复杂机电系统的软件与物理统一的形式化功能分析

曹 悦[1*] 刘玉生[2] 秦绪佳[1] 汤 颖[1]
1.浙江工业大学计算机科学与技术学院,杭州,310023
2.浙江大学计算机辅助设计与图形学全国重点实验室,杭州,310058

摘要:复杂机电系统的功能分析长期以来主要关注物理子系统实现的连续物理变换,忽略了软件子系统控制的物理过程之间的复杂执行顺序。针对这一挑战,提出了一种软件与物理统一的形式化功能表征与分析方法。对基于流的功能表示方法进行扩展,形成软件与物理统一的形式化功能表征。在此基础上,提出一种基于规则的功能分解方法,支持软件与物理混合功能的自动分解。以移动机器人系统为例,展示了软件与物理统一的功能分析过程。

关键词:基于模型的系统工程;概念设计;机电一体化;功能分析
中图分类号:U463.212
DOI:10.3969/j.issn.1004-132X.2025.02.007 **开放科学(资源服务)标识码(OSID)**:

Software-physical Unified Formal Functional Analysis for Complex Mechatronic Systems

CAO Yue[1*] LIU Yusheng[2] QIN Xujia[1] TANG Ying[1]
1.College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, 310023
2.State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, 310058

Abstract: Functional analysis of complex mechatronic systems focused on the continuous physical transformations achieved by physical subsystems, and ignored the complex execution sequences among the physical processes controlled by software subsystems. In response to this challenge, a software-physical integrated formal functional representation and analysis method was proposed. First, the flow-based functional representation was extended to form a unified formal functional representation. Then, a rule-based function decomposition method was proposed to support the automatic decomposition of software-physical hybrid functions. A mobile robot system was used as a case to illustrate the proposed software-physical unified functional analysis processes.

Key words: model-based systems engineering; conceptual design; mechatronics; functional analysis

0 引言

功能分析是复杂系统早期概念设计的重要环节,其目的是从自然语言描述的系统功能需求中抽取结构化的系统功能。该过程通过对功能进行形式化、规范化的建模,以及笼统模糊功能的逐渐分解细化,形成可直接用于系统设计集成的子功能结构。功能分析主要包括功能表示与功能分解两个领域。

功能表示主要研究功能的形式化表征。在众多的形式化功能表征中,最主流的方法为基于流的功能表示[1-2]。这种方法将系统的预期行为即功能视为系统对流施加的转换。在此基础上,HIRTZ 等[3]给出流及转换的标准化定义,即功能基(functional basis)。这种流分类方法在本质上是以流的物理特征为分类依据。此外,一些研究人员尝试采用更为形式化的方法对功能的操作语义进行说明。YUAN 等[4-5]提出功能效应(functional effect)的概念,通过定性描述功能操作的流属性的初始值、结束值和变化趋势来描述功能产生的效果。CHEN 等[6]通过对流属性施加约束来描述功能对流施加的作用。

功能分解的目的是将复杂的功能分解为相对简单、便于实现的子功能,具体可分为自底向上和自顶向下两类。自底向上的功能分解主要以待分解目标功能的输入输出流为基准,通过对知识库中的子功能进行查找及组合连接来生成符合目标功能接口的功能结构。SRIDHARAN 等[7]基于流类型匹配及文法规则生成总功能的内部结构。SANGELKAR 等[8]利用图挖掘算法从知识库中获取功能分解规则,并将这些规则应用到具体的功能分解问题。CHEN 等[9]基于流接口特征匹配实现对总功能的分解。自顶向下的功能分解基

收稿日期:2024-01-24
基金项目:国家自然科学基金(62102367);浙江省自然科学基金(LQ22F020019,LZ23F020010)

于待分解的目标功能的输入流和输出流的属性差异，推断实现功能的可能物理原理组合，从而实现功能分解。MALMQVIST[10]采用功能方法树为目标功能寻找实现方法并从实现方法中提取多组子功能。UMEDA 等[11]采用 FBS（function-behavior-state）功能模型，基于总功能的输入输出流的状态改变来实现任务的拆分。YUAN 等[4]采用定性表示及定性推理实现功能的自动分解。CHEN 等[12]采用基于仿真的检索方法查找功能的解决方案。GUI 等[13]提出一种针对上下文感知系统的功能分解方法。

从上述研究可知，当前的功能表示及分解方法主要关注物理子系统实现的连续物理过程，忽略了软件子系统承担的主要功能——物理过程之间复杂的执行顺序。事实上，复杂机电系统是软件与物理子系统的有机集成[14]，而物理子系统与软件承担的功能差异巨大。更为重要的是，软件与物理子系统设计之间可能存在隐蔽但巨大的影响，如果不能在早期功能分析中识别并处理这些潜在的设计影响，则可能将设计缺陷带入后期的详细设计，造成项目的返工甚至失败。因此，如何在早期设计中统一分析软件与物理子系统功能是复杂机电系统设计的重要挑战。

针对这一困难，本文提出一种面向复杂机电系统的软件与物理子系统统一的形式化功能表征与分析方法。首先，基于流的功能表示并针对软件功能特点引入时序逻辑相关表示元素，形成软件与物理统一的形式化功能表征。之后，提出一种针对混合功能的自动化功能分解方法。该方法能根据功能的形式化表征判断功能类型，并基于混合功能的层次化语义模型实现混合功能的自动分解。

1 方法概述

为实现从顶层功能到底层功能的结构分析，本文提出软件与物理融合功能的分析方法框架，如图 1 所示，该框架由两部分组成：

1）软件与物理的统一形式化功能语义表征。文字化的功能需求描述无法直接用于设计，需要对功能指定的预期行为进行形式化描述以驱动设计的开展。传统的功能表示方法仅能描述物理子系统的功能，忽略了软件子系统对执行过程的协调。本文方法结合软件与物理功能的不同特征，对基于流的功能表示进行扩展，提出新的流分类方法及层次化的操作语义表征。

2）面向混合功能的功能自动分解。系统功能具有软件与物理相混合的特征，因此以物理为核心的传统功能分解方法无法直接使用。本文在形式化功能语义表征的基础上，首先对功能进行分类，识别出物理功能、软件功能和混合功能。之后，针对混合功能给出相应的自动分解策略，通过自顶向下迭代开展功能分解，产生直接用于设计集成的底层功能结构。

