ROBOTICS

应用手册

Externally Guided Motion

Trace back information:
Workspace 21A version a10
Checked in 2021-03-16
Skribenta version 5.4.005

应用手册

Externally Guided Motion

RobotWare 7.2

文档编号: 3HAC073318-010
修订: D

本手册中包含的信息如有变更，恕不另行通知，且不应视为 ABB 的承诺。 ABB 对本手册中可能出现的错误概不负责。

除本手册中有明确陈述之外，本手册中的任何内容不应解释为 ABB 对个人损失、财产损坏或具体适用性等做出的任何担保或保证。

ABB 对因使用本手册及其中所述产品而引起的意外或间接伤害概不负责。

保留以备将来参考。

可从 ABB 处获取此手册的额外复印件。

本出版物为译本。

目录

手册概述

关于本手册

本手册包含了 RobotWare 选装件 Externally Guided Motion [3124-1]（通常称为 EGM）的相关信息。

手册用法

本手册可用于理解 Externally Guided Motion 的定义以及其使用方法。本手册还提供了关于 Externally Guided Motion 相关 RAPID 组件的信息以及它们的使用方法示例。

本手册的阅读对象

本手册主要供机械臂编程员使用。

操作前提

读者应该熟悉的内容：

- 工业机器人及术语
- RAPID 编程语言
- 系统参数及其配置方式

参考信息

参考文档	文档编号
应用手册 - 控制器软件 *OmniCore*	*3HAC066554-010*
操作手册 - *OmniCore*	*3HAC065036-010*
操作手册 - *RobotStudio*	*3HAC032104-010*
技术参考手册 - *RAPID Overview*	*3HAC065040-010*
技术参考手册 - *RAPID*指令、函数和数据类型	*3HAC065038-010*
技术参考手册 - 系统参数	*3HAC065041-010*

修订版

版本号	描述
A	随 RobotWare 7.0 发布。
B	随 RobotWare 7.0.1 发布。 • 更改了参数第*40*页的*Default Ramp Time*的默认值。
C	随 RobotWare 7.1 发布。 • 关于*K*因子（默认比例位置增益）的说明添加在第第*20*页的输入数据节中。 • 关于不可预测移动的说明添加在第第*42*页的*Default Low Pass Filter Bandwidth*节中。 • 指令第*60*页的*EGMRunJoint - 执行一次含一个关节目标点的EGM移动*和第*63*页的*EGMRunPose - 执行一次含一个姿态目标点的EGM移动*更新的限制。 • 第第*11*页的概述节中更新的限制。 • 已从指令EGMSetupAI、EGMSetupAO和EGMSetupGI中移除参数 LATR，该参数已不再有效。 • 第*14*页的*EGM Position Guidance*介绍中添加了新的设置部分。

下一页继续

续前页

版本号	描述
D	随 RobotWare 7.2 发布。 • 章节第*14*页的*EGM Position Guidance*介绍和第*33*页的配置*UdpUc*装置中更新了系统参数类型UDP Unicast Device的信息。 • 第*32*页的构建*EGM*传感器通信端点已更新。 • 章节第*27*页的概述更新了关于EGM传感器协议数据结构的信息。

产品文档

ABB 机器人用户文档类别

ABB机器人用户文档分为多个类别。以下列表基于文档的信息类型编制，而未考虑产品为标准型还是选购型。

提示

所有文档都可从myABB门户网*www.abb.com/myABB*上获得。

产品手册

机械臂、控制器、DressPack/SpotPack 和其他大多数硬件交付时一般都附有包含以下内容的**产品手册**：

- 安全信息。
- 安装与调试（介绍机械安装或电气连接）。
- 维护（介绍所有必要的预防性维护程序，包括间隔周期和部件的预计使用寿命）。
- 维修（介绍所有建议的维修程序，包括零部件）。
- 校准。
- 停用。
- 参考信息（安全标准、单位换算、螺钉接头和工具列表）。
- 备件清单附相关图示（或各备件清单索引）。
- 请参阅电路图。

技术参考手册

技术参考手册介绍了机器人产品参考信息，如润滑、RAPID语言和系统参数等。

应用手册

应用手册中将介绍具体应用产品（例如软件或硬件选项）。一本应用手册可能涵盖一个或多个应用产品。

应用手册通常包含以下信息：

- 应用产品用途（作用及使用场合）。
- 所含内容（如电缆、I/O板、RAPID指令、系统参数或软件等）。
- 如何安装所包含的或所需的硬件。
- 如何使用应用产品。
- 应用产品使用示例。

操作手册

操作手册介绍了产品的实际处理流程。手册面向直接接触产品的操作人员，即生产车间操作员、程序员和故障排除人员。

安全

人员安全

机器人速度慢，但是很重并且力度很大。在机器人运动过程中的停顿和停止之后都有可能发生危险。即使可以预测运动轨迹，但外部信号也可能改变操作，会在没有任何警告的情况下，产生意想不到的运动。

因此，在进入机器人工作区域前请确保所有安全守则都被严格执行。

安全守则

在开始操作机器人之前，请确保已经熟悉了解手册机器人安全手册 - 机械臂和 *IRC5* 或 *OmniCore* 控制器中描述的安全守则。

1 Externally Guided Motion介绍

1.1 概述

目的

*Externally Guided Motion (EGM)*提供了3种不同的特性：

- *EGM Position Stream*：

  RAPID任务中机械单元的当前和计划位置发送至外部系统。
- *EGM Position Guidance*:

  相关机器人不会沿RAPID中的编程路径移动，而是沿某件外部装置所生成的路径移动。
- *EGM Path Correction*:

  用某件外部装置提供的测量值来修改 / 校正所编写的机器人路径。

EGM Position Stream

EGM Position Stream旨在向外部设备提供受机器人控制器控制的机械单元的当前和计划位置。

一些应用装置的实例如下所示：

- 在激光头正在动态控制激光束的地方进行激光焊接。
- 任何用外部控制器控制“机器人”-TCP的机器人设备。

EGM Position Guidance

*EGM Position Guidance*的作用是用一件外部装置来生成一台或多台机器人所用的位置数据。这些机器人将会移到给定位置处。

一些应用示例：

- 将一个对象（比如一扇车门或车窗）放在某个外部传感器给出的位置（比如车体）处。
- 仓中取物。从一个料仓中捡拾对象（该料仓使用了一个外部传感器来识别这一对象及其位置）。

EGM Path Correction

*EGM Path Correction*的作用是用安装在装置上的外部机器人来生成一件外部装置来生成一台或多台机器人所用的路径校正数据。后一类机器人将会沿校正后的路径（即添加了所测校正值的相应编程路径）。

一些应用示例：

- 焊缝跟踪
- 跟踪在已知路径附近移动的对象。

其中包括

您可通过RobotWare选项*Externally Guided Motion*来访问：

- 开始和停止EGM Position Stream指令。
- 用于设置、激活和重置EGM Position Guidance的指令。
- 用于设置、激活和重置EGM Path Correction的指令。
- 启动EGM Position Guidance运动的指令，是否与执行同步，然后停止。

下一页继续

- 用于实现EGM Path Correction移动的指令。
- 用于检索当前EGM状态的一则函数。
- 用于配置EGM和设置默认值的系统参数。

限制

EGM Position Stream的限制

- EGM Position Stream仅适用于UdpUc通信。
- 在活跃位置流中不能动态更改工具数据和加载数据。
- 不能串流协调的MultiMove系统的位置。
- 如果使用`EGMStreamStart`开始串流，则不支持Absolute Accuracy；但如果使用`EGMActXXX\StreamStart`开始串流，则支持。
- EGM Position Stream不兼容EGM Path Correction。
- 当EGM Position Stream启用时，不允许激活或停用机械单元。

EGM Position Guidance的限制

- 必须从某个精确点开始和结束。
- 控制器重启后执行的第一个动作不能使EGM动作。
- 姿势模式只支持6轴机器人。
- 由于EGM Position Guidance不包含插补器功能，因此用户无法用EGM Position Guidance来实现直线移动。相关机器人的实际路径将取决于该机器人的配置、启动位置和所生成的位置数据。
- EGM Position Guidance不支持MultiMove。
- 如果某个工件正在移动，那么用户就无法用EGM Position Guidance来指引该工件中的某个机械单元。
- 如果相关机械臂陷入了某个奇异点附近（也就是两根机械臂轴近乎平行），那么系统会停止该机械臂的移动并产生一则错误消息。此时唯一的办法就是以点动方式让该机械臂脱离奇异点。
- 当EGM处于活动状态时，Motion Supervision 的行为可能与正常移动时不同。碰撞后的建议行动是从开始就禁用EGM并启动EGM序列。

EGM Path Correction的限制

- 只支持6轴机器人。
- 必须从某个精确点开始和结束。
- 该外部装置必须安装有机器人。
- 只能在相关的路径坐标系上实施校正。
- 只能进行y和z方向上的位置校正。用户既无法实施方位校准，也无法在x方向（即路径方向 / 切线方向）上校准。
- 当EGM处于活动状态时，Motion Supervision 的行为可能与正常移动时不同。碰撞后的建议行动是从开始就禁用EGM并启动EGM序列。

1.2 EGM Position Stream简介

什么是EGM位置流？

EGM Position Stream 仅适用于 UdpUc 通信。它可以定期从机器人控制器发送计划的以及实际的机械单元（例如机器人、定位器、轨道运动装置）位置数据。消息内容由 Google Protobuf 定义文件 *egm.proto* 指定。循环通信通道 (UDP) 可以在机器人控制器（该控制器可确保频率高达 250 Hz 的稳定数据交换）的高优先级网络环境中执行。每项运动任务都必须有一条通信通道。

EGM Position Stream 可与 EGM Position Guidance一同使用。

1.3 EGM Position Guidance介绍

什么是EGM Position Guidance

EGM Position Guidance是为高级用户设计的，它通过绕过相关路径规划（需要高响应性的机器人移动时可以采用这种规划）的方式为相关的机器人控制器提供了一个低级接口。用户可用EGM Position Guidance来高速读取相关运动系统的位置和向该系统高速写入位置（可达到每4毫秒一次，并伴随10到20毫秒的控制延迟。具体的延迟取决于相关的机器人类型）。可以用关节值或某种姿态来制定相关的引用项。用户可在EGM Position Guidance移动期间不会移动的任何工件上定义这种姿态。

注意

在所有关于 EGM 和 RRI 的进一步描述中，实际采样时间是真实机器人系统上的 4.032 毫秒，而在虚拟机器人系统中大约是4 毫秒。

EGM Position Guidance会处理一切必要的滤波、引用项监控和状态处理事宜。状态处理事宜包括程序启动 / 停止和紧急停止等。

与其它外部运动控制手段相比，EGM Position Guidance的主要优点在于较高的速率和较低的延时 / 等待时间。从“写入一个新位置”到“该给定位置开始影响实际的机器人位置”之间的时间通常约为20毫秒。

EGM会处理*Absolute Accuracy*。

EGM Position Guidance 可以与逻辑设置（I/O 设置等）或使用“强制控制”组件指令启用其他模式。这可以通过使用参数不等待 EGM 收敛 (\NoWaitCond) 来实现。

EGM Position Guidance所不为之事

EGM会直接进入相关的电机引用项生成过程，即是说不会提供任何路径规划。这意味着您无法下令移到某个姿态目标点并期望这是一次直线移动。用户既无法下令进行一次指定速度下的移动，也无法下令进行一次应耗费指定时间的移动。

如果要下令进行此类需要路径规划的移动，那么您可使用RAPID中的标准移动指令，即`MoveL`和`MoveJ`等等。

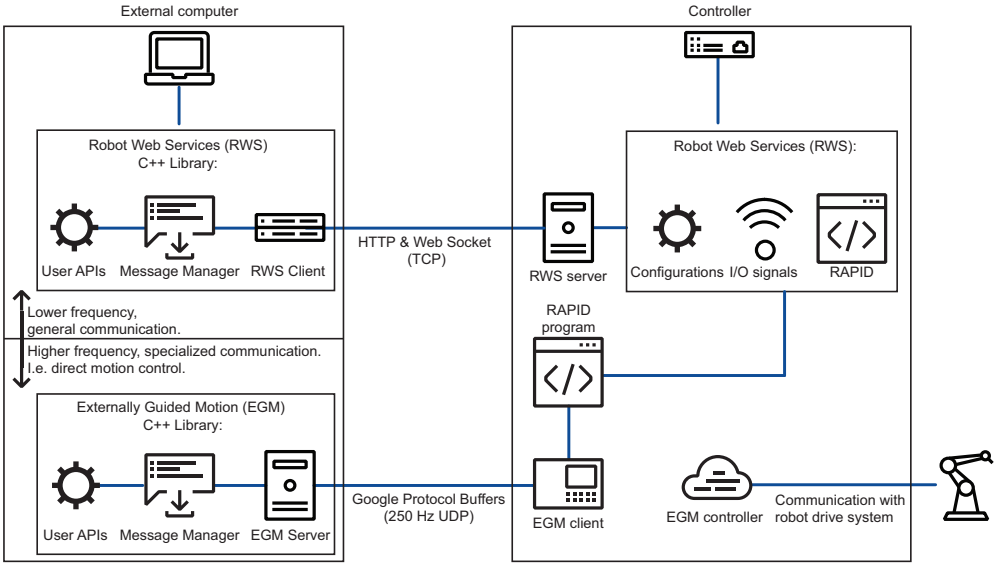
警告

由于相关机器人控制器中的EGM绕过了路径规划，因此用户输入项会直接创建相应的机器人路径，所以很重要的一点就是确保发送给该控制器的位置引用项流要尽量平顺。相关机器人会迅速对发送给相关控制器的所有位置引用项作出反应，连存在故障的机器人也是如此。

下一页继续

设置EGM Position Guidance

当设置正确EGM Position Guidance时，外部设备通过协议与控制器通信，从而控制机械臂移动。下图显示了EGM应用程序部分设置的示意图：



xx2000000152

为此，必须采取以下步骤：

1 设置您的RobotWare系统，包括RobotWare的选项*Externally Guided Motion*。

> **注意**
>
> 有关设置中可以包括哪些类型的机械臂和选项的信息，请参见第*12*页的限制。

2 使用EGM传感器协议设置您的UdpUc设备，以便它可以与机器人控制器通信。有关如何构建EGM传感器通信端点的基本说明，请参见第*27*页的*EGM*传感器协议。

> **注意**
>
> 发行版RobotWare中提供了代码示例，请参见第*107*页的*UdpUc*代码示例。

3 设置定义EGM Position Guidance功能详细信息的RAPID代码。有关代码所需基本元素的描述，请参见第*17*页的基本方法；有关RAPID代码的完整示例，请参见第*95*页的使用带一件*UdpUc*装置的*EGM Position Guidance*。

4 为EGM提供输入数据的设备必须配置为UDP Unicast Device (UdpUc)。此配置在RobotStudio中进行，使用主题*Communication*中类型*UDP Unicast Device*的系统参数。确定设备的名称和IP地址，并将传输协议设置为UDPUC，请参阅第*33*页的配置*UdpUc*装置。

> **注意**
>
> 进行这种配置改动后必须重启相关控制器。重启后，可通过EGM使用该设备指引机械臂。

1.4 EGM Path Correction介绍

什么是EGM Path Correction

EGM Path Correction使用户或可校正一条编写好的机器人路径。用于测量实际路径的装置或传感器必须安装在相关机器人的工具法兰上，且该装置或传感器必须能够校准相应的传感器框架。

在相关的路径坐标系上实施校正。该坐标系的x轴来自相关路径的切线，其y轴为该路径切线的叉积，其z轴和已激活工具框架的z方向则是x轴和y轴的叉积。

EGM Path correction必须在精细点启动和结束。传感器测量结果可以在约48 ms的倍数时间提供。

2 使用 Externally Guided Motion

2.1 基本方法

EGM Position Stream的基本方法

若UdpUc用于与外部设备通信，则EGM Position Stream适用。可用两种不同的方式开始EGM Position Stream。一种是使用EGMStreamStart，另一种是使用EGMActJoint\StreamStart或EGMActPose\StreamStart。通过EGMStop、EGMReset且当EGMRunJoint或EGMRunPose的指令完成时，EGM Position Stream自动停止。还有另一个特定指令EGMStreamStop，停止数据流。

位置流不支持工具或负载的动态变化。当EGMStreamStart用于启动位置流时，有效工具和负载将传至控制器。当使用EGMActJoint 或 EGMActPose时，有效工具和负载（如另有规定，则指定工具和/或负载）将传至控制器。此类工具和负载数据随后为EGM所用，用于计算位置，直至位置流停止。对于各运动任务，必须启动单独的位置流。

	操作
1	登记一个EGM客户端，然后获取一个EGM标识。之后要用该标识来把设置、激活、移动和停用与一种特定的EGM用法关联起来。EGM的状态仍为EGM_STATE_DISCONNECTED。
2	调用EGM设置指令EGMSetupUC，使用UdpUc协议连接设置外部设备。EGM状态更改为EGM_STATE_CONNECTED。
3	或者： A 通过指令EGMStreamStart启动位置流。 B 使用EGMActJoint或EGMActPose以及可选变元\StreamStart开始位置流。
4	将会激活EGM Position Stream，发送实际和计划位置，直到停止。
5	A 当采用EGMStreamStart启动时： 通过EGMStreamStop停止位置流。 B 当采用EGMActJoint或EGMActPose启动时： 通过EGMStop或EGMReset，停止位置流。

EGM Position Guidance的基本用法

如果用一件外部装置（传感器）来提供移动目标点，那么这就是移动 / 指引一台机器人的一般方式。

	操作
1	将相关机器人移到一个精确点处。
2	登记一个EGM客户端，然后获取一个EGM标识。之后要用该标识来把设置、激活、移动和停用与一种特定的EGM用法关联起来。EGM的状态仍为EGM_STATE_DISCONNECTED。
3	调用一条EGM设置指令来设置使用了信号或UdpUc协议连接的位置数据源。EGM的状态会变成EGM_STATE_CONNECTED。
4	选择是否将该位置作为关节值或一种姿态，并给出该位置的收敛标准（即何时视为已抵达该位置）。
5	如果选择了姿态，那么就定义使用哪个框架来定义相应的目标点位置以及在哪个框架中移动。
6	给出停止模式（一项可选的超时），然后执行移动本身。此时EGM的状态为EGM_STATE_RUNNING。这是相关机器人正在移动时的状态。

下一页继续

	操作
7	系统将在视为已抵达该位置（即是说已满足了相应的收敛标准）时停止EGM移动。此时EGM的状态又变回EGM_STATE_CONNECTED。

EGM Path Correction的基本用法

这是用EGM Path Correction校正一条编程路径的一般方式。

	操作
1	将相关机器人移到一个精确点处。
2	登记一个EGM客户端，然后获取一个EGM标识。之后要用该标识来把设置、激活、移动和停用与一种特定的EGM用法关联起来。EGM的状态仍为EGM_STATE_DISCONNECTED。
3	调用一条EGM设置指令来设置使用了信号或UdpUc协议连接的位置数据源。EGM的状态会变成EGM_STATE_CONNECTED。
4	定义相关传感器的校正框架（该框架始终属于工具框架）。
5	执行移动本身。此时EGM的状态为EGM_STATE_RUNNING。
	EGM将在下一个精确点处返回状态EGM_STATE_CONNECTED。
6	若要释放一个EGM标识以用于其它传感器，您就必须重置EGM，从而使EGM返回状态EGM_STATE_DISCONNECTED。

