(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 111267097 B
(45)授权公告日 2021.03.02

(21)申请号 202010066644.6

(22)申请日 2020.01.20

(65)同一申请的已公布的文献号
申请公布号 CN 111267097 A

(43)申请公布日 2020.06.12

(73)专利权人 杭州电子科技大学
地址 310018 浙江省杭州市下沙高教园区2号大街

(72)发明人 胡海洋 刘翰文 陈洁 李忠金 黄彬彬

(74)专利代理机构 杭州君度专利代理事务所(特殊普通合伙) 33240
代理人 朱月芬

(51)Int.Cl.
B25J 9/16(2006.01)

(56)对比文件
CN 108447477 A,2018.08.24
CN 108345583 A,2018.07.31
CN 108073392 A,2018.05.25
US 2016259767 A1,2016.09.08
David Weintrop.Blockly Goes to Work: Block-based Programming for Industrial Robots.《2017 IEEE Blocks and Beyond Workshop》.IEEE,2017,29-36.

审查员 周思远

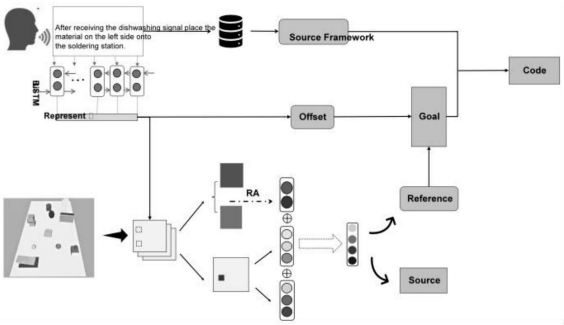
权利要求书3页 说明书5页 附图2页

(54)发明名称

基于自然语言的工业机器人辅助编程方法

(57)摘要

本发明提供一种基于自然语言的工业机器人辅助编程方法,根据语言指令和环境图像生成相应的机器人执行代码。本发明分为三部分:1)分别使用带长短时记忆(LSTM)的双向循环神经网络(Bi-RNN)和快速区域卷积神经网络(F-RCNN)提取语言指令和工厂环境的特征。2)提出一种“多注意力机制”模型和机器翻译的对齐算法将环境中的物体与指令正确匹配,从而识别指定的物体并输出放置该物体的坐标点。3)使用上述模型输出的结果配合CoBlox模块化编程方式生成操作的机器人代码。本发明采用的“多注意力机制”模型提高识别精度,解决了当前方法在工业环境中无法精确识别物体的问题。模块化编程技术方案简化工程师编程复杂度,有效提升开发效率。



CN 111267097 B

1.一种基于自然语言的工业机器人辅助编程方法,其特征在于,步骤如下:

步骤(1)、对输入数据进行预处理;所述的输入数据为语言指令和环境图像,通过使用带LSTM的Bi-RNN提取语言指令的特征,通过F-RCNN处理环境图像,获得目标候选区域;

步骤(2)、采用机器翻译的对齐算法,解析出环境中被语言指令确定的物体即目标物体;所述的对齐算法通过多注意力机制模型完成,多注意力机制模型包括词-物体注意力、指令-物体注意力和物体-物体注意力机制;

步骤(3)、训练多注意力机制模型,识别目标物体在环境中的位置和物体放置点参考特征;

步骤(4)、通过多注意力机制预测参照物位置,并结合语言指令特征,使用蒙特卡洛算法(MCMC)预测目标物体将要被放置的位置,并输出坐标;

步骤(5)、构建可编程逻辑控制器约束的数据库(PLC约束库)和CoBlox模块化编程;

步骤(6)、解析语言指令,使解析结果与PLC约束库和模块化的编程代码相匹配,结合步骤(4)输出的坐标生成最终的机器人辅助代码。

2.根据权利要求1所述的一种基于自然语言的工业机器人辅助编程方法,其特征在于,步骤(1)、对输入的语言指令和环境图像进行预处理;所述的预处理包括使用带有LSTM的Bi-RNN提取语言指令的语言特征和使用F-RCNN预处理环境图像,从而获得目标候选区域特征;具体步骤如下:

1.1、指令编码:将i个单词组成的指令$I_i=\{x_1,x_2,x_3,\ldots,x_i\}$输入RNN网络;通过带有LSTM的Bi-RNN对语言指令进行编码,递归生成隐藏状态序列I_i,然后通过学习函数$\psi_x(x_i)$将指令映射到固定维度;

$$I_i=\text{Bi\_LSTM}(\psi_x(x_i),I_{i-1}) \quad (1)$$

$$\bar{x}=\frac{1}{n}\Sigma_{i=1}^{n}I_i \quad (2)$$

$\bar{x}$表示指令均值,I_i是指令$\bar{x}$的词嵌入表示;

1.2、环境编码:使用F-RCNN对环境图像进行预处理,得到全连接层获取图像候选区域特征:

$$V_m=RCNN\left(W_B f_{b_m}+a\right) \quad (3)$$

$V_m\in R^{m\times d_v}$表示前m个检测框的d维视觉嵌入表示;f_{bm}是每个物体的空间和性状的特征表示,W_B和a分别是物体的权重和偏差值参数;

通过一个全连接层和一个一维卷积层分别将V和I映射到相同维度:

$V=\text{relu}(\text{CONV1d}(V_m))$ (4)

$I=\text{relu}(\text{LINEAR}(I_i))$ (5)

其中$V\in R^{n\times d}$是m个对象$V=\{V_1,V_2,\cdots,V_m\}$的集合;$I\in R^d$是指令的特征。

3.根据权利要求2所述的一种基于自然语言的工业机器人辅助编程方法,其特征在于,步骤(2)、识别语言指令指定的物体;采用对齐匹配算法解决语言和物体匹配问题,另外使用多注意力机制提高机器视觉精度;具体过程如下:

提出一个新的多注意力机制处理过程:由词-物体注意力即Word-Object,指令-物体注意力即Object-Instruction和物体-物体注意力即Object-Object三个注意力联合;使用多

注意力机制衡量语言指令与每个环境中物体的匹配概率，即预测环境中每个物体的可能性；通过注意力模块$\mathcal{A}(V_m,I_i)$匹配物体和语言指令，然后使用Softmax函数归一化获得条件分布P，由$P(\mathcal{S}=b_m|I)$指令确定的目标物体分布概率；

$$P(\mathcal{S}=b_m|I)\propto \exp\left(\mathcal{A}(V_m,I_i)\right) \qquad (6)$$

$\mathcal{S}$是源目标物体的离散表示；

预测源目标物体的损失函数是条件分布概率$P(\mathcal{S}=b_m|I)$和物体在环境中的真实位置G(E)的交叉熵，本发明使用Adam优化器对损失函数进行调优；

$$Loss_{\mathcal{S}}=-\sum_i G(b_m)\log P(\mathcal{S}=b_m|I) \qquad (7)$$

多步注意力机制流程如下：

2.1、Object-Object：首先将步骤(1)F-RCNN抽取的图像候选区域特征计算差值，生成O-O关系注意力机制矩阵：

$A_W^P=W_{f\times p}(V_i-V_j)$ (8)

(V_i-V_j)是m×m的矩阵，表示第i个目标物体与第j个物体图像特征表示的差异；W_f是训练的注意力矩阵，表示执行n次后的关系注意力矩阵；

2.2、Word-Object：

使用对齐算法计算语言指令中每个单词x_i每个时态的隐藏单元输出h_t，从而表示x_i与环境中每个物体m的匹配分数score：

$$a_{m,t}=align(V_m,h_t)=\frac{\exp\left(score(V_m,h_t)\right)}{\Sigma\exp\left(score(V_m,h_{t'})\right)} \qquad (9)$$