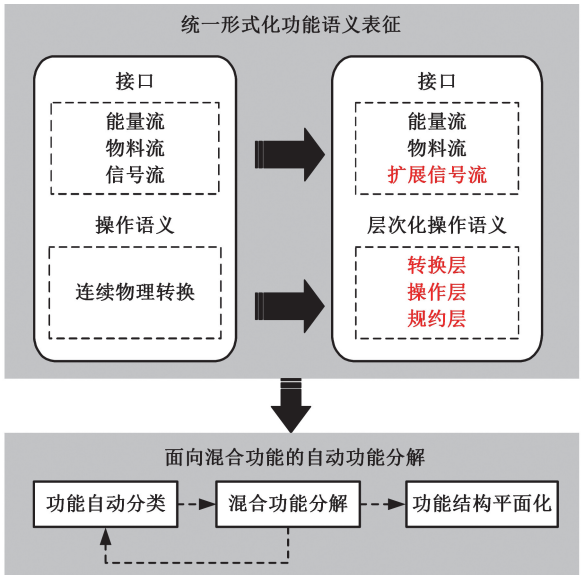


图 1 软件与物理融合功能分析方法概述

Fig.1 Method overview of the software-physical integrated functional analysis

需要说明的是，软件根据其所实现的控制功能在复杂机电系统中充当不同的角色。复杂机电系统的控制通常由多个层次的控制器实现[15-16]，其中，低层控制器用于调节连续的物理过程，如闭环或开环控制；中层控制器协调多个连续过程的执行，实现逻辑控制和顺序控制；高层控制器主要完成系统全局管理的相关功能，如计划、监视、诊断等。目前已存在大量低层控制器与物理子系统协同功能分析的相关研究，而高层控制对系统早期概念设计的影响相对较小，因此，本文方法研究的软件特指中层控制部分。

2 统一形式化功能表征

基于流的功能表示是形式化功能表征的主流方法之一。该方法将功能视为流的转换，但从该方法对流的分类和对语义的描述可以看出，它主要针对物理系统的功能而无法表达软件功能。

复杂机电系统中的软件部分与物理部分具有截然不同的功能特征。一方面，软件处理的主要对象是具有不同语义的信息，这必须体现在流的分类中；另一方面，软件的主要功能是协调多个连续物理过程的执行流程，软件功能的操作语义必

须能表征物理过程之间的逻辑及时序关系。因此，本文从流分类与操作语义两个方面对基于流的功能表示方法进行扩展，使其能统一表征复杂机电系统的软件与物理功能语义。

2.1　系统接口描述扩展

当前，工程设计领域公认的流分类标准由功能基[3]定义，它将系统处理的流分为物料、能量和信号，并定义流的二级子类和三级子类。信号流与其他两种流的不同之处在于它同时具有物理与信息的特征，即信号由物料或能量承载，同时，信号承载了具有不同语义的信息。信号的物料或能量体现信号的物理特征，信号的语义体现信号的信息属性。信号的信息属性却没有在当前的标准流分类中得到体现。

软件工程中，系统的处理对象通常分为数据流和控制流，前者表示系统执行动作的输入输出数据，后者表示激活动作的控制令牌[17]。这两种流均通过信号承载，体现了信号的信息属性。因此，本文将功能基(functional basis)中的流分类方法与软件工程中的流分类进行融合，为信号流定义一个新的流属性——信息类型(information type)。信息类型 data、control 分别对应信息属性“数据”与“控制”。

2.2　层次化功能语义表征

复杂机电系统的功能包含连续物理转换，而物理转换之间存在复杂的执行逻辑。物理转换的执行需要遵循一定的顺序。多个物理转换可根据控制模式按照不同的顺序执行。为描述执行顺序和控制模式这类复杂的功能语义，本文扩展转换的概念，添加描述转换之间时序及执行逻辑的操作层和规约层，形成具有三层结构的层次化语义表征方法。

2.2.1　转换层

转换层 $T=\{t_1,t_2,\cdots,t_n\}$ 包含系统需要执行的一系列基本物理过程或计算过程。根据执行操作对象的不同，将转换分为物理转换(physical transformation)和计算转换(computational transformation)。物理转换表示改变操作对象(物料流和能量流)的物理属性。计算转换的操作对象是信号流，表示对数据信号的处理或逻辑判断等。此外，在系统设计早期中，由于对系统的了解十分有限，操作的对象可能同时包含多种流，因此，除了上述两种基本转换外，本文还提出复杂转换(complex transformation)的概念，以表示语义尚不明确的转换。

已有的功能表示方法已对物理转换提供了许多计算机可理解的描述方法。这些表示方法的共同思想是，将物理转换视为对流的属性值的改变。例如，功能“加热气体”的操作语义描述为将“气体”的“温度”从“低”升高为“高”，其中，“气体”是转换的对象流，“温度”是流属性，“低”和“高”是属性的定性值，“升高”是属性值的变化趋势。这种基于流的功能表示是一种与具体实现方法无关的描述方法，仅表示客观、可观察的改变，不涉及实现这一改变的具体方式，如物理效应、软件算法等。功能效应不但能支持物理功能的自动分解，而且提供了基于系统建模语言(systems modeling language，SysML)的图形化描述手段[4]，因此，本文选取功能效应作为物理转换的具体表示方法。

计算转换也可视为对流的改变，但由于计算过程的复杂性与多变性，很难找到一个类似于功能效应的形式化概念来表示计算转换的执行语义[18]，因此本文方法采用自然语言描述计算转换的语义。

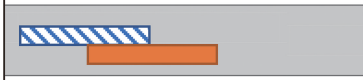
2.2.2　操作层

操作层 $P=\langle T,O_s,C\rangle$ 描述基本转换之间所有可能的执行序列。每个执行序列称为一个操作(operation)。若一个功能包含多个转换，则必须建立操作层来说明转换之间的执行序列。每个操作表示为转换通过时序算子及约束相连形成的表达式。时序算子集合 $O_s=\{\&,|,\cup\}$ 中，集合元素“$\&$”“$|$”“$\cup$”分别表示顺序、并发、无时序的执行方式。每个时序算子可以添加约束 c，以对时序关系进行进一步说明。

按照起止时间的不同，任意两个转换 t_1 和 t_2 之间的执行顺序有表 1 所示的 4 种情况。“$\&$”表示 t_2 在 t_1 结束之后才开始，即二者的执行过程不存在重叠。“$|$”表示 t_2 在 t_1 还未结束时就开始，即二者的执行过程存在重叠。c 对 t_1 的结束时间与 t_2 的开始时间的时差进行约束，例如，最典型的约束是直接指定 2 个时间点之间的时间间隔，如 $c=5$ s 表示时间间隔为 5 s。当 $c=0$ 时，可省略时序算子上的约束。