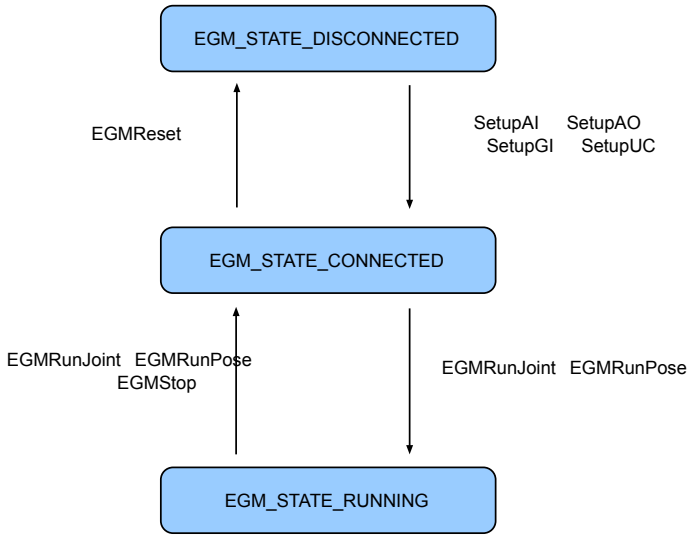
2.2 执行状态

描述

EGM进程拥有不同的状态：

值	描述
EGM_STATE_DISCONNECTED	未定义这一具体进程的EGM状态。 未激活任何设定。
EGM_STATE_CONNECTED	未激活指定的EGM进程。 已进行过设置，但并未激活任何EGM移动。
EGM_STATE_RUNNING	正在执行指定的EGM进程。 EGM移动处于激活状态，即是说移动了相关机器人。

不同状态之间的过渡情况如下图所示。



xx1400001082

RAPID指令EGMRunJoint和EGMRunPose始于EGM_STATE_CONNECTED，而只要还未达到相关目标点位置的收敛标准，或还未超过超时时间，那么这两条指令就会使相关状态变为EGM_STATE_RUNNING。当满足了其中一项条件时，EGM的状态会再次变为EGM_STATE_CONNECTED，而这两条指令则就此结束（即是说RAPID会继续执行下一条指令）。

如果EGM的状态为EGM_STATE_RUNNING，且停止了RAPID的执行过程，那么EGM则会进入状态EGM_STATE_CONNECTED。当程序重启时，EGM会返回状态EGM_STATE_RUNNING。

如果用从程序指针到主例程（PP to Main）或从PP到光标来移动相应的程序指针，那么EGM之前的状态又是EGM_STATE_RUNNING，那么EGM的状态就会变为EGM_STATE_CONNECTED。

2.3 输入数据

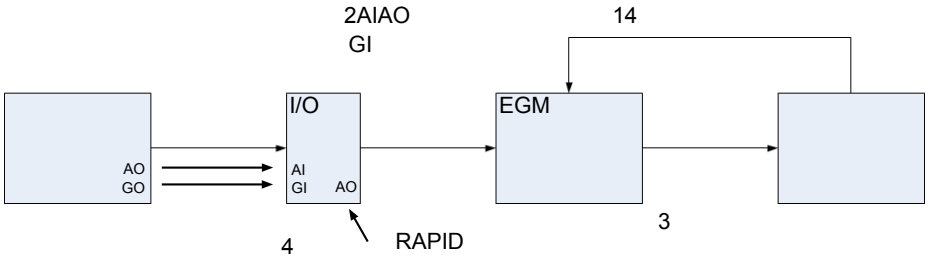
用于EGM Position Guidance的输入数据

用相关的EGM设置指令来选择输入数据来源。头三条指令会选择一个信号接口，而最后一条指令会选择一个UdpUc接口（*User Datagram Protocol Unicast Communication*）。

指令：	描述
EGMSetupAI	为EGM设置模拟输入信号
EGMSetupAO	为EGM设置模拟输出信号
EGMSetupGI	为EGM设置编组输入信号
EGMSetupUC	为EGM设置UdpUc协议

用于EGM的输入数据主要包含了作为关节或某种姿态的位置数据，也就是加上方位后的笛卡尔位置。

相关信号接口的数据流如下所示：



xx1400002016

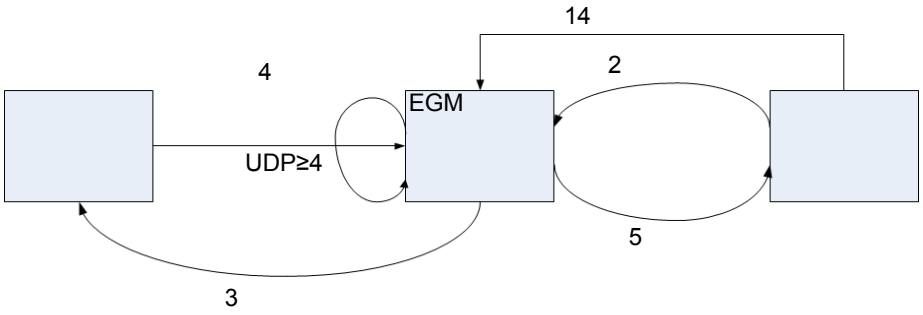
1 运动控制会调用到EGM。
2 EGM会读取相关信号的位置值。
3 EGM会在运动控制中写入相关的位置数据。
- 相关传感器会把位置数据写入相应的信号中。

如果把信号作为数据源，那么该输入项就仅限于相关机器人的6，即6关节值或3笛卡尔位置值（x、y和z）再加上3欧拉角度值（rx、ry和rz），最高可达附加轴的6数值。

当对7轴机器人使用EGM关节模式时，由第一根附加轴输入信号提供附加机器人轴的位置。

下一页继续

用于UdpUc接口的数据流如下所示：



xx1400002017

1 运动控制会调用到EGM。
2 EGM会从运动控制中读取反馈数据。
3 EGM会把反馈数据发送给相关的传感器。
4 EGM会按相关传感器的消息来检查UDP队列。
5 如果有一条消息，那么EGM会读取下一条消息，而到步骤5时则会在运动控制中写入相关的位置数据。如果没有发送任何位置数据，那么运动控制会继续使用EGM之前写入的最后一项位置数据。

- 传感器会向相应的控制器（EGM）发送位置数据。我们的建议是将这种发送与步骤3耦合起来，使该传感器与该控制器同相。

该控制环路是建立以下速度 – 位置关系上的：

*speed = k * (pos_ref – pos) + speed_ref*	*k*-因子（默认比例位置增益） *pos_ref* – 参考位置 *pos* – 所需位置 *speed_ref* – 参考速度

至于如何用各种指令来执行针对某一外部装置的UdpUc协议，则请参见第27页的*EGM传感器协议*。该链接还描述了相关的输入数据。

用于EGM Path Correction的输入数据

用相关的EGM设置指令来选择输入数据来源。头三条指令会选择一个信号接口，而最后一条指令会选择一个UdpUc接口（*User Datagram Protocol Unicast Communication*）。

指令：	描述
EGMSetupAI	为EGM设置模拟输入信号
EGMSetupAO	为EGM设置模拟输出信号
EGMSetupGI	为EGM设置编组输入信号
EGMSetupUC	为EGM设置UdpUc协议

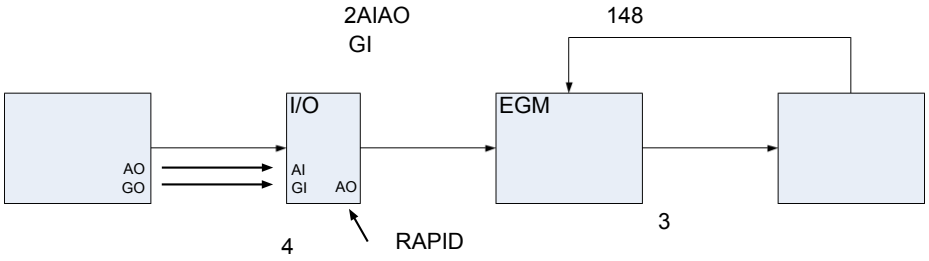
用于EGM的输入数据主要包含了位置数据。

下一页继续

2.3 输入数据

续前页

相关信号接口的数据流如下所示：



xx1400002016

1 运动控制会调用到EGM。
2 测量数据（y和x值）从信号读取或在48 ms的倍数时间从传感器获取。
3 EGM会计算相关的位置校正情况，然后将其写入运动控制中。如果采用了UdpUc协议，那么系统就会向相关传感器发送反馈。

2.4 输出数据

描述

输出数据仅适用于UdpUc接口。

至于如何用各种指令来执行针对某一外部装置的UdpUc协议，则请参见第27页的*EGM传感器协议*。该链接还描述了相关的输出数据。

2.5 配置

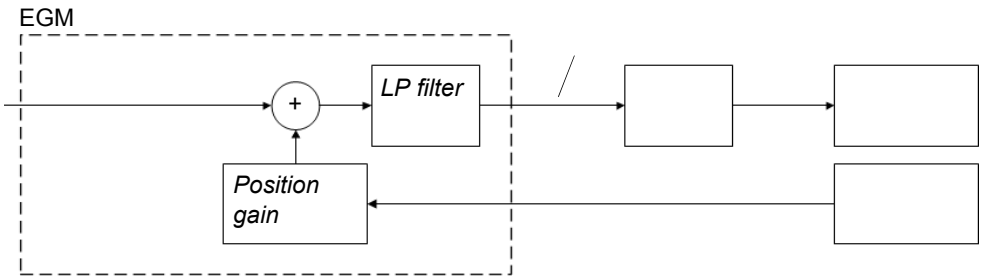
EGM位置流配置

RobotWare中预定义的默认配置适用于任何位置流，将使用与EGM位置引导相同的配置。

用于EGM Position Guidance的配置

主题*Motion*下类型为*External Motion Interface Data*的系统参数会影响到EGM的行为。第*35*页的系统参数描述了所有可用的EGM参数。

下文更仔细地描述了会影响到EGM控制环路的两个参数。其中展示了EGM控制系统的一幅简图。



xx1400001083

Default proportional Position Gain	图中的参数位置增益会影响到前往目标点位置（由相应的传感器给定，并与当前的机器人位置有关）的响应移动。该数值越高，响应速度就越快。
Default Low Pass Filter Bandwith Time	图中的参数*LP Filter*是过滤EGM的速度贡献量时所用的默认值。

用于EGM Path Correction的配置

EGM Path Correction 的配置必须将 *Level* 设置到 *Path*。没有使用任何其他值。

2.6 框架

EGM位置流的框架

位置流不支持工具或负载的动态变化。当EGMStreamStart用于启动位置流时，有效工具和负载将传至控制器。当使用EGMActJoint 或 EGMActPose时，有效工具和负载（如另有规定，则指定工具和/或负载）将传至控制器。此类工具和负载数据随后为EGM所用，用于计算位置，直至位置流停止。

用于EGM Position Guidance的框架

EGM可在两种不同模式——关节模式和姿态模式——下运行，而下一节的内容仅适用于EGM姿态模式。

在关节模式下，由于传感器值和位置值都是相对于每根轴校准位置的给定轴角度（以度为单位），因此不需要参考框架。但姿态模式却离不开参考框架，毕竟系统只能以相对于参考框架的方式给出相关传感器的测量值和位置变动的方向。

RAPID指令EGMActPose定义了可在EGM中使用的所有框架：

框架	描述
工具	由可选\Tool自变数来定义将用于EGM进程的工具数据。
工件坐标	由可选\Wobj自变数来定义将用于EGM进程的工件数据。
校正	由强制CorrFrame自变数来定义给出最终移动方向时使用的框架。
传感器	由强制SensorFrame自变数来定义解读相关传感器数据时使用的框架。

工具与工件

只能按两种组合来定义这些工具和工件：

1 如果将工具连接到机器人上，那么就必须固定相应的工件。
2 如果固定住工具，那么就必须将相应的工件连接到机器人上。

注意

用户无法把工件或工具连接到非EGM机器人的任何其它机械单元上。

预定义框架类型

用户有必要了解框架CorrFrame和SensorFrame的关联情况。用数据类型egmframetype中的预定义框架类型来指定这一信息：

值	描述
EGM_FRAME_BASE	以相对于基本框架的方式来定义该框架（姿态模式）。
EGM_FRAME_TOOL	坐标系是相对于tool0（姿势模式）定义的。
EGM_FRAME_WOBJ	坐标系是相对于当前工件（姿势模式）定义的。
EGM_FRAME_WORLD	以相对于全局框架的方式来定义该框架（姿态模式）。
EGM_FRAME_JOINT	这些数值为关节值（关节模式）。

用于EGM Path Correction的框架

EGM Path Correction只能在姿势模式下运行。

下一页继续

RAPID指令EGMActMove定义了EGM Path Correction所需的唯一框架。工具和工作对象在EGMMoveL或EGMMoveC中规定。

工具与工件

工具必须连接到机器人上，同时可用另一个机械单元来固定或移动相关工件。

3 EGM传感器协议

3.1 概述

3.1.1 EGM传感器协议概述

通信和传输协议

EGM传感器协议被设计用于以最少的经常性费用来实现机器人控制器与某个通信端点之间的高速通信。

通信端点通常为一个传感器，所以从此处起，我们将使用传感器来代替通信端点。有时该传感器会与一台PC相连，而该PC则会把传感器数据发送给相关机器人。这种传感器协议的用途是在相关的机器人控制器和传感器之间频繁交流传感器数据。EGM sensor protocol正在使用Google Protocol Buffers来进行编码，并把UDP则作为一则传送协议。选择Google Protocol Buffers的原因是其具有速度和语言中立性方面的优势。由于所发送的数据是高频发送的实时数据，且一旦丢失数据包，那么重新发送这些数据也无济于事，所以我们才选择了UDP作为传送协议。

数据结构和系统参数

由EGM proto文件定义EGM sensor protocol的数据结构。请在系统参数中配置相关的传感器名称、IP地址和传感器的端口号。最多能配置八个传感器。

消息和队列处理

该传感器正起到服务器的作用。在从相关机器人控制器处收到第一则消息前，该传感器无法向相关机器人发送任何内容；在收到第一则消息后，便可双向发送彼此独立的多则消息。采用了该协议的应用可能会对其使用情况作出一定限制，但该协议本身并不具备对请求响应或丢失消息监控的内置同步。这里没有专用的连接消息或断开消息，只有能双向独立流动的数据。来自相关机器人的第一则消息将是数据消息。此外用户还必须谨记，即使接收方的队列已满，UDP消息的发送方也仍会继续发送，所以接收方必须确保其队列是空的。

按默认设定，不论传感器何时发送数据，相关机器人都会每隔4毫秒向该传感器发送一次数据或从该传感器读取一次数据。用户可用RAPID指令`EGMStreamStart`、`EGMActJoint`或`EGMActPose`的可选变元`\SampleRate`来把这一周期时间改为为4毫秒的某个倍数。

各运动任务需要其专属UDP通道。

3.1.2 Google Protocol Buffers

Protobuf概述

Google Protocol Buffers或*Protobuf*能非常高效地对数据进行串行化 / 去串行化。Protobuf大致上比 XML快10到100倍。互联网上有大量关于Protobuf的信息，而从*Google overview*入手是个不错的选择。

简而言之，*.proto*文件描述了各种消息结构。此后要对*.proto*文件进行编译，而编译器则会生成供相关应用之后使用的串行化 / 去串行化代码。该应用会从相应的网络中读取一则消息，执行去串行化，创建一则消息，调用串行化方法，最后发送这一消息。

Protobuf具有语言中立性，所以大部分编程语言都或可使用Protobuf。不同的语言带来了许多不同的执行方式。

Protobuf的主要缺点是Protobuf消息会被串行化成某种二进制格式，这使用户更难通过网络分析仪来调试数据包。

第三方工具

除*Google C++*工具外，我们也验证了下列第三方工具和代码：

- *Nanopb*，生成C代码，且其无需分配任何动态内存。
- *Protobuf-net*，一个Google Protobuf .NET文库。
- *Protobuf-csharp*，是 Google Protobuf .NET，C# API与Google C++ API类似。

注意

注意上述代码为开源代码，这意味这您必须检查许可证，以确定是否允许在您的产品中使用相关代码。

3.1.3 EGM传感器协议说明

数据结构

EGM传感器协议不是一项请求 / 响应协议，当传感器从相关机器人处获得第一则消息后，该传感器便可以任何频率发送数据。

EGM传感器协议主要有两个数据结构，即 *EgmRobot*和*EgmSensor*。*EgmRobot*由机器人发送，*EgmSensor*由传感器发送。两种数据结构的所有消息字段都会被定义为可选，这意味着一则消息中可能有字段，也可能没有字段。如果某一应用使用了*Google Protocol Buffers*，则其必须检查是否存在可选字段。

Google protobuf数据结构可包括重复元素（即一批类型相同的元素）。在EGM sensor protocol中，重复元素计数值最多为六个元素。

传感器协议消息

本节说明了一些传感器协议消息。请注意，可能有未全部说明的新的可选字段未包含在本手册中。

EgmHeader

EgmHeader由*EgmRobot*和*EgmSensor*共用。

```
message EgmHeader
{
optional uint32 seqno = 1; // sequence number (to be able to find
     lost messages)
optional uint32 tm = 2; // time stamp in milliseconds

enum MessageType {
MSGTYPE_UNDEFINED = 0;
MSGTYPE_COMMAND = 1; // for future use
MSGTYPE_DATA = 2; // sent by robot controller
MSGTYPE_CORRECTION = 3; // sent by sensor
}

optional MessageType mtype = 3 [default = MSGTYPE_UNDEFINED];
}
```

变量	描述
seqno	序列号。 相关应用每发送一则消息，就宜将序列号加一，这样才能检查一连串消息中是否有消息丢失。
tm	时间戳以毫秒为单位。 （可用于监测延时）。
mtype	消息类型。 相关传感器应设置成MSGTYPE_CORRECTION，相关机器人控制器应设置成MSGTYPE_DATA。

EgmRobot

```
message EgmRobot
{
optional EgmHeader              header = 1;
optional EgmFeedBack            feedBack = 2;
```

下一页继续

3.1.3 EGM传感器协议说明

续前页

```
optional EgmPlanned              planned = 3;

optional EgmMotorState           motorState = 4;
optional EgmMCIState             mciState = 5;
optional bool                    mciConvergenceMet = 6;
optional EgmTestSignals          testSignals = 7;
optional EgmRapidCtrlExecState   rapidExecState = 8;
optional EgmMeasuredForce        measuredForce = 9;
optional double                  utilizationRate=10;
}
```

变量	描述
header	请参阅EgmHeader。
feedback	反馈位置，即机器人和附加轴的测量位置。更多详细信息，请参阅EgmFeedBack。
planned	机器人和附加轴的参考位置。更多详细信息，请参阅更多EgmPlanned。
motorState	电机开启状态（开，关）
mciState[1]	内部EGM状态（运行、停止、错误）。
mciConvergenceMet[2]	显示是否满足EGM指令中定义的收敛标准的布尔值。
testSignals	仅限内部使用
rapidExecState	快速程序执行状态（运行、停止）。
measuredForce	测量的接触力。仅在力控制激活时有效。
utilizationRate	该值显示的是根据最后发送的EgmSensor包执行运动所需的可用机器人性能的百分比值。如果该值超过100%，那么控制器将无法执行完整的运动，EGM将修改参考值，使机器人只能在所需方向上执行部分运动。 如果EgmSensor包丢失或延迟，将使用最后发送的EgmPackage，这也会导致利用率暂时非常高。

EgmSensor

```
message EgmSensor
{
optional EgmHeader               header = 1;
optional EgmPlanned              planned = 2;
optional EgmSpeedRef             speedRef = 3;
}
```