全局向量$\tilde{z}_m$是目标物体b_m的权重之和：

$$\tilde{z}_m=\sum_t a_{m,t}\,h_t \qquad (10)$$

2.3、Object-instruction：将所有目标物体与O-O关系注意力矩阵A_W^P相乘，在全局自然语言向量$\tilde{z}_m$的引导下，计算全局向量和目标物体的嵌入特征矩阵；

$$P(V_m,I)=W_{BLOCK}(V_m^T A_W)\odot\tilde{z}_m \qquad (11)。$$

4.根据权利要求3所述的一种基于自然语言的工业机器人辅助编程方法，其特征在于，步骤(3)：训练多注意力机制模型，将结果归一化得到目标物体的概率分布，确定目标物体的位置(Source)。

5.根据权利要求4所述的一种基于自然语言的工业机器人辅助编程方法，其特征在于，步骤(4)、再次使用多注意力机制模型，预测参照物位置并结合语言指令特征；通过蒙特卡洛算法(MCMC)预测目标物体将要被放置的位置，输出坐标；

将预测过程分解为两个子过程：参照物识别(Reference)和偏移(Offset)，具体过程如下：

4.1Reference(R)：预测参照物位置的方法与步骤(3)预测目标物体位置相同，所以使用步骤(3)中相同的方法计算参照物位置的概率：

$$P(R=b_m|I) \propto \exp\left(\mathcal{A}_T(V_m,I_i)\right) \quad (12)$$

b_m为环境中m个对象，$\mathcal{A}_T$为当前环境下的注意力矩阵

4.2Offset(O)：根据语言指令特征对偏移量O(真实目标位置和预测位置的差值)进行建模，假定语言指令特征服从高斯分布，采用固定协方差的多维高斯分布拟合指令，从而预测偏移量O；

$$P(O=o|I) \propto N(\mu_o,\Sigma_o) \quad (13)$$

$$\mu_o=W_1\sigma(W_2h_{fc6}+b_1)+b_2 \quad (14)$$

h_{fc6}是F-RCNN的倒数第二个全连接层，μ_o是由全连接层和指令特征生成的高斯分布的中心(物体坐标(x,y,z))；b_1，b_2是偏置参数，W_1，W_2分别是指令和物体的权重矩阵；

4.3预测目标位置：将目标位置定义为T=Offset+Reference，采用蒙特卡洛采样(MCMC)法确定物体放置点的坐标；

对reference和offset进行分布采样，用一组$o-r$的序列$\{o_0,r_0,o_1,r_1,\ldots,o_n,r_n\}$表示采样样本集合，$t_n$是由$o-r$预测的位置；

$$o\sim P(O) \quad (15)$$

$$r\sim P(R) \quad (16)$$

$$L_{MCMC}=\mathbb{E}||t_{GT}-t_n|| \quad (17)$$

将$t_{GT}-t_n$真实位置与预测位置的距离作为负奖励，采用Reinforce Learning思想，通过蒙特卡罗方法拟合N个随机变量$o-r$的采样序列样本；具体方法如下：

$$L_{MCMC}=\frac{1}{N}\sum\left[\log P(R=r_n)-\frac{1}{2}(o_n-\mu_o)^{\top}\Sigma_o^{-1}(o_n-\mu_o)\cdot||t_{GT}-t_n||\right] \quad (18)。$$

6.根据权利要求5所述的一种基于自然语言的工业机器人辅助编程方法，其特征在于，步骤(5)、构建可编程控制器约束的数据库(PLC约束库)和CoBlox模块化编程；

5.1PLC约束库：实际生产环境中，机器人任务会受到PLC信号的约束；根据实际情况选择合适的PLC约束作为PLC约束库；

5.2CoBlox模块化编程：采用CoBlox模块化编程的方式封装带有默认参数的函数体。

7.根据权利要求6所述的一种基于自然语言的工业机器人辅助编程方法，其特征在于，步骤(6)、采用StanfordNLP工具解析语言指令，在PLC约束库中匹配分词与PLC信号，随后采用BM25算法组合匹配CoBlox模块化编程，自动生成机器人程序框架；最后结合步骤(4)预测的目标位置坐标即放置点坐标，填充到机器人程序中，生成完整的机器人代码。

