Zhang Chunjia received his Bachelor of Engineering in 2023 from Qingdao University of Technology. He is currently a master's student in the School of Mechanical Engineering at Hangzhou Dianzi University.

His-research focuses on model-based systems engineering MBSE.

Yin Chengfang received a Master's degree in Mechanical Engineering from Hangzhou Dianzi University in 2023.

His-research focuses on model-based systems engineering MBSE.