变量	描述
header	请参阅EgmHeader。
planned	机器人和附加轴的参考位置。
speedRef	机器人和附加轴的参考速度。

EgmPlanned

```
message EgmPlanned
{
optional EgmJoints               joints = 1;
```

1 MCI EGM

2 MCI EGM

下一页继续

```
optional EgmPose                cartesian = 2;
optional EgmJoints external     Joints = 3;
optional EgmClock               time = 4;
}
```

变量	描述
joints	机器人的计划关节座标位置。
cartesian	机器人的计划笛卡尔座标位置。
externalJoints	外部轴的计划位置（6个值）。
time	机器人和外部轴何时安装到计划位置的时间标记。

EgmFeedBack

```
message EgmFeedBack
{
optional EgmJoints              joints = 1;
optional EgmPose                cartesian = 2;
optional EgmJoints external     Joints = 3;
optional EgmClock               time = 4;
}
```

变量	描述
joints	机器人的测量关节座标位置。
cartesian	机器人的测量笛卡尔座标位置。
externalJoints	外部轴的测量位置（6个值）。
time	机器人和外部轴何时安装到测量位置的时间标记。

3.2 构建EGM传感器通信端点

如何用.Net构建一个EGM传感器通信端点

本指南假设您会用Visual Studio进行构建和编译，并熟悉其工作方式。

此处简单描述了如何用*protobuf-csharp-port*来安装和创建一项简单的测试应用。

	操作
1	在Visual Studio中，创建一个C#应用程序。
2	选择**工具**，然后选择**NuGet包管理器**，并安装*Google.Protobuf*。
3	在**NuGet包管理器**中，同时安装*Google.Protobuf.Tools*。
4	导航至*Solution package\packages\Google.Protobuf.Tools.3.xx.x\tools\windows_x64*。
5	将*C:\ProgramData\ABB\DistributionPackages\ABB.RobotWare-7.yy\RobotPackages\RobotControl_7.zz\utility* 中的*EGM*文件夹复制到*\packages\Google.Protobuf.Tools.3.xx.x\tools\windows_x64*。
6	打开一个cmd行，运行“ *.\protoc .\egm\egm.proto --csharp_out=.\egm*”。 => Egm.cs文件以建立 **注意** *egm.proto*语法为`proto2`。
7	在Visual Studio项目中添加所生成的文件*egm.cs*（添加现有项目）。
8	将示例代码复制到Visual Studio Windows Console应用文件*(egm-sensor.cs)*中，然后依次编译、链接和执行。

如何用C++构建一个EGM传感器通信端点

用C++构建时无需其它第三方文库。

Google支持C++。在Windows中构建Google工具可能有点棘手，但这里会指导您如何构建针对Windows的protobuf。

当您已构建了*libprotobuf.lib*和*protoc.exe*时，则请使用以下程序：

	操作
1	执行Google protoc来生成访问等级`protoc --cpp_out=. egm.proto`
2	创建一项win32控制台应用。
3	添加Protobuf源来作为包含目录。
4	在相关项目中添加所生成的*egm.pb.cc*文件，并从预编译标题中排除该文件。
5	编译并执行。

3.3 配置UdpUc装置

如何配置UdpUc设备

UdpUc最多通过Udp与八件装置进行通信。这些装置会起到服务器的作用，而相关的机器人控制器则会起到一个客户端的作用，即是说该机器人控制器会初始化相关传感器的连接状况。

将各UDP通道定义为一个装置，即需要针对想要运用EGM的各运动任务，设置一个装置。

系统参数

此处简述了配置一件装置时所用的相关参数。关于这些参数的更多信息请参见技术参考手册 - 系统参数。

这些参数属于主题*Communication*下的类型*UDP Unicast Device*。

参数	描述
Name	UDP Unicast Device 实例的名称。 比如*EGMsensor*。
Type	待使用的UDP Unicast Device 协议类型。 唯一可用的 UDP Unicast Device 类型为*UDPUC*。
Remote Address	传感器等外部设备的 IP 地址。
Remote Port Number	通过*Remote Address*所识别的网络节点上的端口编号。
Local Port Number	控制器侦听广播消息的端口号。

配置示例

若要让该装置为EGM提供输入数据，那么就必须按以下方式将其配置成一件UdpUc装置：

Name	Type	Remote Address	Remote Port Number
UCdevice	UDPUC	192.168.10.20	6510

此页刻意留白

4 系统参数

4.1 类型 *External Motion Interface Data*

4.1.1 External Motion Interface Data类型

概述

本节描述的类型*External Motion Interface Data*属于主题*Motion*。关于该类型的每个参数，本节都会用一个单独的信息主题来加以说明。

类型说明

*External Motion Interface Data*类型包含的许多参数定义了一项*External Motion Interface Data*的特性。

4.1.2 Name

父级

*Name*属于主题*Motion*中的类型*External Motion Interface Data*。

描述

*External Motion Interface Data*的名称。

手册用法

这是*External Motion Interface Data*的公共标识。
修改该参数后无需重启相关控制器。

允许值

一段最多32个字符的字符串。

4.1.3 Level

父级

*Level*属于主题*Motion*中的类型*External Motion Interface Data*。

描述

*External Motion Interface Level*决定了在哪个系统等级上实行校正。

手册用法

等级值可能如下：

值：	名称	描述：
0	原物	与原物校正相对应，添加在紧靠伺服控制器的前侧
1	滤波	在校正时使用额外滤波，但这也会引入一些额外的延时和等待
2	路径	采用路径校正。

修改该参数后无需重启相关控制器。

限制

使用*Level 0*时需要低通滤波器来避免相关机器人出现振动。

允许值

允许值为*0*、*1*或*2*

默认值为1。

4.1.4 Do Not Restart after Motors Off

父级

*Do Not Restart after Motors Off*属于主题*Motion*中的类型*External Motion Interface Data*。

描述

Do Not Restart after Motors Off 决定了对紧急停止后的实例来说，是否宜在相关控制器处于“电机关闭”状态后自动重启*External Motion Interface*。

手册用法

如果为*False*（默认），那么进入“电机关闭”状态后的系统将在同一状态下继续执行相关校正。

如果为*True*，则会按“待机”状态下的所有校正来继续执行。

允许值

True或False。

4.1.5 Return to Program Position when Stopped

父级

*Return to Program Position when Stopped*属于主题*Motion*中的类型*External Motion Interface Data*。

描述

Return to Program Position when Stopped 决定了当程序执行过程停止时，正在运行*External Motion Interface*的各轴是否宜返回相应的编程位置。

手册用法

如果为*False*（默认），那么各轴将停在其当前位置处。

如果为*True*，那么各轴将移到相应的编程启动位置处。

限制

将在关节空间中定义使各轴返回各自编程位置的运动。如果将*Return to Programmed Position when Stopped*定义为*False*时各轴远离各自的编程位置，那么可能导致意料之外的抛掷，所以建议仅在用户知道编程位置与相应已校正位置相距不远的情况下，才把该数值设置成*False*。

允许值

True或False。

4.1.6 Default Ramp Time

父级

*Default Ramp Time*属于主题*Motion*中的类型*External Motion Interface Data*。

描述

*Default Ramp Time*定义了在停止*External Motion Interface*的执行过程时，停止*External Motion Interface*移动所需的默认总时间。

手册用法

该数值将用于决定两项内容：当停止程序执行过程时，*External Motion Interface*的速度贡献量宜以多快的速率减少至零；如果*Return to Programmed Position when Stopped*为*True*，那么各轴宜以多快的速率返回各自的编程位置。

此值通常可以小于 1。应在应用过程中调整并检查该值。带有大量负载且已高速运行的大型机器人将需要较高的值，而带有小量负载且低速运行的小型机器人只需要较低的值即可快速停止。

> **注意**
>
> 由于在斜坡运动期间的运动将是联合运动，因此机器人将在停止过程中偏离其当前位置和引导位置。

限制

该数值只会影响到*External Motion Interface*执行过程产生的那部分运动，而不会影响到任何同步移动（比如已在RAPID中编写好的同步移动）。

允许值

0.005 ~ 10.0秒之间的一个数值。

默认值为0.5秒。

4.1.7 Default Proportional Position Gain

父级

*Default Proportional Position Gain*属于主题*Motion*中的类型*External Motion Interface Data*。

描述

*Default Proportional Position Gain*定义了*External Motion Interface*位置反馈控制的默认比例增益。

允许值

0.0秒到20.0秒之间的一个数值。

默认值为5.0。

4.1.8 Default Low Pass Filter Bandwidth

父级

*Default Low Pass Filter Bandwidth*属于主题*Motion*中的类型*External Motion Interface Data*。

描述

*Default Low Pass Filter Bandwidth Time*定义了过滤*External Motion Interface*执行过程的速度贡献量时所用低通滤波器的默认带宽。

允许值

0.0赫兹到100.0赫兹之间的一个数值。

默认值为20.0赫兹。

注意

过低或过高的值会导致不可预测的移动。

5 RAPID 参考信息

5.1 指令：

5.1.1 EGMActJoint – 为一个关节目标点编写一次EGM移动

手册用法

EGMActJoint会激活一项特定的EGM进程，并为在传感器指引下前往某关节目标点的移动定义所需的静态数据（即那些不会因EGM移动的不同而频繁改变的数据）。

基本示例

```
VAR egmident egmID1;
PERS pose pose1:=[[0,0,0], [1,0,0,0]];
CONST egm_minmax egm_minmax1:=[-1,1];

EGMGetId egmID1;
EGMSetupAI ROB_1, egmID1, "default" \Pose \aiR1x:=ai_01
     \aiR2y:=ai_02 \aiR3z:=ai_03 \aiR4rx:=ai_04 \aiR5ry:=ai_05
     \aiR6rz:=ai_06;
EGMActJoint egmID1 \J1:=egm_minmax1 \J3:=egm_minmax1
     \J4:=egm_minmax1;
```

变元

```
EGMActJoint EGMid [\StreamStart] [\Tool] [\WObj] [\TLoad] [\J1]
     [\J2] [\J3] [\J4] [\J5] [\J6] [\J7] [\LpFilter] [\SampleRate]
     [\MaxPosDeviation] [\MaxSpeedDeviation]
```

EGMid

数据类型：egmident

EGM标识。

[\StreamStart]

数据类型：switch

StreamStart 启动向外部设备的位置数据流动。数据以使用 \SampleRate 定义的周期发送，内容符合 Google Protobuf 定义文件 *egm.proto*中的协议规范。

仅当EGM采用EGMSetupUC设置时，即协议UdpUc用于与外部设备通信时，StreamStart才可用。

[\Tool]

数据类型： tooldata

按指令EGMRunJoint来执行移动的工具。

自变数[\Tool]是一个可选项。当忽略该自变数时，其默认值为tool0。

[\Wobj]

数据类型： wobjdata

按指令EGMRunJoint来执行移动的工件。

自变数[\Wobj]是一个可选项。当忽略该自变数时，其默认值为wobj0。

下一页继续

5.1.1 EGMActJoint – 为一个关节目标点编写一次EGM移动

Externally Guided Motion

续前页

[\TLoad]

Total load

数据类型：loaddata

按指令EGMRunJoint来执行移动的负载。

自变数[\TLoad]是一个可选项。当忽略该自变数时，其默认值为load0。

\TLoad主动轴描述了移动中使用的总负载。总负载就是相关的工具负载加上该工具正在处理的有效负载。如果使用了\TLoad自变数，那么就不考虑当前tooldata中的loaddata。

如果\TLoad自变数被设置成load0，那么就不考虑\TLoad自变数，而是以当前tooldata中的loaddata作为代替。

想要使用\TLoad自变数，就必需将系统参数ModalPayLoadMode的数值设置成0。如果将ModalPayLoadMode设置成0，那么就再也无法使用指令GripLoad。

可用服务例程“负载标识”（LoadIdentify）来识别总负载。如果系统参数ModalPayLoadMode被设置成0，且系统正在运行该服务例程，那么操作员便可将相关工具的loaddata复制到一个现有的或新的loaddata永久变量中。

如果使用了关联到系统输入项SimMode（仿真模式）上的一个数字输入信号，那么便可在没有任何有效负载的情况下试运行该程序。如果该数字输入信号被设置成1，那么就不考虑可选自变数\TLoad中的loaddata，而是以当前tooldata中的loaddata作为代替。

> **注意**
>
> 处理有效负载的默认功能是使用指令GripLoad，因此系统参数ModalPayLoadMode的默认值为1。

[\J1] [\J2] [\J3] [\J4] [\J5] [\J6] [\J7]

数据类型：egm_minmax

6轴机器人关节1到6以度数为单位的收敛标准以及7轴机器人关节1到7以度数为单位的收敛标准。默认值为±0.5度。

该收敛标准数据的作用是决定相关机器人是否已抵达了预定关节位置。如果预定关节位置与实际关节位置之间的差距在egm_minmax.min和egm_minmax.max的范围之内，那么就视为该关节已抵达了其预定位置。如果并未指定某一关节的收敛标准（在EGMRunJoint中选择），那么系统就会使用默认值。

只要EGMRunJoint中指定的所有关节都抵达了各自的预定位置，相关机器人就会抵达其本身的预定位置，而RAPID则会继续执行下一条RAPID指令。

[\LpFilter]

数据类型：num

低通滤波器带宽，以赫兹（Hz）为单位，用于过滤传感器噪声。

[\SampleRate]

数据类型：num

读取输入数据的取样率（4毫秒的倍数）。有效值为4、8、12和16等。

默认值为4毫秒。

下一页继续

[\MaxPosDeviation]

数据类型：num

与编程位置之间的最大关节偏差（以度为单位），即开始EGM移动的精确点。所有关节都采用同一数值。

默认值为1000度。

[\MaxSpeedDeviation]

数据类型：num

容许的最大关节速度变化（以度 / 秒为单位）。如果某一关节受到限制，则所有关节以相同比率下降。

默认值为1.0度 / 秒。

限制

- 如果用同一EGMid执行了若干次EGMActJoint，那么各条EGMRunJoint指令就会使用最后的激活数据，直到系统执行一次新的EGMActJoint为止。
- EGMActJoint只能用在RAPID运动任务中。

语法

```
EGMActJoint
  [EGMid ':='] <variable (VAR) of egmident>
  ['\'Tool ':=' <persistent (PERS) of tooldata>]
  ['\'Wobj ':=' <persistent (PERS) of wobjdata>
  ['\'TLoad ':=' <persistent (PERS) of loaddata>]
  ['\'J1 ':=' <expression (IN) of egm_minmax>]
  ['\'J2 ':=' <expression (IN) of egm_minmax>]
  ['\'J3 ':=' <expression (IN) of egm_minmax>]
  ['\'J4 ':=' <expression (IN) of egm_minmax>]
  ['\'J5 ':=' <expression (IN) of egm_minmax>]
  ['\'J6 ':=' <expression (IN) of egm_minmax>]
  ['\'J7 ':=' <expression (IN) of egm_minmax>]
  ['\'LpFilter ':=' <expression (IN) of num>]
  ['\'SampleRate ':=' <expression (IN) of num>]
  ['\'MaxPosDeviation ':=' <expression (IN) of num>]
  ['\'MaxSpeedDeviation ':=' <expression (IN) of num>] ';'
```

相关信息

信息，关于	请参阅
指令EGMRunJoint	第60页的*EGMRunJoint – 执行一次含一个关节目标点的EGM移动*
指令EGMStreamStart	第81页的*EGMStreamStart - 启动EGM位置流*
数据类型egm_minmax	第89页的*egm_minmax – EGM的收敛标准*
MoveJ	技术参考手册 - *RAPID*指令、函数和数据类型

5.1.2 EGMActMove – 编写一次经过路径校正的EGM移动

手册用法

EGMActMove会激活一项特定的EGM进程，并为带路径校正的移动定义所需的静态数据（即不会因EGM路径校正移动的不同而频繁改变的数据）。

基本示例

下例说明了指令EGMActMove。

例 1

```
VAR egmident EGMid1;
PERS tooldata tLaser := [TRUE, [[148,50,326],
     [0.3902618,-0.589657,-0.589656,0.3902630]],
     [1,[-0.92,0,-0.39], [1,0,0,0], 0,0,0]];
EGMGetId EGMid1;
EGMSetupLTAPP ROB_1, EGMid1, "pathCorr", "OptSim", 1\LATR;
EGMActMove EGMid1, tLaser.tframe\SampleRate:=48;
```

该程序会登记一项EGM进程，然后设置一个使用通信协议LTAPP且类型为先行的传感器，并以该传感器作为数据来源（传感器）。该传感器应使用关节类型定义编号1来进行跟踪。此外也要设置控制器访问相关装置和该装置传感器框架的速率。

变元

```
EGMActMove EGMid, SensorFrame [\SampleRate]
```

EGMid

数据类型：egmident

EGM标识。

SensorFrame

数据类型：pose

传感器框架。

[\SampleRate]

数据类型：num

读取输入数据的取样率（24毫秒的倍数）。有效值为24、48和72等。

程序执行

该传感器框架和传感器取样率与一个EGM标识相关联，直到要么用EGMReset来重置它们、要么用另一条EGMActMove指令来更改它们为止。

语法

```
EGMActMove
  [EGMid ':='] <variable (VAR) of egmident> ','
  [SensorFrame ':='] < expression (IN) of pose>
  ['\'SampleRate ':=' <expression (IN) of num>] ';'
```

下一页继续

续前页

相关信息

信息，关于	请参阅
EGMReset	第*59*页的*EGMReset* - 重置一项*EGM*进程

5.1.3 EGMActPose – 为一个姿态目标点编写一次EGM移动

手册用法

EGMActPose会激活一项特定的EGM进程，并为在传感器指引下前往某姿态目标点的移动定义所需的静态数据（即那些不会因EGM移动的不同而频繁改变的数据）。

基本示例

```
VAR egmident egmID1;
PERS pose pose1:=[[0,0,0], [1,0,0,0]];
CONST egm_minmax egm_minmax_lin:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot:=[-0.1,0.2];
CONST pose posecor:=[[1200,400,900], [0,0,1,0]];
CONST pose posesens:=[[12.3313,-0.108707,416.142],
     [0.903899,-0.00320735,0.427666,0.00765917]];

EGMGetId egmID1;
EGMSetupAI ROB_1, egmID1, "default" \Pose \aiR1x:=ai_01
     \aiR2y:=ai_02 \aiR3z:=ai_03 \aiR4rx:=ai_04 \aiR5ry:=ai_05
     \aiR6rz:=ai_06;
EGMActPose egmID1 \Tool:=tool0 \WObj:=wobj0, posecor,
     EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin
     \y:=egm_minmax_lin \z:=egm_minmax_lin \rx:=egm_minmax_rot
     \ry:=egm_minmax_rot \rz:=egm_minmax_rot \LpFilter:=20;
```

变元

```
EGMActPose EGMid [\StreamStart] [\Tool] [\WObj] [\TLoad], CorrFrame,
     CorrFrType, SensorFrame, SensorFrType [\x] [\y] [\z] [\rx]
     [\ry] [\rz] [\LpFilter] [\SampleRate] [\MaxPosDeviation]
     [\MaxSpeedDeviation]
```

EGMid

数据类型：egmident

EGM标识。

[\StreamStart]

数据类型：switch

StreamStart 启动向外部设备的位置数据流动。数据以使用 \SampleRate 定义的周期发送，内容符合 Google Protobuf 定义文件 *egm.proto*中的协议规范。

仅当EGM采用EGMSetupUC设置时，即协议UdpUc用于与外部设备通信时，StreamStart才可用。

[\Tool]