表 1　任意两转换之间的时序

Tab.1　Orders between any two transformations

执行顺序示例	时序表达式示例
	$t_1 \&_c t_2$
	$t_1 \& t_2$
	$t_1 \mid_c t_2$
	$t_1 \mid t_2$

2 个物理转换之间可能存在无时序关系的情况，即 2 个转换的执行受物理定律的驱动而非软件的控制。如吹风机的功能包含 3 个物理转换，其中，t_1 表示对空气进行加速，t_2 表示升高空气温度，t_3 表示引导空气流向。这 3 个物理转换的执行是由空气在不同物理机构之间的流动引起的，而不由软件发出控制信号决定，因此，该功能的操作表达式为 $t_1 \cup t_2 \cup t_3$。

2.2.3 规约层

复杂机电系统需根据不同的控制模式执行不同的目标行为即操作。控制模式的选择取决于系统接收到的控制或数据信号，因此，必须在系统的控制信号与操作之间建立因果关系。

时序逻辑常常用于描述反应系统（reactive system）应遵循的动态特性及需求规约[19-20]。在多种时序逻辑中，选取计算树逻辑（computational tree logic，CTL）作为规约层形式化基础的原因是：①CTL 主要用于描述有限状态系统的动态特性，而大部分软件密集型系统的行为范式均可抽象为有限状态系统[21]；② 已有模型检测工具广泛支持 CTL，因此可借助这些模型检测工具实现模型的自动推理。基于上述思想，本文基于 CTL 定义功能语义的规约层。

规约层 $S=\langle P_a, O_{\mathrm{CTL}}\rangle$ 描述了系统依据不同控制模式对操作的选择。每种控制模式通过 1 个 CTL 规约描述，每个规约由原子命题通过 CTL 算子连接而成。若 1 个功能具有多个操作，则必须通过 CTL 规约说明系统在何种条件下选择并执行各个操作。$P_a=P_s \cup P_e \cup P_c$ 表示 CTL 规约中的原子命题。P_s 为形如 $STATE=p$ 的状态命题集合时，若操作 p 正在执行，则状态命题 $STATE=p$ 为真，其中，$STATE$ 表示状态。P_e 为形如 $EVENT=e$ 的事件命题集合时，若事件 e 发生，则该命题为真。对于 P_c 为形如 $PROPERTY=v$ 的条件命题集合，属性 $PROPERTY$ 取值为 v 时，该命题为真。CTL 算子 $O_{\mathrm{CTL}}=O_t \cup O_q \cup O_l$ 中，时态算子 $O_t=\{F, G, X, U\}$ 修饰状态的时间关系，F 表示某未来状态，G 表示所有未来状态；路径量词 $O_q=\{A, E\}$ 修饰时间路径，A 表示对所有路径，E 表示存在一条路径；基本逻辑算子 O_l 由与 &、或 |、非 ¬、蕴含 → 等组成。

下面通过一个例子对上述形式化功能语义描述进行说明。某药生产线的涂层子系统主要负责对压制好的药片进行涂层，以改善口感或便于吞咽[15]。该子系统需要执行转料（对待涂层的药片进行转动）与喷料（向药片喷洒涂料）两个物理过程。两个物理过程的执行顺序由两个模式定义：模式 1 表示 2 个过程顺序执行，模式 2 表示 2 个过程并发执行。基于本文提出的层次化功能语义描述方法，对上述文字化的功能语义进行形式化描述。转换层包含的物理过程 t_r、t_s 分别表示转料和喷料；操作层包含的操作 $OP_1=t_r \& t_s$，$OP_2=t_r \mid t_s$ 分别表示两种执行模式；规约层由 2 条 CTL 规约组成：$AG(MODE=1 \rightarrow EF\ STATE=OP_1)$ 表示模式 1，$AG(MODE=2 \rightarrow EF\ STATE=OP_2)$ 表示模式 2，其中，$MODE$ 为表示模式的变量，$STATE$ 为系统状态，OP_1 和 OP_2 为操作层定义的操作。

3 面向混合功能的功能自动分解

引入层次化的功能表征后，系统功能不仅包含物理变换，还具有软件与物理相混合的复杂特征，以物理变换为核心的功能分解方式不再适用。为此，本文首先根据功能语义模型将功能分为物理功能、软件功能、混合功能，然后针对混合功能提出新的分解策略。

3.1 功能自动分类

根据功能的语义模型的层次结构将功能分为三类：

1）物理功能（physical function）是指仅由物理结构及连续控制器实现的功能，其语义模型仅有 1 个物理转换，不具有操作和 CTL 规约层。物理功能分解需基于功能效应开展。

2）软件功能（software function）是指仅由软件组件（含硬件运行环境）实现的功能，其语义模型仅有 1 个计算转换，不具有操作和 CTL 规约层。软件功能分解仅需考虑计算转换语义。

3）混合功能（hybrid function）是指同时需要物理及软件组件实现的功能，其语义模型同时具有操作层和 CTL 规约层。混合功能分解需要综合考虑操作顺序和 CTL 表达式。

基于上述方法可将功能语义以计算机可理解的形式表征，为功能的自动分类提供基础。根据上述三类功能语义模型可定义如下三条规则，实现功能的自动分类：

规则 1 $f \in F \wedge ((SIZE(f.OPS)=0) \wedge (SIZE(f.TRS_{cp})=1) \vee (SIZE(f.OPS) \geqslant 1)) \rightarrow f \in F_{\mathrm{h}}$

规则 2 $f \in F \wedge (SIZE(f.SPEC)=0) \wedge (SIZE(f.OPS)=0) \wedge (SIZE(f.TRS_p)=1) \rightarrow f \in F_{\mathrm{p}}$

规则 3　$f \in F \wedge (SIZE(f.SPEC)=0) \wedge (SIZE(f.OPS)=0) \wedge (SIZE(f.TRS_c)=1) \rightarrow f \in F_s$

其中，F_h、F_p、F_s 分别为混合功能、物理功能、软件功能的集合；f 为待分类的功能；$f.SPEC$ 为功能 f 语义模型中的 CTL 表达式，$f.OPS$ 为功能 f 语义模型中的操作；$f.TRS_{cp}$、$f.TRS_p$、$f.TRS_c$ 分别表示功能 f 语义模型包含的复杂转换、物理转换、计算转换，$SIZE$ 为集合中元素的数量。