基于自然语言的工业机器人辅助编程方法

技术领域

[0001] 本申请属于机器人编程技术领域,特别是涉及基于自然语言和机器视觉的机器人编程技术。

背景技术

[0002] 随着近几十年机器人技术的飞速发展,智能制造的理念深入人心。机械臂技术已经在工业生产环境中大量运用,协作式机械人融合人类和机械设备的优势,在生产线上和工人紧密合作,可以显著提升生产效率。

[0003] 目前所有的机械任务都要通过工程师精心设计和编码,才能辅助和代替工人执行单一的机械性任务。工程师通常采用在线或离线编程的方式编写机器人代码,然而这些编程方式过于耗时,时效性远不能满足产品需求的变化。例如,编写大型车体电弧焊机器人程序需要耗费八个月的时间,每个焊接节点的改动需要耗费半个月调试程序,这种高额的编程开销迫使中小企业无法在智能制造中受益。

[0004] 近年来不断有学者在机器人编程领域进行探索。手动编程是市面上应用范围最广,使用频率最高的机器人编程工具。一般使用官方编程界面,其编程语言带有早期编程语言特性,如ABB RAPID,KUKA KRL等。但是在现有生产环境下,工程师需要花费高昂的时间成本在该平台中为每一次制造任务编写代码,甚至其中包含大量重复或类似的代码。并且此种编码方式严格遵循固定的语言规范,不利于新手的快速学习和使用。

[0005] CoBlox模块化编程方式(David Weintrop等人在2017年《Blockly goes to work: Block-based programming for industrial robots》中提出,Published in:2017 IEEE Blocks and Beyond Workshop)

[0006] 另外,随着自然语言和人工智能在语言学方面的发展,学者在自动编程领域也获得了喜人的进展,比如机器人通过神经网络解析人类的语言或者动作,从而正确理解人类指令,执行任务。但是此类方法仅输出机器人的行为和状态,并不提供工业工程师所需要的源代码。这种编程方式在工业生产中需要调整方案时,无法进行代码级别的离线修改。因为不生成代码文本,也不利于相似代码在其他工程中的重利用。

[0007] 在这种情形下,现有的编程技术因其固有的缺陷,无法契合工业智能制造的需要。

发明内容

[0008] 为克服上述现有技术的不足,需要提供一种支持抽象输入的快速编程方法,以满足当下工业智能制造的要求。本发明提供一种基于自然语言的工业机器人辅助编程方法,根据语言指令和环境图像生成相应的机器人执行代码。其主要分为三部分:1)分别使用带长短时记忆(LSTM)的双向循环神经网络(Bi-RNN)和快速区域卷积神经网络(F-RCNN)提取语言指令和工厂环境的特征。2)本发明提出一种“多注意力机制”模型和机器翻译的对齐算法将环境中的物体与指令正确匹配,从而识别指定的物体并输出放置该物体的坐标点。3)使用上述模型输出的结果配合CoBlox模块化编程方式生成操作的机器人代码。

[0009] 一种基于自然语言的工业机器人辅助编程方法,步骤如下:

[0010] 步骤(1)、对输入数据进行预处理。所述的输入数据为语言指令和环境图像,通过使用带LSTM的Bi-RNN提取语言指令的特征,通过F-RCNN处理环境图像,获得目标候选区域。

[0011] 步骤(2)、采用机器翻译的对齐算法,解析出环境中被语言指令确定的物体即目标物体。所述的对齐算法通过多注意力机制模型完成,多注意力机制模型包括词-物体注意力、指令-物体注意力和物体-物体注意力机制。