数据类型： tooldata

按指令EGMRunPose来执行移动的工具。

自变数[\Tool]是一个可选项。当忽略该自变数时，其默认值为tool0。

[\Wobj]

数据类型： wobjdata

按指令EGMRunPose来执行移动的工件。

下一页继续

自变数[\Wobj]是一个可选项。当忽略该自变数时，其默认值为wobj0。

[\TLoad]

Total load

数据类型：loaddata

按指令EGMRunPose来执行移动的负载。

自变数[\TLoad]是一个可选项。当忽略该自变数时，其默认值为load0。

\TLoad主动轴描述了移动中使用的总负载。总负载就是相关的工具负载加上该工具正在处理的有效负载。如果使用了\TLoad自变数，那么就不考虑当前tooldata中的loaddata。

如果\TLoad自变数被设置成load0，那么就不考虑\TLoad自变数，而是以当前tooldata中的loaddata作为代替。

想要使用\TLoad自变数，就必需将系统参数ModalPayLoadMode的数值设置成0。如果将ModalPayLoadMode设置成0，那么就再也无法使用指令GripLoad。

可用服务例程“负载标识”（LoadIdentify）来识别总负载。如果系统参数ModalPayLoadMode被设置成0，且系统正在运行该服务例程，那么操作员便可将相关工具的loaddata复制到一个现有的或新的loaddata永久变量中。

如果使用了关联到系统输入项SimMode（仿真模式）上的一个数字输入信号，那么便可在没有任何有效负载的情况下试运行该程序。如果该数字输入信号被设置成1，那么就不考虑可选自变数\TLoad中的loaddata，而是以当前tooldata中的loaddata作为代替。

注意

处理有效负载的默认功能是使用指令GripLoad，因此系统参数ModalPayLoadMode的默认值为1。

CorrFrame

数据类型：pose

校正框架。

CorrFrType

数据类型：egmframetype

校正框架的框架类型。

SensorFrame

数据类型：pose

传感器框架。

SensFrType

数据类型：egmframetype

传感器框架的框架类型。

[\x] [\y] [\z]

数据类型：egm_minmax

x、y和z的收敛标准（以毫米为单位）。默认值为±1.0毫米。

下一页继续

5.1.3 EGMActPose – 为一个姿态目标点编写一次EGM移动

Externally Guided Motion

续前页

该收敛标准数据的作用是决定相关机器人是否已从指定的轴方向抵达了预定位置。如果预定位置与实际位置之间的差距在egm_minmax.min和egm_minmax.max的范围之内，那么就视为相关机器人已抵达了其预定位置。如果并未指定某一轴方向的收敛标准（在EGMRunPose中选择），那么系统就会使用默认值。

只要EGMRunPose中指定的所有轴都抵达了各自的预定位置，相关机器人就会抵达其本身的预定位置，而RAPID则会继续执行下一条RAPID指令。

`[\rx] [\ry] [\rz]`

数据类型：egm_minmax

x、y和z的收敛标准（以度为单位）。默认值为±0.5度。

该收敛标准数据的作用是决定相关机器人是否已沿指定轴抵达了预定方位。如果预定方位与实际方位之间的差距在egm_minmax.min和egm_minmax.max的范围之内，那么就视为相关机器人已抵达了其预定方位。如果并未指定某一轴方位的收敛标准（在EGMRunPose中选择），那么系统就会使用默认值。

只要EGMRunPose中指定的所有轴都抵达了各自的预定方位，相关机器人就会抵达其本身的预定位置，而RAPID则会继续执行下一条RAPID指令。

`[\LpFilter]`

数据类型：num

低通滤波器带宽，以赫兹（Hz）为单位，用于过滤传感器噪声。

该默认值取自EGMSetupXX指令的相关配置。

`[\SampleRate]`

数据类型：num

读取输入数据的取样率（4毫秒的倍数）。有效值为4、8、12和16等。

默认值为4毫秒。

`[\MaxPosDeviation]`

数据类型：num

与编程位置之间的最大关节偏差（以度为单位），即开始EGM移动的精确点。所有关节都采用同一数值。

默认值为1000度。

`[\MaxSpeedDeviation]`

数据类型：num

容许的最大关节速度变化（以度 / 秒为单位）。如果某一关节受到限制，则所有关节以相同比率下降。

默认值为1.0度 / 秒。

限制

- 如果用同一EGMid执行了若干次EGMActPose，那么各条EGMRunPose指令就会使用最后的激活数据，直到系统执行一次新的EGMActPose为止。
- EGMActPose只能用在RAPID运动任务中。

语法

```
EGMActPose
  [EGMid ':='] <variable (VAR) of egmident>
```

下一页继续

续前页

```
['\'Tool ':=' <persistent (PERS) of tooldata>]
['\'Wobj ':=' <persistent (PERS) of wobjdata>]
['\'TLoad ':=' <persistent (PERS) of loaddata>] ','
[CorrFrame ':='] < expression (IN) of pose> ','
[CorrFrType ':='] < expression (IN) of egmframetype> ','
[SensorFrame ':='] < expression (IN) of pose> ','
[SensorFrType ':='] < expression (IN) of egmframetype>
['\'x ':=' <expression (IN) of egm_minmax>]
['\'y ':=' <expression (IN) of egm_minmax>]
['\'z ':=' <expression (IN) of egm_minmax>]
['\'rx ':=' <expression (IN) of egm_minmax>]
['\'ry ':=' <expression (IN) of egm_minmax>]
['\'rz ':=' <expression (IN) of egm_minmax>]
['\'LpFilter ':=' <expression (IN) of num>]
['\'SampleRate ':=' <expression (IN) of num>]
['\'MaxPosDeviation ':=' <expression (IN) of num>]
['\'MaxSpeedDeviation ':=' <expression (IN) of num>] ';'
```

相关信息

信息，关于	请参阅
指令EGMRunPose	第63页的*EGMRunPose – 执行一次含一个姿态目标点的EGM移动*
指令EGMStreamStart	第81页的*EGMStreamStart - 启动EGM位置流*
数据类型egm_minmax	第89页的*egm_minmax – EGM的收敛标准*

5.1.4 EGMGetId – 获取一个EGM标识

手册用法

EGMGetId的作用是保留一个EGM标识（EGMid），之后便可在其它所有EGM RAPID指令和函数中使用该标识，从而识别关联到所属RAPID运动任务上的某一EGM进程。

egmident的标识就是其名称，即是说如果用相同的egmident来第二次或第三次调用EGMGetId，那么系统既不会保留一项新的EGM进程，也不会更改其内容。

若要释放其它EGM进程使用的egmident，则必须使用RAPID指令EGMReset。

同一时间最多能用4个不同的EGM标识。

基本示例

```
VAR egmident egmID1;
EGMGetId egmID1;
```

变元

```
EGMGetId EGMid
```

EGMid

数据类型：egmident

EGM标识。

限制

- EGMGetId只能用在RAPID运动任务中。

语法

```
EGMGetId
  [EGMid ':='] <variable (VAR) of egmident> ';'
```

相关信息

信息，关于	请参阅
EGMReset	第59页的*EGMReset – 重置一项EGM进程*

5.1.5 EGMMoveC – 经过路径校正的圆形EGM移动

手册用法

EGMMoveC的作用是沿圆弧将工具中心点（TCP）移到经过路径校正的给定目标点处。在进行这种移动时，相关姿态通常会相对于圆圈保持不变。

基本示例

下例说明了指令EGMMoveC。

例 1

```
VAR egmident EGMid1;
PERS tooldata tReg := [TRUE, [[148,0,326],
     [0.8339007,0,0.551914,0]], [1,[0,0,100], [1,0,0,0], 0,0,0]];
PERS tooldata tLaser := [TRUE, [[148,50,326],
     [0.3902618,-0.589657,-0.589656,0.3902630]],
     [1,[-0.92,0,-0.39], [1,0,0,0], 0,0,0]];
EGMGetId EGMid1;
EGMSetupLTAPP ROB_1, EGMid1, "pathCorr", "OptSim", 1\LATR;
EGMActMove EGMid1, tLaser.tframe\SampleRate:=48;
MoveL p6, v10, fine, tReg\WObj:=wobj0;
EGMMoveL EGMid1, p12, v10, z5, tReg\WObj:=wobj0;
EGMMoveL EGMid1, p7, v10, z5, tReg\WObj:=wobj0;
EGMMoveC EGMid1, p13, p14, v10, z5, tReg\WObj:=wobj0;
EGMMoveL EGMid1, p15, v10, fine, tReg\WObj:=wobj0;
MoveL p8, v1000, z10, tReg\WObj:=wobj0;
EGMReset EGMid1;
```

该程序会登记一项EGM进程，然后设置一个使用通信协议LTAPP且类型为先行的传感器，并以该传感器作为数据来源（传感器）。该传感器应使用关节类型定义编号1来进行跟踪。此外也要设置控制器访问相关装置和该装置传感器框架的速率。

相关机器人会按一条MoveL指令而移到跟踪路径的起点处。EGMMove指令会按相关传感器的校正情况来执行相应的机器人移动。

最后相关机器人会移到一个出发位置，然后释放EGM标识。

变元

```
EGMMoveC EGMid, CirPoint, ToPoint, Speed, Zone, Tool, [\Wobj]
     [\TLoad] [\NoCorr]
```

`EGMid`

数据类型：egmident

EGM标识。

`CirPoint`

数据类型：robtarget

相关机器人的圆弧点。圆弧点是指相关起点与终点间的圆弧上的某个位置。若要获得最好的准确度，则宜把该点放在相关起点与终点的正中间处。如果该点太靠近起点或终点，那么相关机器人就可能发出一条警告。将圆弧点定义为一个已命名的位置，或将其直接保存在相关指令（在指令中用一个*标注）中。勿使用外轴的这一位置。

下一页继续

5.1.5 EGMMoveC – 经过路径校正的圆形EGM移动

Externally Guided Motion

续前页

`ToPoint`

数据类型：`robtarget`

机器人和外部轴的目标点。定义为已命名的位置或直接存储在指令中（在指令中加 * 标记）。

`Speed`

数据类型： `speeddata`

适用于移动的速度数据。速度数据定义了相关TCP的速度、工具的重定方位和外轴。

`Zone`

数据类型：`zonedata`

相关移动的区域数据。区域数据描述了所生成拐角路径的大小。

`Tool`

数据类型： `tooldata`

相关机器人移动时使用的工具。工具中心点是将移过去的指定目标点。

`[\WObj]`

Work Object

数据类型： `wobjdata`

该工件（对象坐标系）与相关指令中的机器人位置相关联。

用户可以忽略该自变数，且如果忽略了该自变数，那么相关位置就会与全局坐标系关联起来。另一方面，如果使用了一个固定TCP或若干协同外轴，那么就必须为一个圆圈（相对于待执行工件的圆圈）指定该自变数。

`[\TLoad]`

Total load

数据类型：`loaddata`

`\TLoad`主动轴描述了移动中使用的总负载。总负载就是相关的工具负载加上该工具正在处理的有效负载。如果使用了`\TLoad`自变数，那么就不考虑当前`tooldata`中的`loaddata`。

如果`\TLoad`自变数被设置成`load0`，那么就不考虑`\TLoad`自变数，而是以当前`tooldata`中的`loaddata`作为代替。

想要使用`\TLoad`自变数，就必需将系统参数`ModalPayLoadMode`的数值设置成0。如果将`ModalPayLoadMode`设置成0，那么就再也无法使用指令`GripLoad`。

可用服务例程“负载标识”（LoadIdentify）来识别总负载。如果系统参数`ModalPayLoadMode`被设置成0，且系统正在运行该服务例程，那么操作员便可将相关工具的`loaddata`复制到一个现有的或新的`loaddata`永久变量中。

如果使用了关联到系统输入项`SimMode`（仿真模式）上的一个数字输入信号，那么便可在没有任何有效负载的情况下试运行该程序。如果该数字输入信号被设置成1，那么

下一页继续

就不考虑可选自变数\TLoad中的loaddata，而是以当前tooldata中的loaddata作为代替。

> **注意**
>
> 处理有效负载的默认功能是使用指令GripLoad，因此系统参数ModalPayLoadMode的默认值为1。

[\NoCorr]

数据类型：switch

关闭路径校正。

程序执行

EGMMoveC会沿一条经某传感器双重校正的编程圆弧路径移动。在移动期间，相关指令会按EGMActMove设置的速率向相关传感器请求校正数据。如果存在可选自变数\NoCorr，那么就不会向该编程路径添加校正。

错误处理

系统会生成下列可恢复错误，并在错误处理器中处理这些错误。系统变量ERRNO将被设置成：

名称	错误原因
ERR_UDPUC_COMM	与相关UdpUc装置进行通信时发生的一项错误。 在同步模式下，ERR_UDPUC_COMM始终是一个可恢复错误，可以由错误处理器处理。 在异步模式下（EGMRunX\NoWait），ERR_UDPUC_COMM始终是一个由执行EGM引发的致命错误。

限制

- EGMMoveC只能用在RAPID运动任务中。
- EGMMoveC 无法在与下列任何特殊系统事件连接的 UNDO 或 RAPID 程序中执行 PowerOn、Stop、QStop、Restart、Reset 或者 Step。

语法

```
EGMMoveC
  [GMid ':='] <variable (VAR) of egmident> ','
  [CirPoint ':='] < expression (IN) of robtarget> ','
  [ToPoint ':='] < expression (IN) of robtarget> ','
  [Speed ':='] < expression (IN) of speeddata> ','
  [Zone ':='] < expression (IN) of zonedata> ','
  [Tool ':='] < persistent (PERS) of tooldata>
  ['\'WObj ':=' < persistent (PERS) of wobjdata>]
  ['\'TLoad ':=' < persistent (PERS) of loaddata>]
  ['\'NoCorr] ';'
```

相关信息

信息，关于	请参阅
MoveC	技术参考手册 - *RAPID*指令、函数和数据类型

5.1.6 EGMMoveL – 经过路径校正的直线EGM移动

手册用法

EGMMoveL的作用是沿直线将工具中心点（TCP）移到经过路径校正的给定目标点处。当该TCP保持固定时，用户也可用该指令来重定工具方位。

基本示例

下例说明了指令EGMMoveL。

例 1

```
VAR egmident EGMid1;
PERS tooldata tReg := [TRUE, [[148,0,326],
     [0.8339007,0,0.551914,0]], [1,[0,0,100], [1,0,0,0], 0,0,0]];
PERS tooldata tLaser := [TRUE, [[148,50,326],
     [0.3902618,-0.589657,-0.589656,0.3902630]],
     [1,[-0.92,0,-0.39], [1,0,0,0], 0,0,0]];
EGMGetId EGMid1;
EGMSetupLTAPP ROB_1, EGMid1, "pathCorr", "OptSim", 1\LATR;
EGMActMove EGMid1, tLaser.tframe\SampleRate:=48;
MoveL p6, v10, fine, tReg\WObj:=wobj0;
EGMMoveL EGMid1, p12, v10, z5, tReg\WObj:=wobj0;
EGMMoveL EGMid1, p7, v10, z5, tReg\WObj:=wobj0;
EGMMoveC EGMid1, p13, p14, v10, z5, tReg\WObj:=wobj0;
EGMMoveL EGMid1, p15, v10, fine, tReg\WObj:=wobj0;
MoveL p8, v1000, z10, tReg\WObj:=wobj0;
EGMReset EGMid1;
```

该程序会登记一项EGM进程，然后设置一个使用通信协议LTAPP且类型为先行的传感器，并以该传感器作为数据来源（传感器）。该传感器应使用关节类型定义编号1来进行跟踪。此外也要设置控制器访问相关装置和该装置传感器框架的速率。

相关机器人会按一条MoveL指令而移到跟踪路径的起点处。EGMMove指令会按相关传感器的校正情况来执行相应的机器人移动。

最后相关机器人会移到一个出发位置，然后释放EGM标识。

变元

```
EGMMoveL EGMid, ToPoint, Speed, Zone, Tool, [\Wobj] [\TLoad]
     [\NoCorr]
```

EGMid

数据类型：egmident

EGM标识。

ToPoint

数据类型：robtarget

机器人和外部轴的目标点。定义为已命名的位置或直接存储在指令中（在指令中加 * 标记）。

Speed

数据类型： speeddata

适用于移动的速度数据。速度数据定义了相关TCP的速度、工具的重定方位和外轴。

下一页继续

`Zone`

数据类型：`zonedata`

相关移动的区域数据。区域数据描述了所生成拐角路径的大小。

`Tool`

数据类型： `tooldata`

相关机器人移动时使用的工具。工具中心点是将移动过去的指定目标点。

`[\WObj]`

Work Object

数据类型： `wobjdata`

该工件（对象坐标系）与相关指令中的机器人位置相关联。

用户可以忽略该自变数，且如果忽略了该自变数，那么相关位置就会与全局坐标系关联起来。另一方面，如果使用了一个固定TCP或若干协同外轴，那么就必须为一个圆圈（相对于待执行工件的圆圈）指定该自变数。

`[\TLoad]`

Total load

数据类型：`loaddata`

`\TLoad`主动轴描述了移动中使用的总负载。总负载就是相关的工具负载加上该工具正在处理的有效负载。如果使用了`\TLoad`自变数，那么就不考虑当前`tooldata`中的`loaddata`。

如果`\TLoad`自变数被设置成`load0`，那么就不考虑`\TLoad`自变数，而是以当前`tooldata`中的`loaddata`作为代替。

想要使用`\TLoad`自变数，就必需将系统参数`ModalPayLoadMode`的数值设置成0。如果将`ModalPayLoadMode`设置成0，那么就再也无法使用指令`GripLoad`。

可用服务例程“负载标识”（LoadIdentify）来识别总负载。如果系统参数`ModalPayLoadMode`被设置成0，且系统正在运行该服务例程，那么操作员便可将相关工具的`loaddata`复制到一个现有的或新的`loaddata`永久变量中。

如果使用了关联到系统输入项`SimMode`（仿真模式）上的一个数字输入信号，那么便可在没有任何有效负载的情况下试运行该程序。如果该数字输入信号被设置成1，那么就不考虑可选自变数`\TLoad`中的`loaddata`，而是以当前`tooldata`中的`loaddata`作为代替。

> **注意**
>
> 处理有效负载的默认功能是使用指令`GripLoad`，因此系统参数`ModalPayLoadMode`的默认值为1。

`[\NoCorr]`

数据类型：`switch`

关闭路径校正。

下一页继续

5.1.6 EGMMoveL – 经过路径校正的直线EGM移动

Externally Guided Motion

续前页

程序执行

EGMMoveL会沿一条经某传感器双重校正的编程直线路径移动。在移动期间，相关指令会按EGMActMove设置的速率向相关传感器请求校正数据。如果存在可选自变数\NoCorr，那么就不会向该编程路径添加校正。

错误处理

系统会生成下列可恢复错误，并在错误处理器中处理这些错误。系统变量ERRNO将被设置成：

名称	错误原因
ERR_UDPUC_COMM	与相关UdpUc装置进行通信时发生的一项错误。 在同步模式下，ERR_UDPUC_COMM始终是一个可恢复错误，可以由错误处理器处理。 在异步模式下（EGMRunX\NoWait），ERR_UDPUC_COMM始终是一个由执行EGM引发的致命错误。

限制

- EGMMoveL只能用在RAPID运动任务中。
- EGMMoveL 无法在与下列任何特殊系统事件连接的 UNDO 或 RAPID 程序中执行 PowerOn、Stop、QStop、Restart、Reset 或者 Step。