3.2　混合功能分解

采用 1.2 节所述的功能分解方法对物理功能进行分解，软件功能的分解需要软件专业人员根据业务流程进行分析，因此，本文主要讨论混合功能的分解方法。

混合功能可根据其语义模型结构细分为图 2 所示的几种情况。

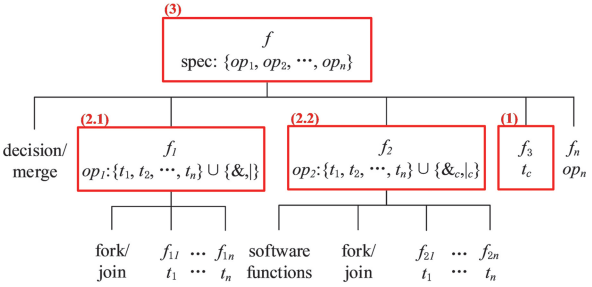


图 2　混合功能分解方式

Fig.2　Decomposition of hybrid functions

3.2.1　仅有转换层且转换层仅包含 1 个复杂转换的功能

对于仅有转换层且转换层仅包含 1 个复杂转换的功能，由于复杂转换的语义尚不明确，因此，首先对该功能进行语义分析，构建更为明确的语义模型，再进行功能分解。

3.2.2　具有操作层和转换层的功能

将具有操作层和转换层功能的语义模型表示为一个操作表达式，表达式的操作符为带约束的时序算子，操作数为转换。操作数中的每个转换都可识别出一个子功能，时序算子则需进一步讨论：

1）时序算子中的并发算子“|”表示 2 个转换同时发生。功能结构中，这一控制逻辑用并发节点表示。因此，每个并发算子需分解为 1 个分支节点(fork)及其集合节点(join)。

2）对于带约束的时序算子，其约束的判断需要相应的子功能实现，因此，这类算子需要额外创建软件功能来判断约束。

某功能语义模型的操作表达式为 $(t_1 \mid t_2)\&_{5s}t_3$。基于该表达式可识别出如下子功能：① 实现转换 t_1、t_2、t_3 的 f_1、f_2、f_3；② 表示 t_1 和 t_2 并发执行的分支节点 fork 和集合节点 join；③ 延时子功能 f_d 表示 f_1、f_2 执行完毕并延时 5 s 后，f_3 开始执行。

3.2.3　具有规约层、操作层和转换层的功能

CTL 规约主要用于判断模式，因此需引入 1 个判断节点(decision)及其合并节点(merge)来连接每个操作对应的路径。对规约涉及的每个操作按操作层分解方法进行分解。

上述功能分解过程涉及多个操作，操作使用的转换可能会有重复，因此，需对所有操作分解出的子功能集合求并集，形成底层子功能集合。

3.3　功能结构平面化

功能分解识别出的底层子功能集合需通过功能结构模型对底层子功能的执行顺序及交互信息进行进一步描述。由于功能分解过程逐层迭代开展，因此该过程产生的功能结构模型为一个层次化的图。这种层次化的图无法展示最底层叶子功能之间的直接连接关系，因而难以直接应用于设计集成。如图 3 所示，总功能 f 分解为子功能 f_1、f_2，f_1 进一步分解为 $f_{1,1}$、$f_{1,2}$，f_2 进一步分解为 $f_{2,1}$、$f_{2,2}$。分析过程中，建立的 3 个功能结构图分别记录了 f_1 和 f_2 之间的连接、$f_{1,1}$ 和 $f_{1,2}$ 之间的连接、$f_{2,1}$ 和 $f_{2,2}$ 之间的连接，却没有展示所有最底层子功能 $f_{1,1}$、$f_{1,2}$、$f_{2,1}$ 和 $f_{2,2}$ 之间的连接。最底层的功能结构是设计集成过程所需的，因此，还需对层次化的功能结构图进行“平面化”，将其展开为由最底层功能连接形成的功能结构模型。该模型被称为底层功能结构模型，是后续软件与物理协同正向设计的输入。

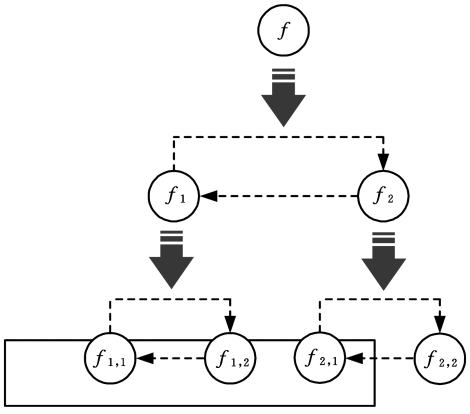
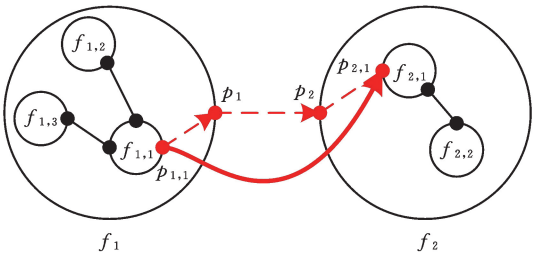


图 3　功能分解子图示例

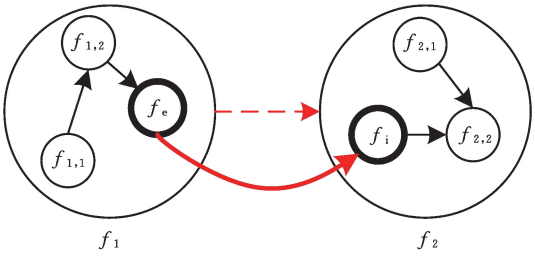
Fig.3　Sample functional decomposition subgraphs

平面化算法的基本思路：对功能分解形成的树形结构进行宽度优先遍历，对树形结构中的任意 2 个非叶子功能之间的流用这两个功能的子功能之间的流替换。对象流的替换方法如图 4a 所示，非叶子功能 f_1、f_2 之间有一条从接口 p_1 指向

接口 p_2 的对象流，接口 p_1、p_2 分别与子功能 f_{11} 的接口 p_{11}、f_{21} 的接口 p_{21} 相连，因此，可将非叶子功能之间的对象流替换为子功能的接口 p_{11} 到 p_{21} 的流。控制流的替换方法如图 4b 所示，非叶子功能 f_1、f_2 之间具有 1 条控制流，其中，f_e 为 f_1 的终止功能，f_i 为 f_2 的起始功能，则非叶子功能 f_1、f_2 之间的控制流可替换为其子功能 f_e 和 f_i 之间的流。