[0012] 步骤(3)、训练多注意力机制模型,识别目标物体在环境中的位置和物体放置点参考特征。

[0013] 步骤(4)、通过多注意力机制预测参照物位置,并结合语言指令特征,使用蒙特卡洛算法(MCMC)预测目标物体将要被放置的位置,并输出坐标。

[0014] 步骤(5)、构建可编程逻辑控制器约束的数据库(PLC约束库)和CoBlox模块化编程。

[0015] 步骤(6)、解析语言指令,使解析结果与PLC约束库和模块化的编程代码相匹配,结合步骤(4)输出的坐标生成最终的机器人辅助代码。

[0016] 本发明有益效果如下:

[0017] 与现有技术相比,本发明解决了现有机器视觉识别率低和工程师重复性编程的问题。本发明所提供的基于自然语言的工业机器人编程方法主要有几点创新:1)使用自然语言来辅助产生工业机器人代码;2)使用多步注意力机制模型提高机器视觉精度;3)采用CoBlox模块化编程方式获得机器人代码,不需要重复编写相似的代码,简化开发人员开发负担。

[0018] 本发明不需要严格遵循语言规范,而是人类的自然语言即可生成源码,提高了抽象性的同时降低了机器人编程门槛,为新手提供了好的编程方法。本发明采用的“多注意力机制”模型提高识别精度,解决了当前方法在工业环境中无法精确识别物体的问题。模块化编程技术方案简化工程师编程复杂度,有效提升开发效率。

附图说明

[0019] 图1为本发明总体模型结构示意图;

[0020] 图2为ABB码垛机器人的PLC信号表格;

[0021] 图3为本发明任务3执行流程示意图;

[0022] 图4为本发明任务流程图。

具体实施方式

[0023] 本发明包括三个子任务,下面结合附图和实施例对本发明作进一步说明。

[0024] 图1为本发明总体模型结构示意图。

[0025] 本发明方法分为三个衔接的子任务,具体步骤如下:

[0026] 任务1.识别目标物体

[0027] 步骤(1)、对输入的语言指令和环境图像进行预处理。所述的预处理包括使用带有LSTM的Bi-RNN提取语言指令的语言特征和使用F-RCNN预处理环境图像,从而获得目标候选区域特征。具体步骤如下:

[0028] 1.1指令编码:将i个单词组成的指令$I_i=\{x_1,x_2,x_3,\cdots,x_i\}$输入RNN网络。通过带有LSTM的Bi-RNN对语言指令进行编码,递归生成隐藏状态序列I_i,然后通过学习函数$\psi_x x_i$将指令映射到固定维度。

[0029] $I_i=Bi\_LSTM(\psi_x(x_i),I_{i-1})$ (1)

[0030] $$\bar{x}=\frac{1}{n}\Sigma_{i=1}^{n}I_i \qquad (2)$$

[0031] $\bar{x}$表示指令均值,I_i是指令$\bar{x}$的词嵌入表示。

[0032] 1.2环境编码:使用F-RCNN对环境图像进行预处理,得到全连接层获取图像候选区域特征:

[0033] $$V_m=RCNN\left(W_B f_{b_m}+a\right) \qquad (3)$$

[0034] $V_m\in R^{m\times d_v}$表示前m个检测框的d维视觉嵌入表示。f_{b_m}是每个物体的空间和性状的特征表示,W_B和a分别是物体的权重和偏差值参数。

[0035] 通过一个全连接层和一个一维卷积层分别将V和I映射到相同维度:

[0036] $V=relu(CONV1d(V_m))$ (4)

[0037] $I=relu(LINEAR(I_i))$ (5)

[0038] 其中$V\in R^{m\times d}$是m个对象$V=\{V_1,V_2,\cdots,V_m\}$的集合。$I\in R^d$是指令的特征。