语法

```
EGMMoveL
  [EGMid ':='] <variable (VAR) of egmident> ','
  [ToPoint ':='] < expression (IN) of robtarget> ','
  [Speed ':='] < expression (IN) of speeddata> ','
  [Zone ':='] < expression (IN) of zonedata> ','
  [Tool ':='] < persistent (PERS) of tooldata>
  ['\'WObj ':=' < persistent (PERS) of wobjdata>]
  ['\'TLoad ':=' < persistent (PERS) of loaddata>]
  ['\'NoCorr] ';'
```

相关信息

信息，关于	请参阅
MoveL	技术参考手册 - *RAPID*指令、函数和数据类型

5.1.7 EGMReset – 重置一项EGM进程

手册用法

EGMReset重置了一项特定的EGM进程（EGMid），即是说取消保留。

基本示例

```
VAR egmident egmID1;
PERS pose pose1:=[[0,0,0], [1,0,0,0]];
CONST egm_minmax egm_minmax_lin:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot:=[-0.1,0.2];
CONST pose posecor:=[[1200,400,900], [0,0,1,0]];
CONST pose posesens:=[[12.3313,-0.108707,416.142],
     [0.903899,-0.00320735,0.427666,0.00765917]];

EGMGetId egmID1;
EGMSetupAI ROB_1, egmID1, "default" \Pose \aiR1x:=ai_01
     \aiR2y:=ai_02 \aiR3z:=ai_03 \aiR4rx:=ai_04 \aiR5ry:=ai_05
     \aiR6rz:=ai_06;
EGMActPose egmID1 \Tool:=tool0 \WObj:=wobj0, posecor,
     EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin
     \y:=egm_minmax_lin \z:=egm_minmax_lin \rx:=egm_minmax_rot
     \ry:=egm_minmax_rot \rz:=egm_minmax_rot \LpFilter:=20;
EGMRunPose egmID1, EGM_STOP_HOLD \x \y \z \rx \ry \rz
     \RampInTime:=0.05;
EGMReset egmID1;
```

变元

```
EGMReset EGMid
```

EGMid

数据类型：egmident

EGM标识。

语法

```
EGMReset
  [EGMid ':='] <variable (VAR) of egmident>';'
```

5.1.8 EGMRunJoint – 执行一次含一个关节目标点的EGM移动

手册用法

EGMRunJoint会从一项特定的EGM进程（EGMid）的一个精确点处执行一次受传感器指引而前往某个关节目标点的移动，并定义将要移动的关节。

基本示例

```
VAR egmident egmID1;
PERS pose pose1:=[[0,0,0],[1,0,0,0]];
CONST egm_minmax egm_minmax1:=[-1,1];

EGMGetId egmID1;
EGMSetupAI ROB_1, egmID1, "default" \Joint \aiR1x:=ai_01
     \aiR2y:=ai_02 \aiR3z:=ai_03 \aiR4rx:=ai_04 \aiR5ry:=ai_05
     \aiR6rz:=ai_06;
EGMActJoint egmID1, \J1:=egm_minmax1 \J3:=egm_minmax1
     \J4:=egm_minmax1;
EGMRunJoint egmID1, EGM_STOP_HOLD \J1 \J3 \RampInTime:=0.05;
```

变元

```
EGMRunJoint EGMid, Mode [\NoWaitCond] [\J1] [\J2] [\J3] [\J4] [\J5]
     [\J6] [\J7] [\CondTime] [\RampInTime] [\RampOutTime]
     [\PosCorrGain]
```

EGMid

数据类型：egmident

EGM标识。

Mode

数据类型：egmstopmode

定义如何结束移动（EGM_STOP_HOLD, EGM_STOP_RAMP_DOWN）

[\NoWaitCond]

数据类型：switch

如果使用该可选变元，则EGMRunJoint将在移动完成前释放RAPID程序指针。继而，必须使用RAPID指令EGMWaitCond来完成EGM Position Guidance移动。在EGMRunJoint和EGMWaitCond之间不允许使用其他移动指令。

[\J1] [\J2] [\J3] [\J4] [\J5] [\J6] [\J7]

数据类型：switch

移动6轴机器人的关节1到6以及7轴机器人的关节1到7。

[\CondTime]

数据类型：num

时间（以秒为单位），为EGMActJoint中定义的收敛标准。在视为抵达相关目标点、并由EGMRunJoint释放RAPID的执行过程来继续执行下一条指令前，必选先满足这项标准。

默认值为 1 s。

下一页继续

[\RampInTime]

数据类型：num

定义以多快的速率开始移动（以秒为单位）。

[\RampOutTime]

数据类型：num

定义以多快的速率来停止EGM。

如果参数Mode被设置成EGM_STOP_HOLD，那么该参数就毫无意义。

[\PosCorrGain]

数据类型：num

位置校正增益。为0到1之间的一个数值，默认为1。

错误处理

系统会生成下列可恢复错误，并在错误处理器中处理这些错误。系统变量ERRNO将被设置成：

名称	错误原因
ERR_UDPUC_COMM	与相关UdpUc装置进行通信时发生的一项错误。 在同步模式下，ERR_UDPUC_COMM始终是一个可恢复错误，可以由错误处理器处理。 在异步模式下（EGMRunX\NoWait），ERR_UDPUC_COMM始终是一个由执行EGM引发的致命错误。

限制

- 在首次使用EGMRunJoint前，由于已经启动过相关控制器，因此执行了RAPID的Move指令启动了控制器，机器人肯定移动过。
- EGMRunJoint移动的起点必须是一个精确点。
- EGMRunJoint只能用在RAPID运动任务中。
- 如果执行了指令EGMActPose而非EGMActJoint，那么将会出现以下致命错误：*41826 EGM mode mismatch*。
- 如果并未指定从\J1到\J7中的任何一个开关，那么系统就不会执行移动，而RAPID则会继续执行下一条RAPID指令。

语法

```
EGMRunJoint
  [EGMid ':='] <variable (VAR) of egmident> ','
  [Mode ':='] < expression (IN) of egmstopmode>
  ['\'NoWaitCond]
  ['\'J1]
  ['\'J2]
  ['\'J3]
  ['\'J4]
  ['\'J5]
  ['\'J6]
  ['\'J7]
  ['\'CondTime ':=' <expression (IN) of num>]
  ['\'RampInTime ':=' <expression (IN) of num>]
```

下一页继续

5.1.8 EGMRunJoint – 执行一次含一个关节目标点的EGM移动

Externally Guided Motion

续前页

```
['\'RampOutTime ':=' <expression (IN) of num>]
['\'PosCorrGain ':=' <expression (IN) of num>] ';'
```

相关信息

信息，关于	请参阅
egmstopmode	第*91*页的*egmstopmode – 定义EGM*所需的停止模式
MoveJ	技术参考手册 - *RAPID*指令、函数和数据类型

5.1.9 EGMRunPose – 执行一次含一个姿态目标点的EGM移动

手册用法

EGMRunPose会从一项特定的EGM进程（EGMid）的一个精确点处执行一次受传感器指引而前往某个姿态目标点的移动，并定义可能将要改动的方向和方位。

基本示例

```
VAR egmident egmID1;
PERS pose pose1:=[[0,0,0],[1,0,0,0]];
CONST egm_minmax egm_minmax_lin:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot:=[-0.1,0.2];
CONST pose posecor:=[[1200,400,900],[0,0,1,0]];
CONST pose posesens:=[[12.3313,-0.108707,416.142],
     [0.903899,-0.00320735,0.427666,0.00765917]];

EGMGetId egmID1;
EGMSetupAI ROB_1, egmID1, "default" \Pose \aiR1x:=ai_01
     \aiR2y:=ai_02 \aiR3z:=ai_03 \aiR4rx:=ai_04 \aiR5ry:=ai_05
     \aiR6rz:=ai_06;
EGMActPose egmID1 \Tool:=tool0 \WObj:=wobj0, posecor,
     EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin
     \y:=egm_minmax_lin \z:=egm_minmax_lin \rx:=egm_minmax_rot
     \ry:=egm_minmax_rot \rz:=egm_minmax_rot \LpFilter:=20;
EGMRunPose egmID1, EGM_STOP_HOLD \x \y \z \rx \ry \rz
     \RampInTime:=0.05;
```

变元

```
EGMRunPose EGMid, Mode [\NoWaitCond] [\x] [\y] [\z] [\rx] [\ry]
     [\rz] [\CondTime] [\RampInTime] [\RampOutTime] [\Offset]
     [\PosCorrGain]
```

EGMid

数据类型：egmident

EGM标识。

Mode

数据类型：egmstopmode

定义如何结束移动（EGM_STOP_HOLD, EGM_STOP_RAMP_DOWN）

[\NoWaitCond]

数据类型：switch

如果使用该可选变元，则EGMRunPose将在移动完成前释放RAPID程序指针。继而，必须使用RAPID指令EGMWaitCond来完成EGM Position Guidance移动。在EGMRunPose和EGMWaitCond之间不允许使用其他移动指令。

[\x] [\y] [\z]

数据类型：switch

x、y和z方向上的移动。

下一页继续

5.1.9 EGMRunPose – 执行一次含一个姿态目标点的EGM移动

Externally Guided Motion

续前页

[\rx] [\ry] [\rz]

数据类型：switch

围绕x、y和z轴的重定方位。

[\CondTime]

数据类型：num

时间（以秒为单位），为EGMActPose中定义的收敛标准。在视为抵达相关目标点、并由EGMRunPose释放RAPID的执行过程来继续执行下一条指令前，必选先满足这项标准。

默认值为 1 s。

[\RampInTime]

数据类型：num

定义以多快的速率开始移动（以秒为单位）。

[\RampOutTime]

数据类型：num

定义以多快的速率来停止EGM。

如果参数Mode被设置成EGM_STOP_HOLD，那么该参数就毫无意义。

[\Offset]

数据类型：pose

在相关传感器给出的数值最上方定义一项静态偏移量的可能性。

[\PosCorrGain]

数据类型：num

位置校正增益。为0到1之间的一个数值，默认为1。

错误处理

系统会生成下列可恢复错误，并在错误处理器中处理这些错误。系统变量ERRNO将被设置成：

名称	错误原因
ERR_UDPUC_COMM	与相关UdpUc装置进行通信时发生的一项错误。 在同步模式下，ERR_UDPUC_COMM始终是一个可恢复错误，可以由错误处理器处理。 在异步模式下（EGMRunX\NoWait），ERR_UDPUC_COMM始终是一个由执行EGM引发的致命错误。

限制

- 在首次使用EGMRunPose前，由于已经启动过相关控制器，因此执行了RAPID的Move指令启动了控制器，机器人肯定移动过。
- EGMRunPose移动的起点必须是一个精确点。
- EGMRunPose只能用在RAPID运动任务中。
- 如果执行了指令EGMActJoint而非EGMActPose，那么将会出现以下致命错误：*41826 EGM mode mismatch*。

下一页继续

续前页

- 如果并未指定从\x到\rz中的任何一个开关，那么系统就不会执行移动，而RAPID则会继续执行下一条RAPID指令。

语法

```
EGMRunPose
  [EGMid ':='] <variable (VAR) of egmident>','
  [Mode ':='] < expression (IN) of egmstopmode>
  ['\'NoWaitCond]
  ['\'x]
  ['\'y]
  ['\'z]
  ['\'rx]
  ['\'ry]
  ['\'rz]
  ['\'CondTime ':=' <expression (IN) of num>]
  ['\'RampInTime ':=' <expression (IN) of num>]
  ['\'RampOutTime ':=' <expression (IN) of num>]
  ['\'Offset ':=' <expression (IN) of pose>]
  ['\'PosCorrGain ':=' <expression (IN) of num>] ';'
```

相关信息

信息，关于	请参阅
egmstopmode	第91页的*egmstopmode – 定义EGM所需的停止模式*

5.1.10 EGMSetupAI – 为EGM设置模拟输入信号

手册用法

EGMSetupAI的作用是为一项特定的EGM进程（EGMid）设置模拟输入信号，以作为指引相关机器人（最多6根附加轴）前往的位置目标点数值的来源。

EGM关节模式是唯一支持7轴机器人的EGM模式。对于7轴机器人，第一个附加轴输入提供了附加机器人轴的位置。

基本示例

```
VAR egmident egmID1;

EGMGetId egmID1;
EGMSetupAI ROB_1, egmID1, "default" \Pose \aiR1x:=ai_01
     \aiR2y:=ai_02 \aiR3z:=ai_03 \aiR4rx:=ai_04 \aiR5ry:=ai_05
     \aiR6rz:=ai_06;
```

变元

```
EGMSetupAI MecUnit, EGMid, ExtConfigName [\Joint] | [\Pose] |
     [\PathCorr] [\APTR] [\aiR1x] [\aiR2y] [\aiR3z] [\aiR4rx]
     [\aiR5ry] [\aiR6rz] [\aiE1] [\aiE2] [\aiE3] [\aiE4] [\aiE5]
     [\aiE6]
```

`MecUnit`

数据类型：`mecunit`

机械单元名称。

`EGMid`

数据类型：`egmident`

EGM标识。

`ExtConfigName`

数据类型：`string`

相关系统参数中定义的外部运动接口数据的名称。

更多信息请参见 技术参考手册 - 系统参数,类型 *External Motion Interface Data*,主题 *Motion*。

`[\Joint]`

数据类型：`switch`

选择用于位置引导的关节移动。

必须有 `\Joint`、`\Pose`、或`\PathCorr`的至少一个开关。

`[\Pose]`

数据类型：`switch`

选择用于位置引导的姿势移动。

必须有 `\Joint`、`\Pose`、或`\PathCorr`的至少一个开关。

`[\PathCorr]`

数据类型：`switch`

下一页继续

选择路径纠正。

必须有 \Joint、\Pose、或\PathCorr的至少一个开关。

[\APTR]

数据类型：switch

建立一个当前点跟踪器类型的传感器，用于路径纠正。例如WeldGuide或AWC。

如果使用\PathCorr，必须要有\APTR。

[\aiR1x] [\aiR2y] [\aiR3z]

数据类型：signalai

指定为姿态移动提供x、y和z值（以毫米为单位）的信号。

指定为关节移动提供机器人关节1到3角度（以度为单位）的信号。

[\aiR4rx] [\aiR5ry] [\aiR6rz]

数据类型：signalai

指定为姿态移动提供相关机器人的x、y和z旋转值（以度为单位）的信号。

指定为关节移动提供机器人关节4到6角度（以度为单位）的信号。

[\aiE1] [\aiE2] [\aiE3] [\aiE4] [\aiE5] [\aiE6]

数据类型：signalai

指定提供附加轴关节1到6之位置的信号。

当对7轴机器人使用EGM关节模式时，\aiE1提供附加机器人轴的位置。

错误处理

系统会生成下列可恢复错误，并在错误处理器中处理这些错误。系统变量ERRNO将被设置成：

名称	错误原因
ERR_NO_ALIASIO_DEF	该信号变量是在RAPID中声明的一个变量，与用指令AliasIO在I / O配置中定义的I / O信号无关。
ERR_NORUNUNIT	与I / O设备没有接触。
ERR_SIG_NOT_VALID	无法访问该I / O信号（仅对ICI现场总线有效）。

限制

- EGMSetupAI只能用在RAPID运动任务中。
- 该机械单元必须是TCP机器人。
- 必须至少指定一个信号，否则系统会发出一项错误，并停止执行RAPID。

语法

```
EGMSetupAI
  [MecUnit ':='] <variable (VAR) of mecunit> ','
  [EGMid ':='] <variable (VAR) of egmident> ','
  [ExtConfigName ':='] <expression (IN) of string>
  [['\'Joint] | ['\'Pose] | ['\'PathCorr ]]
  ['\'APTR]
  ['\'aiR1x ':=' <variable (VAR) of signalai>]
  ['\'aiR2y ':=' <variable (VAR) of signalai>]
```

下一页继续

5.1.10 EGMSetupAI – 为EGM设置模拟输入信号

Externally Guided Motion

续前页

```
['\'aiR3z ':=' <variable (VAR) of signalai>]
['\'aiR4rx ':=' <variable (VAR) of signalai>]
['\'aiR5ry ':=' <variable (VAR) of signalai>]
['\'aiR6rz ':=' <variable (VAR) of signalai>]
['\'aiE1 ':=' <variable (VAR) of signalai>]
['\'aiE2 ':=' <variable (VAR) of signalai>]
['\'aiE3 ':=' <variable (VAR) of signalai>]
['\'aiE4 ':=' <variable (VAR) of signalai>]
['\'aiE5 ':=' <variable (VAR) of signalai>]
['\'aiE6 ':=' <variable (VAR) of signalai>] ';'
```

相关信息

信息，关于	请参阅
系统参数	第35页的系统参数

5.1.11 EGMSetupAO – 为EGM设施模拟输出信号

手册用法

EGMSetupAO的作用是为一项特定的EGM进程（EGMid）设置AO信号，以作为指引相关机器人（最多6根附加轴）前往的位置目标点数值的来源。

EGM关节模式是唯一支持7轴机器人的EGM模式。对于7轴机器人，第一个附加轴输入提供了附加机器人轴的位置。

基本示例

```
VAR egmident egmID1;

EGMGetId egmID1;
EGMSetupAO ROB_1, egmID1, "default" \Pose \aoR1x:=ao_01
     \aoR2y:=ao_02 \aoR3z:=ao_03 \aoR4rx:=ao_04 \aoR5ry:=ao_05
     \aoR6rz:=ao_06;
```

变元

```
EGMSetupAO MecUnit, EGMid, ExtConfigName [\Joint] | [\Pose] |
     [\PathCorr] [\APTR] [\aoR1x] [\aoR2y] [\aoR3Z] [\aoR4rx]
     [\aoR5ry] [\aoR6rz] [\aoE1] [\aoE2] [\aoE3] [\aoE4] [\aoE5]
     [\aoE6]
```

`MecUnit`

数据类型：`mecunit`

机械单元名称。

`EGMid`

数据类型：`egmident`

EGM标识。

`ExtConfigName`

数据类型：`string`

相关系统参数中定义的外部运动接口数据的名称。

更多信息请参见 技术参考手册 - 系统参数 ， 类型 *External Motion Interface Data*, 主题 *Motion*。

`[\Joint]`

数据类型：`switch`

选择关节移动。

至少得有开关`\Joint`或`\Pose`中的一个。

`[\Pose]`

数据类型：`switch`

选择姿态移动。

至少得有开关`\Joint`或`\Pose`中的一个。

`[\PathCorr]`

数据类型：`switch`

下一页继续

5.1.11 EGMSetupAO – 为EGM设施模拟输出信号

Externally Guided Motion

续前页

选择路径纠正。

必须有 \Joint、\Pose、或\PathCorr的至少一个开关。

[\APTR]

数据类型：switch

建立一个当前点跟踪器类型的传感器，用于路径纠正。例如WeldGuide或AWC。

如果使用\PathCorr，必须要有\APTR。

[\aoR1x] [\aoR2y] [\aoR3z]

数据类型：signalao

指定为姿态移动提供x、y和z值（以毫米为单位）的信号。

指定为关节移动提供机器人关节1到3角度（以度为单位）的信号。

[\aoR4rx] [\aoR5ry] [\aoR6rz]

数据类型：signalao

指定为姿态移动提供相关机器人的x、y和z旋转值（以度为单位）的信号。

指定为关节移动提供机器人关节4到6角度（以度为单位）的信号。

[\aoE1] [\aoE2] [\aoE3] [\aoE4] [\aoE5] [\aoE6]