(a) 对象流替换



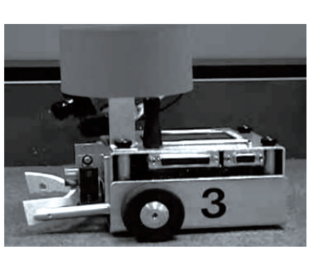
(b) 控制流替换

图 4　流替换方法

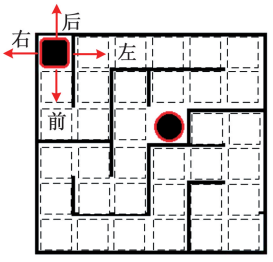
Fig.4　Flow replacement method

4　案例分析

本文以移动机器人系统[22]为例来说明软件与物理融合功能分析方法。图 5a 为该系统的概念图，系统需实现图 5b 所示的迷宫觅食任务，即机器人需要找到放置于迷宫中任意位置的 1 个铁罐。图 5b 中，方形表示机器人的起始位置，圆形表示待拾取的铁罐。



(a)移动机器人系统概念示例



(b)迷宫示意图

图 5　移动机器人系统的概念和任务

Fig.5　Concept and task of the mobile robot

机器人首先检测其四周环境，以识别其前后左右四个方向是否存在墙壁或罐子。之后，根据识别结果，机器人判断下一步探索的方向或准备拾取前方罐子。最后，根据判断结果，机器人根据指定方向移动一格或拾取罐子。该系统需求可采用 SysML 需求图(图 6)表示。根据上述需求，对系统逐层开展功能分析。

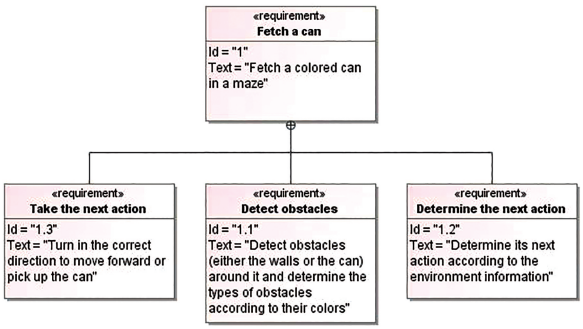


图 6　移动机器人系统需求

Fig.6　System requirements of mobile robot

4.1　迷宫觅食(FetchCan)的功能分析

迷宫觅食(FetchCan)功能被定义为构造型为《Function》的活动。对该功能的分析包括以下 3 个步骤：

1)功能语义建模。FetchCan 的语义模型如图 7 所示。功能语义包含操作层和转换层，其中，转换层由复杂转换障碍检测(DetectObstacble)和采取行动(TakeAction)，以及计算转换引导行动(GuideAction)组成，操作 Sequential3 表示 3 个转换顺序执行。

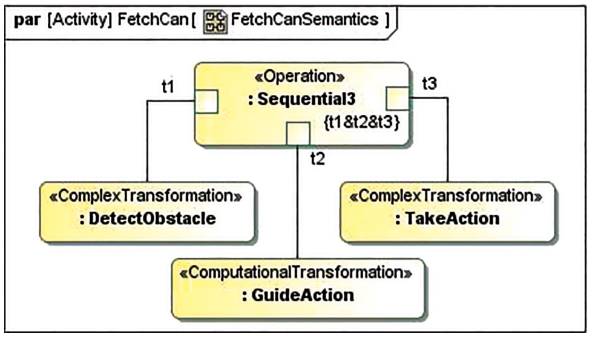


图 7　FetchCan 功能语义模型

Fig.7　Functional semantics model of FetchCan

2)功能分解。FetchCan 功能语义模型包含操作层，因此，可判断该功能为混合功能。根据功能操作表达式，将 FetchCan 分解为 3 个子功能，即 DetectObstacle、GuideAction、TakeAction。

3)功能结构建模。根据分解结果创建 3 个子功能，并将其实例化为 FetchCan 的组成部分。将 DetectObstacle、GuideAction 和 TakeAction 功能实例通过对象流与控制流相互连接，形成图 8 所示的 FetchCan 功能的结构模型。

4.2　采取行动(TakeAction)的功能分析

总功能分析完毕后，需要对识别出的每个子功能做进一步的功能分析。这里以采取行动(TakeAction)功能为例说明功能分析过程。

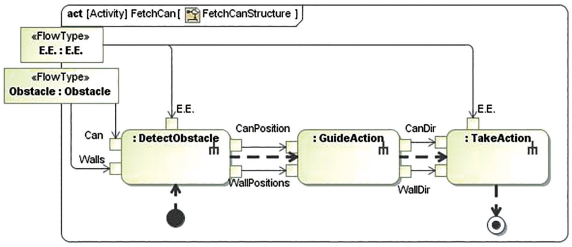


图 8　FetchCan 功能结构模型

Fig.8　Functional structure model of FetchCan

1)功能语义建模。TakeAction 功能语义的文字描述为:移动机器人系统根据探索方向信息作出相应操作,即检测到铁罐则拾取铁罐,未检测到铁罐则根据周围墙壁情况选择前进方向。根据上述文字描述可知 TakeAction 包括两种模式:$CanDir$ 不为空即 $CanDir \neq 0$ 时,进行拾取操作;$WallDir$ 不为空即 $WallDir \neq 0$ 时,进行移动操作,因此,需要定义 2 个 CTL 规约分别表示拾取和移动两种模式。以拾取模式为例,该规约表示为 $AG(CANDIR \neq 0 \rightarrow EFSTATE = OP_1)$,$OP_1 = t$,$t = PICKCAN$。根据上述分析,得到图 9 所示的 TakeAction 功能语义模型。