数据类型：signalao

指定提供附加轴关节1到6之位置的信号。

当对7轴机器人使用EGM关节模式时，\aoE1提供附加机器人轴的位置。

错误处理

系统会生成下列可恢复错误，并在错误处理器中处理这些错误。系统变量ERRNO将被设置成：

名称	错误原因
ERR_NO_ALIASIO_DEF	该信号变量是在RAPID中声明的一个变量，与用指令AliasIO在I / O配置中定义的I / O信号无关。
ERR_NORUNUNIT	与I / O设备没有接触。
ERR_SIG_NOT_VALID	无法访问该I / O信号（仅对ICI现场总线有效）。

限制

- EGMSetupAO只能用在RAPID运动任务中。
- 该机械单元必须是TCP机器人。
- 必须至少指定一个信号，否则系统会发出一项错误，并停止执行RAPID。

语法

```
EGMSetupAO
  [MecUnit ':='] <variable (VAR) of mecunit> ','
  [EGMid ':='] <variable (VAR) of egmident> ','
  [ExtConfigName ':='] <expression (IN) of string>
  [['\'Joint] | ['\'Pose] | ['\'PathCorr]]
  ['\'APTR]
  ['\'aoR1x ':=' <variable (VAR) of signalao>]
  ['\'aoR2y ':='] <variable (VAR) of signalao>]
```

下一页继续

续前页

```
['\'aoR3z ':='] <variable (VAR) of signalao>]
['\'aoR4rx ':='] <variable (VAR) of signalao>]
['\'aoR5ry ':='] <variable (VAR) of signalao>]
['\'aoR6rz ':='] <variable (VAR) of signalao>]
['\'aoE1 ':='] <variable (VAR) of signalao>]
['\'aoE2 ':='] <variable (VAR) of signalao>]
['\'aoE3 ':='] <variable (VAR) of signalao>]
['\'aoE4 ':='] <variable (VAR) of signalao>]
['\'aoE5 ':='] <variable (VAR) of signalao>]
['\'aoE6 ':='] <variable (VAR) of signalao>] ';'
```

相关信息

信息，关于	请参阅
系统参数	第35页的系统参数

5.1.12 EGMSetupGI – 为EGM设置编组输入信号

手册用法

EGMSetupGI的作用是为一项特定的EGM进程（EGMid）设置编组输入信号，以作为指引相关机器人（最多6根附加轴）前往的位置目标点数值的来源。

EGM关节模式是唯一支持7轴机器人的EGM模式。对于7轴机器人，第一个附加轴输入提供了附加机器人轴的位置。

基本示例

```
VAR egmident egmID1;

EGMGetId egmID1;
EGMSetupGI ROB_1, egmID1, "default" \Pose \giR1x:=gi_01
     \giR2y:=gi_02 \giR3z:=gi_03 \giR4rx:=gi_04 \giR5ry:=gi_05
     \giR6rz:=gi_06;
```

变元

```
EGMSetupGI MecUnit, EGMid, ExtConfigName [\Joint] | [\Pose] |
     [\PathCorr] [\APTR] [\giR1x] [\giR2y] [\giR3Z] [\giR4rx]
     [\giR5ry] [\giR6rz] [\giE1] [\giE2] [\giE3] [\giE4] [\giE5]
     [\giE6]
```

MecUnit

数据类型：mecunit

机械单元名称。

EGMid

数据类型：egmident

EGM标识。

ExtConfigName

数据类型：string

相关系统参数中定义的外部运动接口数据的名称。

更多信息请参见 技术参考手册 - 系统参数,类型 *External Motion Interface Data*,主题 *Motion*。

[\Joint]

数据类型：switch

选择关节移动。

至少得有开关\Joint或\Pose中的一个。

[\Pose]

数据类型：switch

选择姿态移动。

至少得有开关\Joint或\Pose中的一个。

[\PathCorr]

数据类型：switch

下一页继续

选择路径纠正。

必须有 \Joint、\Pose、或\PathCorr的至少一个开关。

[\APTR]

数据类型：switch

建立一个当前点跟踪器类型的传感器，用于路径纠正。例如WeldGuide或AWC。

如果使用\PathCorr，必须要有\APTR。

[\giR1x] [\giR2y] [\giR3z]

数据类型：signalgi

指定为姿态移动提供x、y和z值（以毫米为单位）的信号。

指定为关节移动提供机器人关节1到3角度（以度为单位）的信号。

[\giR4rx] [\giR5ry] [\giR6rz]

数据类型：signalgi

指定为姿态移动提供相关机器人的x、y和z旋转值（以度为单位）的信号。

指定为关节移动提供机器人关节4到6角度（以度为单位）的信号。

[\giE1] [\giE2] [\giE3] [\giE4] [\giE5] [\giE6]

数据类型：signalgi

指定提供附加轴关节1到6之位置的信号。

当对7轴机器人使用EGM关节模式时，\giE1提供附加机器人轴的位置。

错误处理

系统会生成下列可恢复错误，并在错误处理器中处理这些错误。系统变量ERRNO将被设置成：

名称	错误原因
ERR_NO_ALIASIO_DEF	该信号变量是在RAPID中声明的一个变量，与用指令AliasIO在I / O配置中定义的I / O信号无关。
ERR_NORUNUNIT	与I / O设备没有接触。
ERR_SIG_NOT_VALID	无法访问该I / O信号（仅对ICI现场总线有效）。

限制

- EGMSetupGI只能用在RAPID运动任务中。
- 该机械单元必须是TCP机器人。
- 编组信号只能处理正值，因此它们在EGM中用处有限。
- 必须至少指定一个信号，否则系统会发出一项错误，并停止执行RAPID。

语法

```
EGMSetupGI
  [MecUnit ':='] <variable (VAR) of mecunit> ','
  [EGMid ':='] <variable (VAR) of egmident> ','
  [ExtConfigName ':='] <expression (IN) of string>
  [['\'Joint] | ['\'Pose] | ['\'PathCorr]]
  ['\'APTR]
  ['\'giR1x ':=' <variable (VAR) of signalgi>]
```

下一页继续

5.1.12 EGMSetupGI – 为EGM设置编组输入信号

Externally Guided Motion

续前页

```
['\'giR2y ':=' <variable (VAR) of signalgi>]
['\'giR3z ':=' <variable (VAR) of signalgi>]
['\'giR4rx ':=' <variable (VAR) of signalgi>]
['\'giR5ry ':=' <variable (VAR) of signalgi>]
['\'giR6rz ':=' <variable (VAR) of signalgi>]
['\'giE1 ':=' <variable (VAR) of signalgi>]
['\'giE2 ':=' <variable (VAR) of signalgi>]
['\'giE3 ':=' <variable (VAR) of signalgi>]
['\'giE4 ':=' <variable (VAR) of signalgi>]
['\'giE5 ':=' <variable (VAR) of signalgi>]
['\'giE6 ':=' <variable (VAR) of signalgi>] ';'
```

相关信息

信息，关于	请参阅
系统参数	第35页的系统参数

5.1.13 EGMSetupLTAPP - 为EGM设置相应的LTAPP协议

手册用法

EGMSetupLTAPP的作用是为一项特定的EGM进程（EGMid）设置一项*LTAPP*协议，以作为路径校正的来源。

基本示例

下例说明了指令EGMSetupLTAPP。

例 1

```
VAR egmident EGMid1;
EGMGetId EGMid1;
EGMSetupLTAPP ROB_1, EGMid1, "pathCorr", "OptSim", 1\LATR;
```

该程序会登记一项EGM进程，然后设置一个使用通信协议LTAPP且类型为先行的OptSim传感器，并以该传感器作为数据来源（传感器）。该传感器应使用关节类型定义编号1来进行跟踪。

变元

```
EGMActMove MecUnit, EGMid, ExtConfigName, Device, JointType [\APTR]
     | [\LATR]
```

`MecUnit`

数据类型：mecunit

机械单元名称。

`EGMid`

数据类型：egmident

EGM标识。

`ExtConfigName`

数据类型：string

相关系统参数中定义的外部运动接口数据的名称。

更多信息请参见 技术参考手册 - 系统参数, 主题 *Motion*, 类型 *External Motion Interface Data*。

`Device`

数据类型：string

LTAPP装置名称。

`JointType`

数据类型：num

定义路径校正期间相关传感器设备应采用的关节类型（用一个数字表达）。

`[\APTR]`

数据类型：switch

建立一个当前点跟踪器类型的传感器，用于路径纠正。例如WeldGuide或AWC。

\APTR或\LATR中至少要有一个。

下一页继续

5.1.13 EGMSetupLTAPP - 为EGM设置相应的LTAPP协议

Externally Guided Motion

续前页

[\LATR]

数据类型：switch

建立一个智能预测跟踪器类型的传感器，用于路径纠正。例如Laser Tracker。

\APTR或\LATR中至少要有一个。

程序执行

EGMSetupLTAPP会关联到用于某个EGM标识使用的传感器特征数据上，之后用户便可在不同的EGMActMove和EGMMove指令中使用该EGM标识。

语法

```
EGMSetupLTAPP
  [MecUnit ':='] <variable (VAR) of mecunit> ','
  [EGMid ':='] <variable (VAR) of egmident> ','
  [ExtConfigName ':='] < expression (IN) of string> ','
  [Device ':='] < expression (IN) of string> ','
  [JointType ':='] < expression (IN) of num>
  [['\'APTR] | ['\'LATR]] ';'
```

5.1.14 EGMSetupUC – 为EGM设置UdpUc协议

手册用法

EGMSetupUC的作用是为一项特定的EGM进程（EGMid）设置UdpUc协议，以作为指引相关机器人（最多6根附加轴）前往的位置目标点数值的来源。

EGM关节模式是唯一支持7轴机器人的EGM模式。对于7轴机器人，第一个附加轴输入提供了附加机器人轴的位置。

基本示例

```
VAR egmident egmID1;
VAR string egmSensor:="egmSensor:";
EGMGetId egmID1;
EGMSetupUC ROB_1, egmID1, "default", egmSensor\Pose;
```

变元

```
EGMSetupUC MecUnit, EGMid, ExtConfigName, UCDevice [\Joint] |
     [\Pose] | [\PathCorr] [\APTR] | [\LATR] [\CommTimeout]
```

`MecUnit`

数据类型：`mecunit`

机械单元名称。

`EGMid`

数据类型：`egmident`

EGM标识。

`ExtConfigName`

数据类型：`string`

相关系统参数中定义的外部运动接口数据的名称。

更多信息请参见 技术参考手册 - 系统参数 ， 类型 *External Motion Interface Data*, 主题 *Motion*。

`UCDevice`

数据类型：`string`

UdpUc装置名称。

`[\Joint]`

数据类型：`switch`

选择用于位置引导的关节移动。

必须有 `\Joint`、`\Pose`、或`\PathCorr`的至少一个开关。

`[\Pose]`

数据类型：`switch`

选择用于位置引导的姿势移动。

必须有 `\Joint`、`\Pose`、或`\PathCorr`的至少一个开关。

`[\PathCorr]`

数据类型：`switch`

下一页继续

5.1.14 EGMSetupUC – 为EGM设置UdpUc协议

Externally Guided Motion

续前页

选择路径纠正。

必须有 \Joint、\Pose、或\PathCorr的至少一个开关。

[\APTR]

数据类型：switch

建立一个当前点跟踪器类型的传感器，用于路径纠正。例如WeldGuide或AWC。

\APTR或\LATR中至少要有一个。

[\LATR]

数据类型：switch

建立一个智能预测跟踪器类型的传感器，用于路径纠正。例如Laser Tracker。

\APTR或\LATR中至少要有一个。

[\CommTimeout]

数据类型：num

与外部UdpUC装置通信的超时时间（以秒为单位）。

错误处理

系统会生成下列可恢复错误，并在错误处理器中处理这些错误。系统变量ERRNO将被设置成：

名称	错误原因
ERR_UDPUC_COMM	与相关UdpUc装置进行通信时发生的一项错误。 在同步模式下，ERR_UDPUC_COMM始终是一个可恢复错误，可以由错误处理器处理。 在异步模式下（EGMRunX\NoWait），ERR_UDPUC_COMM始终是一个由执行EGM引发的致命错误。

限制

- EGMSetupUC只能用在RAPID运动任务中。
- 该机械单元必须是TCP机器人。

语法

```
EGMSetupUC
  [MecUnit ':='] <variable (VAR) of mecunit> ','
  [EGMid ':='] <variable (VAR) of egmident> ','
  [ExtConfigName ':='] <expression (IN) of string> ','
  [UCDevice ':='] <expression (IN) of string>
  [['\'Joint] | ['\'Pose] | ['\'PathCorr]]
  [['\'APTR] | ['\'LATR]]
  ['\'CommTimeout ':=' <expression (IN) of num>] ';'
```

相关信息

信息，关于	请参阅
系统参数	第35页的系统参数

5.1.15 EGMStop – 停止一次EGM移动

手册用法

EGMStop会停止一项特定的EGM进程（EGMid）。

基本示例

在RAPID运动任务中：

```
VAR egmident egmID1;
PERS pose pose1:=[[0,0,0], [1,0,0,0]];
CONST egm_minmax egm_minmax_lin:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot:=[-0.1,0.2];
CONST pose posecor:=[[1200,400,900], [0,0,1,0]];
CONST pose posesens:=[[12.3313,-0.108707,416.142],
     [0.903899,-0.00320735,0.427666,0.00765917]];

EGMGetId egmID1;
EGMSetupAI ROB_1, egmID1 \Pose \aiR1x:=ai_01 \aiR2y:=ai_02
     \aiR3z:=ai_03 \aiR4rx:=ai_04 \aiR5ry:=ai_05 \aiR6rz:=ai_06;
EGMActPose egmID1 \Tool:=tool0 \WObj:=wobj0, posecor,
     EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin
     \y:=egm_minmax_lin \z:=egm_minmax_lin \rx:=egm_minmax_rot
     \ry:=egm_minmax_rot \rz:=egm_minmax_rot \LpFilter:=20;
EGMRunPose egmID1, EGM_STOP_HOLD \x \y \z \rx \ry \rz
     \RampInTime:=0.05;
```

在TRAP例程中：

```
EGMStop egmID1, EGM_STOP_HOLD;
```

变元

```
EGMStop EGMid, Mode [\RampOutTime]
```

EGMid

数据类型：egmident

EGM标识。

Mode

数据类型：egmstopmode

定义如何结束移动（EGM_STOP_HOLD, EGM_STOP_RAMP_DOWN）

[\RampOutTime]

数据类型：num

定义以多快的速率来停止EGM。

如果参数Mode被设置成EGM_STOP_HOLD，那么该参数就毫无意义。

限制

- EGMStop只能用在RAPID运动任务中。

下一页继续

5.1.15 EGMStop – 停止一次EGM移动

Externally Guided Motion

续前页

语法

```
EGMStop
  [EGMid ':='] <variable (VAR) of egmident>','
  [Mode ':='] < expression (IN) of egmstopmode>
  ['\'RampOutTime ':=' <expression (IN) of num>] ';'
```

5.1.16 EGMStreamStart - 启动EGM位置流

手册用法

EGMStreamStart启动特定EGM过程（EGMid）的位置数据流动。

基本示例

下列示例说明了指令EGMStreamStart。

例 1

```
VAR egmident egmID1;

EGMGetId egmID1;
EGMSetupUC ROB_1, egmID1, "default", "UCdevice"\Joint;
EGMStreamStart egmID;
MoveAbsJ jpos20, v100, z20, Weldgun;
MoveAbsJ jpos10\NoEOffs, v1000, fine, Weldgun;
EGMStreamStop egmID1;
EGMReset egmID1;
```

变元

```
EGMStreamStart EGMid [\SampleRate];
```

EGMid

数据类型：egmident

EGM标识。

[\SampleRate]

数据类型：num

读取输入数据的取样率（4毫秒的倍数）。有效值为4、8、12和16等。

默认值为4毫秒。

程序执行

EGMStreamStart启动向外部设备的位置数据流动。数据以使用\SampleRate定义的周期发送，内容符合Google Protobuf定义文件*egm.proto*中的协议规范。

限制

仅当EGM采用EGMSetupUC设置时，即协议UdpUc用于与外部设备通信时，EGMStreamStart才可用。

语法

```
EGMStreamStart
  [EGMid ':='] <variable (VAR) of egmident>';'
  ['\' SampleRate ':=' <expression (IN) of num>]','
```

5.1.17 EGMStreamStop - 停止EGM位置流

手册用法

EGMStreamStop停止特定EGM过程（EGMid）的位置数据流动。

基本示例

下列示例说明了指令EGMStreamStop。

例 1

```
VAR egmident egmID1;

EGMGetId egmID1;
EGMSetupUC ROB_1, egmID1, "default", "UCdevice"\Joint;
EGMStreamStart egmID;
MoveAbsJ jpos20, v100, z20, Weldgun;
MoveAbsJ jpos10\NoEOffs, v1000, fine, Weldgun;
EGMStreamStop egmID1;
EGMReset egmID1;
```

变元

```
EGMStreamStop EGMid;
```

EGMid

数据类型：egmident

EGM标识。

程序执行

EGMStreamStop停止向外部设备的位置数据流动。

限制

仅当EGM采用EGMSetupUC设置时，即协议UdpUc用于与外部设备通信时，EGMStreamStop 才可用。

语法

```
EGMStreamStop
  [EGMid ':='] <variable (VAR) of egmident>';'
```

5.1.18 EGMWaitCond - 等待EGM过程

手册用法

EGMWaitCond用于等待特定EGM过程（EGMid）。

基本示例

下列示例说明了指令EGMWaitCond。

例 1

```
VAR egmident egmID1;
PERS pose pose1:=[[0,0,0],[1,0,0,0]];
CONST egm_minmax egm_minmax_lin:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot:=[-0.1,0.2];
CONST pose posecor:=[[1200,400,900],[0,0,1,0]];
CONST pose posesens:=[[12.3313,-0.108707,416.142],
     [0.903899,-0.00320735,0.427666,0.00765917]];

EGMGetId egmID1;
EGMSetupAI ROB_1, egmID1, "default" \Pose \aiR1x:=ai_01
     \aiR2y:=ai_02 \aiR3z:=ai_03 \aiR4rx:=ai_04 \aiR5ry:=ai_05
     \aiR6rz:=ai_06;
EGMActPose egmID1 \Tool:=tool0 \WObj:=wobj0, posecor,
     EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin
     \y:=egm_minmax_lin \z:=egm_minmax_lin \rx:=egm_minmax_rot
     \ry:=egm_minmax_rot \rz:=egm_minmax_rot \LpFilter:=20;
EGMRunPose egmID1, EGM_STOP_HOLD \x \y \z \rx \ry
     \rz\RampInTime:=0.05;
SetDO doSignal1, 1;
…
EGMWaitCond
```

变元

```
EGMWaitCond EGMid;
```

EGMid

数据类型：egmident

EGM标识。

程序执行

EGMWaitCond将等待EGMRunJoint/Pose指令完成。如果移动在EGMWaitCond运行之前完成，则程序执行将继续，并立即生成下一条RAPID指令。

限制

如果EGMRunJoint或EGMRunPose与可选变元\NoWaitCond一起使用，则在使用EGMWaitCond完成EGM Position Guidance之前不得使用任何移动指令。

下一页继续

5.1.18 EGMWaitCond - 等待EGM过程

Externally Guided Motion

续前页

语法

```
EGMWaitCond
  [EGMid ':='] <variable (VAR) of egmident>';'
```

5.2 函数

5.2.1 EGMGetState – 获取当前的EGM状态

手册用法

EGMGetState会检索一项EGM进程（EGMid）的状态。

基本示例

```
VAR egmident egmID1;
VAR egmstate egmState1:= EGM_STATE_DISCONNECTED;

EGMGetId egmID1;
egmState1 := EGMGetState(egmID1);
```

返回值

数据类型：egmstate

该自变数中指定的EGM标识所识别出的EGM进程的当前状态。

变元

```
EGMGetState (EGMid)
```

EGMid

数据类型：egmident

EGM标识。

限制

- EGMGetState只能用在RAPID运动任务中。
- 该机械单元必须是TCP机器人。

语法

```
EGMGetState '('
  [EGMid ':='] < variable (VAR) of egmident >')'
```

相关信息

信息，关于	请参阅
egmstate	第90页的*egmstate – 定义EGM所需的状态*

5.3 数据类型

5.3.1 egmframetype - 定义EGM所需的框架类型

手册用法

egmframetype的作用是为EGM中的校正和传感器测量定义所需的框架类型。

描述

egmframetype旨在供指令EGMActJ和EGMActPose使用。

基本示例

```
CONST egm_minmax egm_minmax_lin1:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot1:=[-0.1,0.2];

EGMActPose egmID1\Tool:=tFroniusCMT\WObj:=wobj0, posecor,
     EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin
     \y:=egm_minmax_lin \z:=egm_minmax_lin \rx:=egm_minmax_rot
     \ry:=egm_minmax_rot \rz:=egm_minmax_rot \LpFilter:=20;
```

预定义数值

值	描述
EGM_FRAME_BASE	以相对于基本框架的方式来定义该框架（姿态模式）。
EGM_FRAME_TOOL	以相对于所用工具的方式来定义该框架（姿态模式）。
EGM_FRAME_WOBJ	以相对于所用工件的方式来定义该框架（姿态模式）。
EGM_FRAME_WORLD	以相对于全局框架的方式来定义该框架（姿态模式）。
EGM_FRAME_JOINT	这些数值为关节值（关节模式）。

特征

egmframetype是针对num的一种别名数据类型。

相关信息

信息，关于	请参阅
EGMActJ	第43页的*EGMActJoint - 为一个关节目标点编写一次EGM移动*
EGMActPose	第48页的*EGMActPose - 为一个姿态目标点编写一次EGM移动*

5.3.2 egmident – 识别一项特定的EGM进程

手册用法

egmident会识别一项特定的EGM进程。

描述

用指令EGMGetId保留某一egmident，然后用其识别指令EGMSetupXX、EGMActX、EGMRunX和EGMReset，并将这些指令与同一项EGM操作关联起来。

egmident的标识就是其名称，即是说如果用相同的egmident来第二次或第三次调用EGMGetId，那么系统既不会保留一项新进程，也不会更改其内容。只有EGMReset才会释放一项egmident。

基本示例

```
VAR egmident egmID1;
VAR egmstate egmSt1;
TASK PERS wobjdata wobj_EGM1:=[FALSE, TRUE, "", [[500,700,900],
     [1,0,0,0]], [[0,0,0], [1,0,0,0]]];
CONST pose posecor:=[[1200,400,900], [0,0,1,0]];
CONST pose posesens:=[[12.3313,-0.108707,416.142],
     [0.903899,-0.00320735,0.427666,0.00765917]];
CONST egm_minmax egm_minmax_lin1:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot1:=[-0.1,0.2];
CONST egm_minmax egm_minmax_joint1:=[-0.1,0.1];

PROC testAI()
  EGMReset egmID1;
  EGMGetId egmID1;
  mvHome;
  mvHome_EGMLinear;

  egmSt1:=EGMGetState(egmID1);
  TPWrite "EGM state 1: " \Num:=egmSt1;

  IF egmSt1<=EGM_STATE_CONNECTED THEN
    EGMSetupAI ROB_1, egmID1, "default" \Pose \aiR1x:=ai_MoveX
         \aiR2y:=ai_MoveY \aiR3z:=ai_MoveZ \aiR5ry:=ai_RotY
         \aiR6rz:=ai_RotZ;
  ENDIF

  EGMActPose egmID1 \Tool:=tFroniusCMT \WObj:=wobj0, posecor,
       EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin1
       \y:=egm_minmax_lin1 \z:=egm_minmax_lin1 \rx:=egm_minmax_rot1
       \ry:=egm_minmax_rot1 \rz:=egm_minmax_rot1 \LpFilter:=20;
  EGMRunPose egmID1, EGM_STOP_HOLD \x \y \z \rx \ry \rz
       \RampInTime:=0.05;

  egmSt1:=EGMGetState(egmID1);
  IF egmSt1=EGM_STATE_CONNECTED THEN
    TPWrite "Reset lin 1";
    EGMReset egmID1;
```

下一页继续

5.3.2 egmident - 识别一项特定的EGM进程

Externally Guided Motion

续前页

```
        ENDIF
    ENDPROC
```

限制

每项RAPID任务最多有4个并行实例可用。

特征

`egmident`是一种非数值的数据类型。请调用`EGMGetId`来设置该类型。

相关信息

信息，关于	请参阅
`EGMGetId`	第52页的*EGMGetId - 获取一个EGM标识*

5.3.3 egm_minmax - EGM的收敛标准

手册用法

egm_minmax的作用是定义结束EGM的收敛标准。

描述

egm_minmax旨在供指令EGMActJ和EGMActPose使用。

组件

Min

数据类型：num

最小偏差

定义相关位置偏差的最小值。默认值为-0.5度。

Max

数据类型：num

最大偏差

定义相关位置偏差的最大值。默认值为0.5度。

基本示例

```
CONST egm_minmax egm_minmax_lin1:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot1:=[-0.1,0.2];

EGMActPose egmID1\Tool:=tFroniusCMT\WObj:=wobj0, posecor,
     EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin1
     \y:=egm_minmax_lin1 \z:=egm_minmax_lin1 \rx:=egm_minmax_rot1
     \ry:=egm_minmax_rot1 \rz:=egm_minmax_rot1 \LpFilter:=20;
```

特征

Egm_minmax具有下列单位：

- **毫米，用于直线移动下的x、y和z。**
- **度，用于直线移动和关节移动下的rx、ry和rz。**

结构

```
< dataobject of egm_minmax >
  < min of num >
  < max of num >
```

相关信息

信息，关于	请参阅
EGMActJ	第43页的*EGMActJoint - 为一个关节目标点编写一次EGM移动*
EGMActPose	第48页的*EGMActPose - 为一个姿态目标点编写一次EGM移动*

5.3.4 egmstate – 定义EGM所需的状态

手册用法

`egmstate`的作用是为EGM中的校正和传感器测量定义所需的状态。

描述

`egmstate`是函数`EGMGetState`的返回值。

基本示例

```
VAR egmstate egmSt1;
VAR egmident egmID1;

EGMReset egmID1;
EGMGetId egmID1;

egmSt1:=EGMGetState(egmID1);
TPWrite "EGM state: "\Num:=egmSt1;
```

预定义数值

值	描述
`EGM_STATE_DISCONNECTED`	未定义这一具体进程的EGM状态。 未激活任何设定。
`EGM_STATE_CONNECTED`	未激活指定的EGM进程。 已进行过设置，但并未激活任何EGM移动。
`EGM_STATE_RUNNING`	正在执行指定的EGM进程。 EGM移动处于激活状态，即是说移动了相关机器人。

特征

`egmstate`是针对`num`的一种别名数据类型。

相关信息

信息，关于	请参阅
`EGMGetState`	第*85*页的*EGMGetState – 获取当前的EGM状态*

5.3.5 egmstopmode – 定义EGM所需的停止模式

手册用法

egmstopmode的作用是为EGM中的校正和传感器测量定义所需的停止模式。

描述

egmstopmode旨在供指令EGMRunJoint、EGMRunPose和EGMStop使用。

基本示例

来自RAPID运动任务：

```
VAR egmstate egmSt1;
VAR egmident egmID1;

EGMReset egmID1;
EGMGetId egmID1;
CONST egm_minmax egm_minmax_lin1:=[-0.1,0.1];
CONST egm_minmax egm_minmax_rot1:=[-0.1,0.2];

EGMActPose egmID1 \Tool:=tFroniusCMT \WObj:=wobj0, posecor,
     EGM_FRAME_WOBJ, posesens, EGM_FRAME_TOOL \x:=egm_minmax_lin
     \y:=egm_minmax_lin \z:=egm_minmax_lin \rx:=egm_minmax_rot
     \ry:=egm_minmax_rot \rz:=egm_minmax_rot \LpFilter:=20;
EGMRunPose egmID1, EGM_STOP_HOLD \x \y \z \rx \ry \rz
     \RampInTime:=0.05;
```

来自一项RAPID软中断或后台任务：

```
EGMStop egmID1, EGM_STOP_RAMP_DOWN\RampOutTime:=5.0;
```

预定义数值

值	描述
EGM_STOP_HOLD	保持EGM结束位置。
EGM_STOP_RAMP_DOWN	从EGM的结束位置返回启动位置。

特征

egmstopmode是针对num的一种别名数据类型。

相关信息

信息，关于	请参阅
EGMRunJoint	第*60*页的*EGMRunJoint – 执行一次含一个关节目标点的EGM移动*
EGMRunPose	第*63*页的*EGMRunPose – 执行一次含一个姿态目标点的EGM移动*
EGMStop	第*79*页的*EGMStop – 停止一次EGM移动*

5.4 代码示例

5.4.1 使用EGM位置流

描述

若要让该装置为EGM提供输入数据，那么就必须首先将其配置成一件UdpUc装置。具体请参见第33页的如何配置UdpUc设备。

该装置目前可为EGM所用，以将机械单元位置传至外部设备。下面展示了一些简单的例子。

有可能传输几个运动任务的位置，但必须为各运动任务配备一个通信通道。

示例

将EGMStreamStart和EGMStreamStop用于一个机械单元

此方法是使用EGM Position Stream的最简单方法，但对于Absolute Accuracy或重载的机器人来说并不准确。

```
VAR egmident egmID1;
EGMGetId egmID1;
! Set up the EGM data source: UdpUc server using device "UCdevice"
     and configuration "default"
EGMSetupUC ROB_1, egmID1, "default", "UCdevice"\Joint;
! Start the position stream for T_ROB1 including active external
     axis. Cycle time is 16 ms.
EGMStreamStart\SampleRate:=16 egmID;
! Run your program – streaming is active
MoveAbsJ jpos20, v100, z20, Weldgun;
…
…
MoveAbsJ jpos10\NoEOffs, v1000, fine, Weldgun;
! Stop the position stream – but it is not necessary if you want
     to stream until the controller shuts down
EGMStreamStop egmID1;
EGMReset egmID1;
```

将EGMActXX \StreamStart用于一个机械单元

如果您有一个Absolute Accuracy的机器人，因为RAPID指令EGMActPose和EGMActJoint将工具和加载的数据传递给控制器，所以优先采用这种方法。

```
VAR egmident egmID1;
! Used tool
TASK PERS tooldata Weldgun:=[TRUE,[[12.3313,-0.108707,416.142],
  [0.903899,-0.00320735,0.427666,0.00765917]],
  [2.6,[-111.1,24.6,386.6],[1,0,0,0],0,0,0.072]];
! limits for cartesian convergence: +-1 mm
CONST egm_minmax egm_minmax_lin1:=[-1,1];
! limits for orientation convergence: +-2 degrees
CONST egm_minmax egm_minmax_rot1:=[-2,2];
! Correction frame offset: none
VAR pose corr_frame_offs:=[[0,0,0],[1,0,0,0]];
EGMGetId egmID1;
```

下一页继续

```
! Set up the EGM data source: UdpUc server using device "UCdevice"
     and configuration "default"
EGMSetupUC ROB_1, egmID1, "default", "UCdevice"\Joint;
! Correction frame is the World coordinate system and the sensor
     measurements are relative to the tool frame of the used tool
     (Weldgun). Start the position stream for T_ROB1 including
     active external axis. Cycle time is 16 ms.
EGMActPose egmID1\StreamStart\Tool:= Weldgun, corr_frame_offs,
EGM_FRAME_WORLD, Weldgun.tframe, EGM_FRAME_TOOL
\x:=egm_minmax_lin1\y:=egm_minmax_lin1\z:=egm_minmax_lin1
\rx:=egm_minmax_rot1\ry:=egm_minmax_rot1\rz:=egm_minmax_rot1
\LpFilter:=20;
! Run your program – streaming is active
MoveAbsJ jpos20, v100, z20, Weldgun;
…
…
MoveAbsJ jpos10\NoEOffs, v1000, fine, Weldgun;
! Stop the position stream – but this is not necessary if you want
     to stream until the controller shuts down
EGMStreamStop egmID1;
EGMReset egmID1;
```

将EGMStreamStart和EGMStreamStop用于多个机械单元

该例子适用于一个MultiMove系统，该系统配备两个机器人，均安装在一个导轨上。

RAPID任务对应机器人1：

```
VAR egmident egmID1;
! Activate the mechanical unit for the track motion
ActUnit TRACK1;
EGMReset egmID1;
EGMGetId egmID1;
! Set up the EGM streaming destination for ROB1, including active
     additional axis, using device "UCdevice1" and configuration
     "default"
EGMSetupUC ROB_1, egmID1, "default", "UCdevice1"\Joint;
EGMStreamStart egmID1;
! Start the position stream for ROB1 including active additional
     axis. Cycle time is 4 ms (default).
! Run your program – streaming is active
MoveJ p10, v1000, z50, Weldgun;
…
…
MoveAbsJ jpos10\NoEOffs, v1000, fine, Weldgun;
! Stop the position stream
EGMStreamStop egmID1;
! Deactivate the mechanical unit for the track motion
DeactUnit TRACK1;
```

RAPID任务对应机器人2：

```
VAR egmident egmID2;
! Activate the mechanical unit for the track motion
ActUnit TRACK2;
EGMReset egmID2;
```

下一页继续

5.4.1 使用EGM位置流

续前页

```
EGMGetId egmID2;
! Set up the EGM streaming destination for ROB2, including active
     additional axis, using device "UCdevice2" and configuration
     "default"
EGMSetupUC ROB_2, egmID2, "default", "UCdevice2"\Joint;
! Start the position stream for ROB2 including active additional
     axis. Cycle time is 4 ms (default).
EGMStreamStart egmID2;
! Run your program – streaming is active
MoveJ p10, v1000, z50, PKI_500;
…
…
MoveAbsJ jpos10\NoEOffs, v1000, fine, PKI_500;
! Stop the position stream
EGMStreamStop egmID2;
! Deactivate the mechanical unit for the track motion
DeactUnit TRACK2;
```

5.4.2 使用带一件UdpUc装置的EGM Position Guidance

描述

若要让该装置为EGM提供输入数据，那么就必须首先将其配置成一件UdpUc装置。具体请参见第33页的如何配置UdpUc设备。

此时便可让EGM用该装置来指引一台机器人了。下面是一个简单的示例：

示例

```
MODULE EGM_test
  VAR egmident egmID1;
  VAR egmstate egmSt1;

  ! limits for cartesian convergence: +-1 mm
  CONST egm_minmax egm_minmax_lin1:=[-1,1];
  ! limits for orientation convergence: +-2 degrees
  CONST egm_minmax egm_minmax_rot1:=[-2,2];

  ! Start position
  CONST jointtarget
       jpos10:=[[0,0,0,0,40,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  ! Used tool
  TASK PERS tooldata
       tFroniusCMT:=[TRUE,[[12.3313,-0.108707,416.142],
       [0.903899,-0.00320735,0.427666,0.00765917]],
       [2.6,[-111.1,24.6,386.6],[1,0,0,0],0,0,0.072]];
  ! corr-frame: wobj, sens-frame: wobj
  TASK PERS wobjdata wobj_EGM1:=[FALSE,TRUE,"",
       [[150,1320,1140],[1,0,0,0]], [[0,0,0],[1,0,0,0]]];
  ! Correction frame offset: none
  VAR pose corr_frame_offs:=[[0,0,0],[1,0,0,0]];

    PROC main()
      ! Move to start position. Fine point is demanded.
      MoveAbsJ jpos10\NoEOffs, v1000, fine, tFroniusCMT;
      testuc;
    ENDPROC

    PROC testuc()
      EGMReset egmID1;
      EGMGetId egmID1;

      egmSt1:=EGMGetState(egmID1);
      TPWrite "EGM state: "\Num:=egmSt1;

      IF egmSt1 <= EGM_STATE_CONNECTED THEN
        ! Set up the EGM data source: UdpUc server using device
             "EGMsensor:" and configuration "default"
        EGMSetupUC ROB_1, egmID1, "default", "EGMsensor:"\pose;
      ENDIF
```

下一页继续

```
        ! Correction frame is the World coordinate system and the
             sensor measurements are relative to the tool frame of
             the used tool (tFroniusCMT)
        EGMActPose egmID1\Tool:=tFroniusCMT, corr_frame_offs,
             EGM_FRAME_WORLD, tFroniusCMT.tframe, EGM_FRAME_TOOL
             \x:=egm_minmax_lin1 \y:=egm_minmax_lin1
             \z:=egm_minmax_lin1 \rx:=egm_minmax_rot1
             \ry:=egm_minmax_rot1 \rz:=egm_minmax_rot1 \LpFilter:=20;
        ! Run: the convergence condition has to be fulfilled during
             2 seconds before RAPID execution continues to the next
             instruction
        EGMRunPose egmID1, EGM_STOP_HOLD \x \y \z \CondTime:=2
             \RampInTime:=0.05;

        egmSt1:=EGMGetState(egmID1);
        IF egmSt1 = EGM_STATE_CONNECTED THEN
          TPWrite "Reset EGM instance egmID1";
          EGMReset egmID1;
        ENDIF
      ENDPROC
ENDMODULE
```

5.4.3 使用带输入项信号的EGM Position Guidance

描述

必须在本系统的I / O配置中定义要和EGM搭配使用的所有信号，即那些用EGMSetupAI、EGMSetupAO或EGMSetupGI设置的信号。之后EGM便可用这些信号来指引一台机器人。

以下RAPID程序示例将模拟输出信号作为了输入项。采用模拟输出信号的主要原因是这些信号比模拟输入信号更容易仿真。在真实应用中，编组输入信号和模拟输入信号可能更为普遍。

为简便起见，我们将下例中的相关模拟输出信号设置成EGMRun指令前的一个恒定值。通常由一件外部装置来更新相关信号值，从而得出所需的机器人位置。

下面的第二个例子说明了7轴机器人如何在EGM关节模式下使用。

例 1

```
MODULE EGM_test
VAR egmident egmID1;
VAR egmident egmID2;

CONST egm_minmax egm_minmax_lin1:=[-1,1];
CONST egm_minmax egm_minmax_rot1:=[-2,2];
CONST egm_minmax egm_minmax_joint1:=[-0.1,0.1];

CONST robtarget p20:=[[150,1320,1140],
     [0.000494947,0.662278,-0.749217,-0.00783173], [0,0,-1,0],
     [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
CONST robtarget p30:=[[114.50,1005.42,1410.38],
     [0.322151,-0.601023,0.672381,0.287914], [0,0,-1,0],
     [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
CONST jointtarget
     jpos10:=[[0,0,0,0,35,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

CONST pose posecor:=[[1200,400,900],[1,0,0,0]];
CONST pose posesens:=[[12.3313,-0.108707,416.142],
     [0.903899,-0.00320735,0.427666,0.00765917]];

! corr-frame: world, sens-frame: world
VAR pose posecor0:=[[0,0,0],[1,0,0,0]];
VAR pose posesen0:=[[0,0,0],[1,0,0,0]];

TASK PERS tooldata tFroniusCMT:=[TRUE,[[12.3313,-0.108707,416.142],
     [0.903899,-0.00320735,0.427666,0.00765917]],
     [2.6,[-111.1,24.6,386.6],[1,0,0,0],0,0,0.072]];
TASK PERS loaddata load1:=[5,[0,1,0],[1,0,0,0],0,0,0];
! corr-frame: wobj, sens-frame: wobj
TASK PERS wobjdata
     wobj_EGM1:=[FALSE,TRUE,"",[[150,1320,1140],[1,0,0,0]],
     [[0,0,0],[1,0,0,0]]];
VAR pose posecor1:=[[0,0,0],[1,0,0,0]];
VAR pose posesen1:=[[0,0,0],[1,0,0,0]];
```