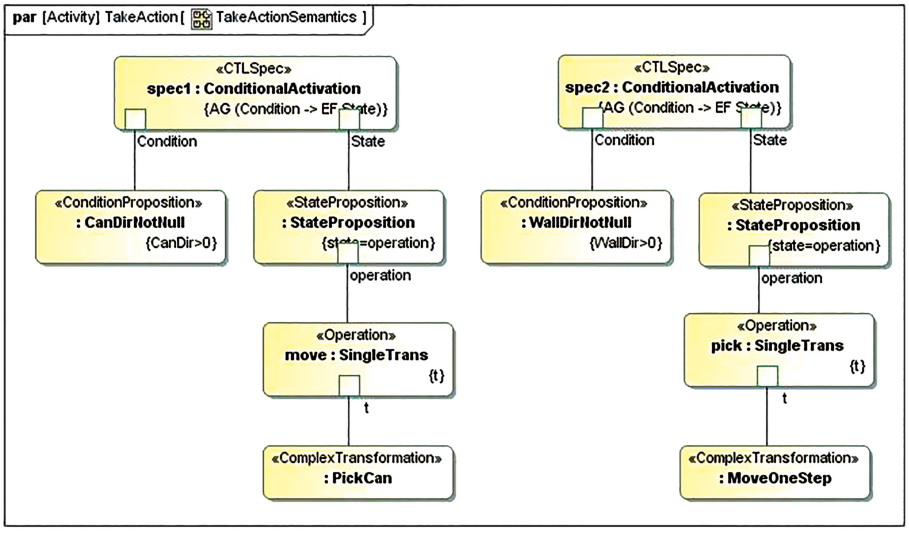


图 9　TakeAction 功能语义模型

Fig.9　Functional semantics model of TakeAction

2)功能分解。由于 TakeAction 功能语义模型包含 2 个 CTL 规约,即图 9 中的 spec1 和 spec2,因此,将 TakeAction 功能自动识别为混合功能。根据混合功能分解原则识别出子功能移动一步(MoveOneStep)、拾取罐子(PickCan),以及 1 个判断/合并节点。

3)功能结构建模。创建步骤 2 识别出的 3 个子功能并将子功能拖入 TakeAction 的活动图以对其进行实例化。通过控制流与对象流连接控制节点与功能实例,形成图 10 所示的功能结构模型。

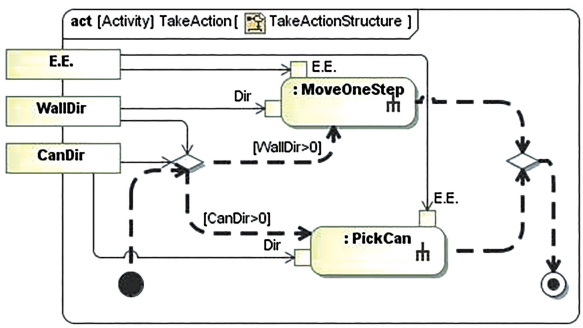


图 10　TakeAction 功能结构模型

Fig.10　Functional structure model of TakeAction

4.3　移动一步(MoveOneStep)功能分析过程

识别出的子功能 MoveOneStep、PickCan 还需要进行进一步分解。这里以 MoveOneStep 功能为例说明分析过程。

1)功能语义建模。MoveOneStep 功能语义的文字描述为:移动机器人系统根据周围墙壁情况移动一格,即 $WallDir = 1$ 时向左移动一格,$WallDir = 2$ 时向前移动一格,$WallDir = 3$ 时向右移动一格,$WallDir = 4$ 时向后移动一格。由于机器人只能原地旋转或向前移动,因此,向左移动一格的执行方式为先左转、再向前移动一格;向右移动一格的执行方式为先右转、再向前移动一格;后退移动一格的执行方式为先左转两次、再向前移动一格。根据文字描述,可定义 4 个 CTL 规约,分别表示 4 种移动模式。以向左移动为例,该规约表示为 $AG(WALLDIR = 1 \rightarrow EFSTATE = OP_1)$,$OP_1 = t_1 \& t_2$,$t_1 = TURNLEFT$,$t_2 = MOVEFORWARD$。根据上述分析,得到图 11 所示的 MoveOneStep 功能语义模型。

2)功能分解。MoveOneStep 的功能语义模型具有规约层,因此,MoveOneStep为混合功能。

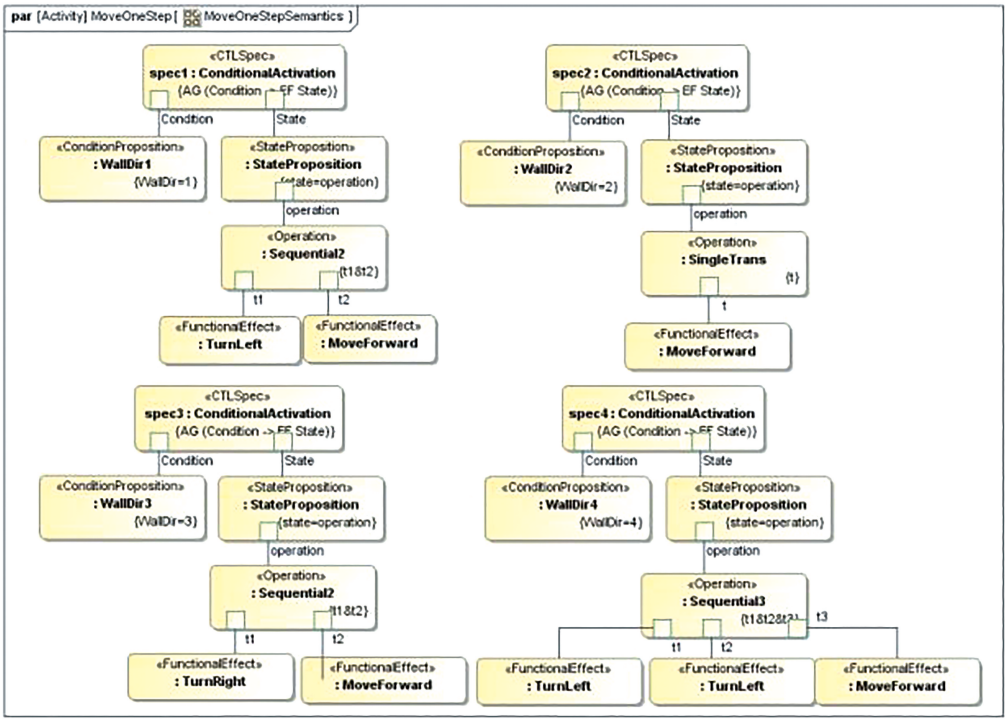


图 11　MoveOneStep 功能语义模型

Fig.11　Functional semantics model of MoveOneStep

根据混合功能分解原则将 MoveOneStep 分解为子功能左转(TurnLeft)、右转(TurnRight)、前进(MoveForward),以及 1 个判断节点。

3)功能结构建模。创建步骤 2 识别出的 3 个子功能及 1 个判断节点,并将子功能拖入 MoveOneStep 的活动图并对其进行实例化。通过控制流与对象流连接控制节点与功能实例,形成图 12 所示的功能结构模型。