下一页继续

5.4.3 使用带输入项信号的EGM Position Guidance

续前页

```
TASK PERS wobjdata
     wobj_EGM2:=[FALSE,TRUE,"",[[0,1000,1000],[1,0,0,0]],
     [[0,0,0],[1,0,0,0]]];
VAR pose posecor2:=[[150,320,0],[1,0,0,0]];
VAR pose posesen2:=[[150,320,0],[1,0,0,0]];

PROC main()
MoveAbsJ jpos10\NoEOffs, v1000, fine, tFroniusCMT;
testAO;
ENDPROC

PROC testAO()
! Get two different EGM identities. They will be used for two
     different eGM setups.
EGMGetId egmID1;
EGMGetId egmID2;

! Set up the EGM data source: Analog output signals and
     configuration "default"
! One guidance using Pose mode and one using Joint mode
EGMSetupAO ROB_1,egmID1,"default" \Pose \aoR1x:=ao_MoveX
     \aoR2y:=ao_MoveY \aoR3z:=ao_MoveZ \aoR5ry:=ao_RotY
     \aoR6rz:=ao_RotZ;
EGMSetupAO ROB_1,egmID2,"default" \Joint \aoR1x:=ao_MoveX
     \aoR2y:=ao_MoveY \aoR3z:=ao_MoveZ \aoR4rx:=ao_RotX
     \aoR5ry:=ao_RotY \aoR6rz:=ao_RotZ;

! Move to the starting point - fine point is needed.
MoveJ p30, v1000, fine, tool0;
! Set the signals
SetAO ao_MoveX, 150;
SetAO ao_MoveY, 1320;
SetAO ao_MoveZ, 900;
! Correction frame is the World coordinate system and the sensor
     measurements are also relative to the world frame
! No offset is defined (posecor0 and posesen0)
EGMActPose egmID1 \Tool:=tFroniusCMT \WObj:=wobj0 \TLoad:=load1,
     posecor0, EGM_FRAME_WORLD, posesen0, EGM_FRAME_WORLD
     \x:=egm_minmax_lin1 \y:=egm_minmax_lin1 \z:=egm_minmax_lin1
     \rx:=egm_minmax_rot1 \ry:=egm_minmax_rot1 \rz:=egm_minmax_rot1
     \LpFilter:=20 \SampleRate:=16 \MaxPosDeviation:=1000;
! Run: keep the end position without returning to the start position
EGMRunPose egmID1,
     EGM_STOP_HOLD\x\y\z\RampInTime:=0.05\PosCorrGain:=1;

! Move to the starting point - fine point is needed.
MoveJ p20, v1000, fine, tFroniusCMT;
! Set the signals
SetAO ao_MoveX, 150;
SetAO ao_MoveY, 1320;
SetAO ao_MoveZ, 1100;
```

下一页继续

```
! Run with the same frame definitions: ramp down to the start
     position after having reached the EGM end position
EGMRunPose egmID1,
     EGM_STOP_RAMP_DOWN\x\y\z\RampInTime:=0.05\PosCorrGain:=1;

! Move to the starting point - fine point is needed.
MoveJ p30, v1000, fine, tool0;
! Set the signals
SetAO ao_MoveX, 50;
SetAO ao_MoveY, -20;
SetAO ao_MoveZ, -20;
! Correction frame is the Work object wobj_EGM1 and the sensor
     measurements are also relative to the same work object. No
     offset is defined (posecor1 and posesen1)
EGMActPose egmID1 \Tool:=tFroniusCMT \WObj:=wobj_EGM1 \TLoad:=load1,
     posecor1, EGM_FRAME_WOBJ, posesen1, EGM_FRAME_WOBJ
     \x:=egm_minmax_lin1 \y:=egm_minmax_lin1 \z:=egm_minmax_lin1
     \rx:=egm_minmax_rot1 \ry:=egm_minmax_rot1 \rz:=egm_minmax_rot1
     \LpFilter:=20;
! Run: keep the end position without returning to the start position
EGMRunPose egmID1,
     EGM_STOP_HOLD\x\y\z\RampInTime:=0.05\PosCorrGain:=1;

! Move to the starting point - fine point is needed.
MoveJ p20, v1000, fine, tFroniusCMT;
! Set the signals
SetAO ao_MoveX, 0;
SetAO ao_MoveY, 0;
SetAO ao_MoveZ, 0;
! Correction frame is the Work object wobj_EGM2 and the sensor
     measurements are also relative to the same work object. This
     time an offset is defined for the correction frame (posecor2),
     and for the sensor frame (posesen2)
EGMActPose egmID1 \Tool:=tFroniusCMT \WObj:=wobj_EGM2 \TLoad:=load1,
     posecor2, EGM_FRAME_WOBJ, posesen2, EGM_FRAME_WOBJ
     \x:=egm_minmax_lin1 \y:=egm_minmax_lin1 \z:=egm_minmax_lin1
     \rx:=egm_minmax_rot1 \ry:=egm_minmax_rot1 \rz:=egm_minmax_rot1
     \LpFilter:=20;
! Run: keep the end position without returning to the start position
EGMRunPose egmID1,
     EGM_STOP_HOLD\x\y\z\RampInTime:=0.05\PosCorrGain:=1;

! Move to the starting point - fine point is needed.
MoveJ p20, v1000, fine, tFroniusCMT;
! Set the signals
SetAO ao_MoveX, 0;
SetAO ao_MoveY, 0;
SetAO ao_MoveZ, 0;
! Correction frame is of tool type and the sensor measurements are
     relative to the work object wobj_EGM2. This time an offset
     is defined for the correction frame (posecor2), and for the
     sensor frame (posesen2)
```

下一页继续

5.4.3 使用带输入项信号的EGM Position Guidance

续前页

```
EGMActPose egmID1 \Tool:=tFroniusCMT \WObj:=wobj_EGM2, posecor2,
     EGM_FRAME_TOOL, posesen2, EGM_FRAME_WOBJ \x:=egm_minmax_lin1
     \y:=egm_minmax_lin1 \z:=egm_minmax_lin1 \rx:=egm_minmax_rot1
     \ry:=egm_minmax_rot1 \rz:=egm_minmax_rot1 \LpFilter:=20;
EGMRunPose egmID1,
     EGM_STOP_HOLD\x\y\z\RampInTime:=0.05\PosCorrGain:=1;

! Move to the starting point - fine point is needed.
MoveJ p20, v1000, fine, tFroniusCMT\TLoad:=load1;
! Set the signals
SetAO ao_MoveX, 150;
SetAO ao_MoveY, 1320;
SetAO ao_MoveZ, 1100;
! Same as last, but with tool0 and wobj0
EGMActPose egmID1, posecor2, EGM_FRAME_TOOL, posesen2,
     EGM_FRAME_WOBJ \x:=egm_minmax_lin1 \y:=egm_minmax_lin1
     \z:=egm_minmax_lin1 \rx:=egm_minmax_rot1 \ry:=egm_minmax_rot1
     \rz:=egm_minmax_rot1 \LpFilter:=20;
! Run: keep the end position without returning to the start position
EGMRunPose egmID1,
     EGM_STOP_HOLD\x\y\z\RampInTime:=0.05\PosCorrGain:=1;

! Move to the starting point - fine point is needed.
MoveJ p20, v1000, fine, tFroniusCMT\TLoad:=load1;
! Set the signals
SetAO ao_MoveX, 70;
SetAO ao_MoveY, -5;
SetAO ao_MoveZ, 0;
SetAO ao_RotX, 0;
SetAO ao_RotY, 0;
SetAO ao_RotZ, 0;
! Joint guidance for joints 2-6
EGMActJoint egmID2 \J2:=egm_minmax_joint1 \J3:=egm_minmax_joint1
     \J4:=egm_minmax_joint1 \J5:=egm_minmax_joint1
     \J6:=egm_minmax_joint1 \LpFilter:=20;
! Run: keep the end position without returning to the start position
EGMRunJoint egmID2, EGM_STOP_HOLD \J2 \J3 \J4 \J5 \J6 \CondTime:=0.1
     \RampInTime:=0.05 \PosCorrGain:=1;

EGMReset egmID1;
EGMReset egmID2;
ENDPROC
ENDMODULE
```

下一页继续

例 2

```
MODULE EGM_IRB14000_test
VAR egmident egmID;
CONST egm_minmax egm_minmax_joint1:=[-0.1,0.1];
! For handling if the test is used with left or right arm.
VAR jointtarget jpos10;
CONST jointtarget jpos10_L:=[[0,-130,30,0,40,0],
     [135,9E+09,9E+09,9E+09,9E+09,9E+09]];
CONST jointtarget jpos10_R:=[[0,-130,30,0,40,0],
     [-135,9E+09,9E+09,9E+09,9E+09,9E+09]];
PROC main()
  IF GetMecUnitName(ROB_ID) = "ROB_L" THEN
    jpos10 := jpos10_L;
    testAO;
  ELSEIF GetMecUnitName(ROB_ID) = "ROB_R" THEN
    jpos10 := jpos10_R;
    testAO;
  ENDIF
ENDPROC
PROC testAO()
  ! Get an EGM idenity.
  EGMGetId egmID;
  ! Set up the EGM data source:
  ! Analog output signals and configuration "default".
  ! Only the EGM Joint mode support IRB14000.
  ! Notice the joint mapping of the analog output signals.
  EGMSetupAO ROB_ID, egmID, "default" \Joint \aoR1x:=ao_J1
       \aoR2y:=ao_J2 \aoR3z:=ao_J4 \aoR4rx:=ao_J5 \aoR5ry:=ao_J6
       \aoR6rz:=ao_J7 \AoE1:=ao_J3;
  ! Move to the starting point - fine point is needed.
  MoveAbsJ jpos10\NoEOffs, v50, fine, tool0;
  ! Set the signals (using an incrementing offset from the initial
       position).
  ! Another set of analog signals should be created, if running
       this code for both arms at the same time.
  ! Notice the joint mapping from a jointtarget to the analog output
       signals.
  SetAO ao_J1, jpos10.robax.rax_1 + 1;
  SetAO ao_J2, jpos10.robax.rax_2 + 2;
  SetAO ao_J3, jpos10.extax.eax_a + 3;
  SetAO ao_J4, jpos10.robax.rax_3 + 4;
  SetAO ao_J5, jpos10.robax.rax_4 + 5;
  SetAO ao_J6, jpos10.robax.rax_5 + 6;
  SetAO ao_J7, jpos10.robax.rax_6 + 7;
  ! Joint guidance for joints 1-7.
  EGMActJoint egmID \J1:=egm_minmax_joint1 \J2:=egm_minmax_joint1
       \J3:=egm_minmax_joint1 \J4:=egm_minmax_joint1
       \J5:=egm_minmax_joint1 \J6:=egm_minmax_joint1
       \J7:=egm_minmax_joint1 \LpFilter:=20;
  ! Run: keep the end position without returning to the start
       position.
```

下一页继续

5.4.3 使用带输入项信号的EGM Position Guidance

续前页

```
    EGMRunJoint egmID, EGM_STOP_HOLD \J1 \J2 \J3 \J4 \J5 \J6 \J7
        \CondTime:=1 \RampInTime:=0.05 \PosCorrGain:=1;
    EGMReset egmID;
ENDPROC ENDMODULE
```

5.4.4 使用协议类型不同的EGM Path Correction

描述

此例包含了不同传感器和协议类型下的示例。其中所有传感器和协议类型都采用了相同的基本RAPID程序结构和相同的外部运动数据配置。

示例

```
MODULE EGM_PATHCORR
! Used tool
PERS tooldata tEGM:=[TRUE,[[148.62,0.25,326.31],
     [0.833900724,0,0.551914471,0]], [1,[0,0,100],
     [1,0,0,0],0,0,0]];
! Sensor tool, has to be calibrated
PERS tooldata
     tLaser:=[TRUE,[[148.619609537,50.250017146,326.310337954],
     [0.390261856,-0.58965743,-0.58965629,0.390263064]],
     [1,[-0.920483747,-0.000000536,-0.390780849],
     [1,0,0,0],0,0,0]];
! Displacement used
VAR pose PP:=[[0,-3,2],[1,0,0,0]];
VAR egmident egmId1;

! Protocol: LTAPP
! Example for a look ahead sensor, e.g. Laser Tracker
PROC Part_2_EGM_OT_Pth_1()
  EGMGetId egmId1;
  ! Set up the EGM data source: LTAPP server using device "Optsim",
       configuration "pathCorr", joint type 1 and look ahead
       sensor.
  EGMSetupLTAPP ROB_1, egmId1, "pathCorr", "OptSim", 1\LATR;
  ! Activate EGM and define the sensor frame. Correction frame is
       always the path frame.
  EGMActMove egmId1, tLaser.tframe\SampleRate:=48;
  ! Move to a suitable approach position.
  MoveJ p100,v1000,z10,tEGM\WObj:=wobj0;
  MoveL p110,v1000,z100,tEGM\WObj:=wobj0;
  MoveL p120,v1000,z100,tEGM\WObj:=wobj0;
  ! Activate displacement (not necessary but possible)
  PDispSet PP;
  ! Move to the start point. Fine point is demanded.
  MoveL p130, v10, fine, tEGM\WObj:=wobj0;
  ! movements with path corrections.
  EGMMoveL egmId1, p140, v10, z5, tEGM\WObj:=wobj0;
  EGMMoveL egmId1, p150, v10, z5, tEGM\WObj:=wobj0;
  EGMMoveC egmId1, p160, p165, v10, z5, tEGM\WObj:=wobj0;
  ! Last path correction movement has to end with a fine point.
  EGMMoveL egmId1, p170, v10, fine, tEGM\WObj:=wobj0;
  ! Move to a safe position after path correction.
  MoveL p180,v1000,z10,tEGM\WObj:=wobj0;
  ! Release the EGM identity for reuse.
  EGMReset egmId1;
```

下一页继续

5.4.4 使用协议类型不同的EGM Path Correction

续前页

```
ENDPROC

! Protocol: LTAPP
! Example for an at point sensor, e.g. Weldguide
PROC Part_2_EGM_WG_Pth_1()
  EGMGetId egmId1;
  ! Set up the EGM data source: LTAPP server using device "wg1sim",
       configuration "pathCorr", joint type 1 and at point sensor.
  EGMSetupLTAPP ROB_1, egmId1, "pathCorr", "wg1sim", 1\APTR;
  ! Activate EGM and define the sensor frame, which is the tool
       frame for at point trackers.
  ! Correction frame is always the path frame.
  EGMActMove egmId1, tEGM.tframe\SampleRate:=48;
  ! Move to a suitable approach position.
  MoveJ p100,v1000,z10,tEGM\WObj:=wobj0;
  MoveL p110,v1000,z100,tEGM\WObj:=wobj0;
  MoveL p120,v1000,fine,tEGM\WObj:=wobj0;
  ! Activate displacement (not necessary but possible)
  PDispSet PP;
  ! Move to the start point. Fine point is demanded.
  MoveL p130, v10, fine, tEGM\WObj:=wobj0;
  ! movements with path corrections.
  EGMMoveL egmId1, p140, v10, z5, tEGM\WObj:=wobj0;
  EGMMoveL egmId1, p150, v10, z5, tEGM\WObj:=wobj0;
  EGMMoveC egmId1, p160, p165, v10, z5, tEGM\WObj:=wobj0;
  ! Last path correction movement has to end with a fine point.
  EGMMoveL egmId1, p170, v10, fine, tEGM\WObj:=wobj0;
  ! Move to a safe position after path correction.
  MoveL p180,v1000,z10,tEGM\WObj:=wobj0;
  ! Release the EGM identity for reuse.
  EGMReset egmId1;
ENDPROC

! Protocol: UdpUc
! Example for an at point sensor, e.g. Weldguide
PROC Part_2_EGM_UDPUC_Pth_1()
  EGMGetId egmId1;
  EGMSetupUC ROB_1, egmId1, "pathCorr", "UCdevice"\PathCorr\APTR;
  EGMActMove egmId1, tEGM.tframe\SampleRate:=48;
  ! Move to a suitable approach position.
  MoveJ p100,v1000,z10,tEGM\WObj:=wobj0;
  MoveL p110,v1000,z100,tEGM\WObj:=wobj0;
  MoveL p120,v1000,fine,tEGM\WObj:=wobj0;
  ! Activate displacement (not necessary but possible)
  PDispSet PP;
  ! Move to the start point. Fine point is demanded.
  MoveL p130, v10, fine, tEGM\WObj:=wobj0;
  ! movements with path corrections.
  EGMMoveL egmId1, p140, v10, z5, tEGM\WObj:=wobj0;
  EGMMoveL egmId1, p150, v10, z5, tEGM\WObj:=wobj0;
  EGMMoveC egmId1, p160, p165, v10, z5, tEGM\WObj:=wobj0;
```

下一页继续

```
  ! Last path correction movement has to end with a fine point.
  EGMMoveL egmId1, p170, v10, fine, tEGM\WObj:=wobj0;
  ! Move to a safe position after path correction.
  MoveL p180,v1000,z10,tEGM\WObj:=wobj0;
  ! Release the EGM identity for reuse.
  EGMReset egmId1;
ENDPROC
ENDMODULE
```

此页刻意留白

6 UdpUc代码示例

文件位置

本RobotWare版本有以下代码示例可用。

文件	描述
egm-sensor.cs	使用了protobuf-csharp-port的示例
egm-sensor.cpp	使用了Google protocol buffers C++的示例
egm.proto	*egm.proto*定义了相关机器人与相关传感器之间的数据契约。

可从相应的PC或机器人控制器处获得这些文件。

- **在RobotStudio的RobotWare安装文件夹中**：*...\RobotPackages\RobotWare_RPK_<version>\utility\Template\EGM*
- **在 OmniCore 控制器上**：*<SystemName>\PRODUCTS\RobotControl_x.x.x-xxx\utility\Template\EGM*

注意

右键单击**插件**浏览器中的已安装RobotWare版本，然后选择**打开数据包文件夹**，系统便会从RobotStudio**插件**标签处将您引导至RobotWare安装文件夹。

此页刻意留白

索引

ABB AB
Robotics & Discrete Automation
S-721 68 VÄSTERÅS, Sweden
Telephone +46 (0) 21 344 400

ABB AS
Robotics & Discrete Automation
Nordlysvegen 7, N-4340 BRYNE, Norway
Box 265, N-4349 BRYNE, Norway
Telephone: +47 22 87 2000

ABB Engineering (Shanghai) Ltd.
Robotics & Discrete Automation
No. 4528 Kangxin Highway
PuDong District
SHANGHAI 201319, China
Telephone: +86 21 6105 6666

ABB Inc.
Robotics & Discrete Automation
1250 Brown Road
Auburn Hills, MI 48326
USA
Telephone: +1 248 391 9000

abb.com/robotics

3HAC073318-010, Rev D, zh-cn