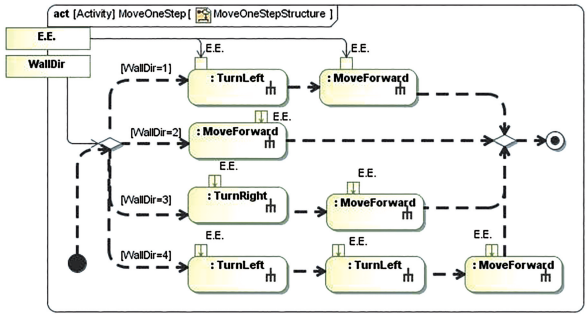


图 12　MoveOneStep 功能结构模型

Fig.12　Functional structure model of MoveOneStep

子功能 TurnLeft、TurnRight 和 MoveForward 的语义模型均只具有物理转换,因此,3 个子功能均为物理功能,至此,功能分析过程结束。所有底层子功能识别完毕后,将所有功能结构模型平面化,形成可直接用于设计集成的底层功能结构模型,如图 13 所示。

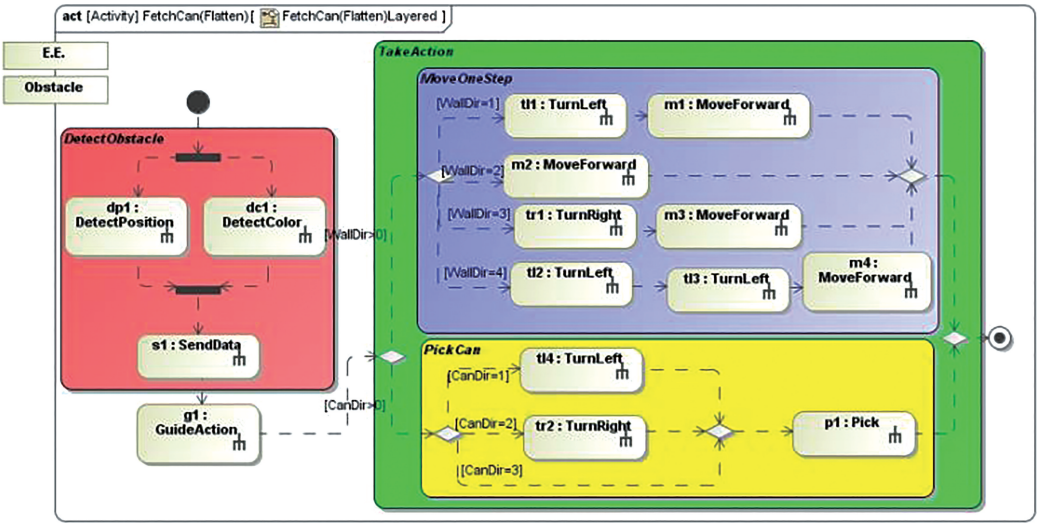


图 13　底层功能结构模型

Fig.13　Leaf-level functional structure model

5　讨论

从上述案例分析可以看出，本方法在功能表示与功能分解过程中考虑了功能的执行序列和模式，因而产生的功能结构模型能完整描述功能之间的执行逻辑。这些执行逻辑对后续设计集成中物理原理解的选择具有重要意义。功能模型要求 TurnLeft、TurnRight 功能与 MoveForward 功能必须顺序执行，因此，所选驱动机构的平动和转动 2 个物理效应必须独立开展，因而能排除阿克曼转向这一物理原理解。传统方法仅能分解出物理功能 TurnLeft、TurnRight、MoveForward，但无法表达功能之间的执行逻辑，因而难以识别逻辑顺序约束。由于本文方法分析产生的功能结构模型包含功能执行逻辑，因此，可从功能分解结果推理出软件控制信息，为后续的软件设计模型生成提供支持。综上可得，本文的功能分析方法产生的功能模型是支撑复杂机电系统软件与物理并行设计开展的重要基础。

值得一提的是，本文方法特别适用于对执行顺序要求较高的各类复杂机电系统。对于物理占主导、软件参与度较低的系统（如吹风机），设计过程只需涵盖其物理子系统，而无需考虑软件与物理的统一功能分析与设计。

6　结论

1）对基于流的传统功能表示方法进行了接口和语义的扩展，同时融入时序逻辑的概念，使功能表示方法可同时表示物理子系统应实现的连续物理变换，以及软件子系统承担的执行顺序控制功能，为复杂机电系统软件与物理统一功能的表示提供形式化的基础。

2）提出的软件与物理融合功能分解方法在统一形式化功能表征的基础上，首先区分 3 种功能（物理功能、软件功能和混合功能）类型，并给出混合功能的自动分解策略。所提方法与已有的物理及软件功能分解方法兼容，因此，可通过与已有的功能分解方法相结合，形成软件与物理融合功能的分析方法。

与面向物理过程的功能建模与分解的传统方法相比，本文方法最主要的优势在于考虑了软件子系统承担的功能。本文方法增加了描述离散执行流程的操作层和规约层，使功能表征方法适用于复杂机电系统软件与物理相融合的特征。此外，本文方法与已有的物理功能分析方法兼容，保证了已有方法的重用性。本文方法目前考虑的软件功能类型还比较有限，仅研究了软件子系统对执行顺序的协调。事实上，由于软件开发的灵活性，软件子系统还承担着多种多样的功能，因此需要分析梳理软件功能类型以改进本文方法。

参考文献：

[1] PAHL G, WALLACE K, BLESSING L T M, et al. Engineering Design: a Systematic Approach[M]. 3rd ed. London: Springer, 2007.

[2] KRUSE B, GILZ T, SHEA K, et al. Systematic Comparison of Functional Models in SysML for Design Library Evaluation[J]. Procedia CIRP, 2014, 21: 34-39.

[3] HIRTZ J, STONE R B, McADAMS D A, et al. A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts[J]. Research in Engineering Design, 2002, 13(2): 65-82.

[4] YUAN Lin, LIU Yusheng, SUN Zhongfei, et al. A Hybrid Approach for the Automation of Functional Decomposition in Conceptual Design[J]. Journal of Engineering Design, 2016, 27(4/6): 333-360.

[5] YUAN Lin, LIU Yusheng, LIN Yunfeng, et al. An Automated Functional Decomposition Method Based on Morphological Changes of Material Flows[J]. Journal of Engineering Design, 2017, 28(1): 47-75.

[6] CHEN Yong, LIU Zelin, XIE Youbai. A Knowledge-based Framework for Creative Conceptual Design of Multi-disciplinary Systems[J]. Computer-Aided Design, 2012, 44(2): 146-153.

[7] SRIDHARAN P, CAMPBELL M I. A Study on the Grammatical Construction of Function Structures[J]. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 2005, 19(3): 139-160.

[8] SANGELKAR S, McADAMS D A. Mining Functional Model Graphs to Find Product Design Heuristics with Inclusive Design Illustration[J]. Journal of Computing and Information Science in Engineering, 2013, 13(4): 041008.

[9] CHEN Bin. Conceptual Design Synthesis Based on Series-parallel Functional Unit Structure[J]. Journal of Engineering Design, 2018, 29(3): 87-130.

[10] MALMQVIST J. Improved Function-means Trees by Inclusion of Design History Information[J]. Journal of Engineering Design, 1997, 8(2): 107-117.

[11] UMEDA Y, TOMIYAMA T. Functional Reasoning in Design[J]. IEEE Expert, 1997, 12(2): 42-48.

[12] CHEN Yong, ZHAO Meng, LIU Ying, et al. A Formal Functional Representation Methodology for Conceptual Design of Material-flow Processing Devices[J]. Artificial Intelligence for Engineering De-

sign, Analysis and Manufacturing, 2016, 30(4): 353-366.

[13] GUI Fajun, CHEN Yong, LI Haomin, et al. A Structured Approach for Functional Analysis of Context-aware Systems[J]. Concurrent Engineering, 2023, 31(1/2):21-35.

[14] TOMIZUKA M. Mechatronics: From the 20th to 21st Century[J]. Control Engineering Practice, 2002, 10(8):877-886.

[15] PENAS O, PLATEAUX R, PATALANO S, et al. Multi-scale Approach from Mechatronic to Cyber-physical Systems for the Design of Manufacturing Systems[J]. Computers in Industry, 2017, 86:52-69.

[16] GROOVER M P. Automation, Production Systems, and Computer-integrated Manufacturing [M]. 3rd ed. Delhi:Pearson, 2007.

[17] KAVI, BUCKLES, BHAT. A Formal Definition of Data Flow Graph Models[J]. IEEE Transactions on Computers, 1986, C-35(11):940-948.

[18] GULWANI S, POLOZOV O, SINGH R. Program Synthesis[J]. Foundations and Trends in Programming Languages, 2017, 4(1/2):1-119.

[19] DEBBABI M, HASSAÏNE F, JARRAYA Y, et al. Verification and Validation in Systems Engineering: Assessing UML/SysML Design Models [M]. Berlin:Springer, 2010.

[20] LJUNGKRANTZ O, AKESSON K, FABIAN M, et al. Formal Specification and Verification of Industrial Control Logic Components [J]. IEEE Transactions on Automation Science and Engineering, 2010, 7(3):538-548.

[21] GIESE H, HENKLER S. A Survey of Approaches for the Visual Model-driven Development of Next Generation Software-intensive Systems[J]. Journal of Visual Languages & Computing, 2006, 17(6): 528-550.

[22] BRÄUNL T. Embedded Robotics: Mobile Robot Design and Applications with Embedded Systems [M]. 2nd ed. Berlin:Springer, 2014

（编辑 张 洋）

作者简介：曹 悦*，女，1986 年生，讲师。研究方向为模型驱动系统设计。发表论文 10 余篇。E-mail:ycao@zjut.edu.cn。

本文引用格式：

曹悦，刘玉生，秦绪佳，等.复杂机电系统的软件与物理统一的形式化功能分析[J]. 中国机械工程，2025，36(2)：245-254.

CAO Yue, LIU Yusheng, QIN Xujia, et al. Software-physical Unified Formal Functional Analysis for Complex Mechatronic Systems[J]. China Mechanical Engineering, 2025, 36(2):245-254.

（上接第 244 页）

[8] 吴玉光，李春光. 夹具定位误差分析的机构学建模方法[J]. 中国机械工程，2011，22(13)：1513-1518.
WU Yuguang, LI Chunguang. Approach to Mechanism Modeling of Fixture Location Error Analyses [J]. China Mechanical Engineering, 2011, 22(13): 1513-1518.

[9] 毛护国. 发动机缸体加工定位误差分析[D]. 武汉：华中科技大学，2014.
MAO Huguo. Analysis of Positioning Error in Engine Cylinder Block Processing[D]. Wuhan: Huazhong University of Science and Technology, 2014.

[10] 吴拓. 现代机床夹具设计[M]. 北京：化学工业出版社，2009：20-35.
WU Tuo. Design of Modern Machine Tool Fixture [M]. Beijing:Chemical Industry Press, 2009:20-35.

[11] 郑伟. 一种汽车轮毂柔性夹具的设计与研究[D]. 南京：南京林业大学，2020.
ZHENG Wei. Design and Research of a Flexible Fixture for Automotive Wheel Hubs[D]. Nanjing: Nanjing Forestry University, 2020.

[12] 郑伟，孙见君，马晨波，等. 汽车轮毂柔性加工夹具的定位误差分析[J]. 机械设计，2021，38(3)：46-52.
ZHENG Wei, SUN Jianjun, MA Chenbo, et al. Positioning Error Analysis of Flexible Machining Fixtures for Automotive Wheel Hubs[J]. Mechanical Design, 2021, 38(3):46-52.

[13] WANG Hui, RONG Y K, LI Hua, et al. Computer Aided Fixture Design: Recent Research and Trends [J]. Computer-Aided Design, 2010, 42 (12):1085-1094.

[14] 陈强. C-25D 连续驱动摩擦焊机结构分析及优化设计[D]. 长春：长春工业大学，2021.
CHEN Qiang. Structural Analysis and Optimization Design of C-25D Continuous Drive Friction Welding Machine[D]. Changchun: Changchun University of Technology, 2021.

（编辑 陈 勇）

作者简介：孙宝玉，女，1971 年生，教授。研究方向为精密机械微驱动技术等。关英俊*（通信作者），男，1978 年生，教授、博士研究生导师。研究方向为大型数控机床数字化设计与制造技术等。E-mail:guanyingjun@ccut.edu.cn。

本文引用格式：

孙宝玉，李焕震，张刚，等.重型高精度惯性摩擦焊机移动夹具定位误差分析[J]. 中国机械工程，2025，36(2)：238-244.

SUN Baoyu, LI Huanzhen, ZHANG Gang, et al. Analysis of Positioning Errors of Heavy Duty High Precision Inertial Friction Welding Mobile Fixtures [J]. China Mechanical Engineering, 2025, 36(2):238-244.