[0039] 步骤(2)、识别语言指令指定的物体。如何正确将指令和环境物体匹配,以及如何从相似的环境物体中选取确定的物体,这提高识别精度的关键。本发明采用对齐匹配算法解决语言和物体匹配问题,另外使用多注意力机制提高机器视觉精度。具体过程如下:

[0040] 提出一个新的多注意力机制处理过程:由Word-Object,Object-Instruction和Object-Object三个注意力联合。使用多注意力机制衡量语言指令与每个环境中物体的匹配概率,即预测环境中每个物体的可能性。通过注意力模块$\mathcal{A}(V_m,I_i)$匹配物体和语言指令,然后使用Softmax函数归一化获得条件分布P,由$P(S=b_m|I)$指令确定的目标物体分布概率。

[0041] $$P(S=b_m|I)\propto \exp\left(\mathcal{A}(V_m,I_i)\right) \qquad (6)$$

[0042] S是源目标物体的离散表示。

[0043] 预测源目标物体的损失函数是条件分布概率$P(S=b_m|I)$和物体在环境中的真实位置G(E)的交叉熵,本发明使用Adam优化器对损失函数进行调优。

[0044] $$Loss_S=-\sum_i G(b_m)\log P(S=b_m|I) \qquad (7)$$

[0045] 多步注意力机制流程如下:

[0046] 2.1Object-Object:首先将步骤(1)F-RCNN抽取的图像候选区域特征计算差值,生成O-O关系注意力机制矩阵:

[0047] $A_W^p=W_{f\times p}(V_i-V_j)$ (8)

[0048] V_i,V_j已在上文中定义,(V_i-V_j)是m×m的矩阵,表示第i个目标物体与第j个物体图像特征表示的差异。W_f是训练的注意力矩阵,表示执行n次后的关系注意力矩阵。

[0049] 2.2Word-Object:

[0050] 使用对齐算法计算语言指令中每个单词x_i每个时态的隐藏单元输出h_t,从而表示x_i与环境中每个物体m的匹配分数score:

[0051] $$a_{m,t} = align(V_m, h_t) = \frac{\exp(score(V_m, h_t))}{\Sigma \exp(score(V_m, h_{t'}))} \quad (9)$$

[0052] 全局向量$\tilde{z}_m$是目标物体b_m的权重之和:

[0053] $$\tilde{z}_m = \sum_t a_{m,t} h_t \quad (10)$$

[0054] 2.3Object-instruction:将所有目标物体与O-O关系注意力矩阵A_w^p相乘,在全局自然语言向量$\tilde{z}_m$的引导下,计算全局向量和目标物体的嵌入特征矩阵。

[0055] $$P(V_m, I) = W_{BLOCK}(V_m^T A_W) \odot \tilde{z}_m \quad (11)$$

[0056] 步骤(3):训练多注意力机制模型,将结果归一化得到目标物体的概率分布,确定目标物体的位置(Source)。

[0057] 任务2:预测目标放置的位置

[0058] 步骤(4)、再次使用多注意力机制模型,预测参照物位置并结合语言指令特征。通过蒙特卡洛算法(MCMC)预测目标物体将要被放置的位置,输出坐标。

[0059] 将预测过程分解为两个子过程:参照物识别(Reference)和偏移(Offset),具体过程如下:

[0060] 4.1 Reference(R):预测参照物位置的方法与步骤(3)预测目标物体位置相同,所以使用步骤(3)中相同的方法计算参照物位置的概率:

[0061] $$P(R = b_m | I) \propto \exp(\mathcal{A}_T(V_m, I_i)) \quad (12)$$

[0062] b_m为环境中m个对象,$\mathcal{A}_T$为当前环境下的注意力矩阵

[0063] 4.2Offset(O):根据语言指令特征对偏移量O(真实目标位置和预测位置的差值)进行建模,假定语言指令特征服从高斯分布,采用固定协方差的多维高斯分布拟合指令,从而预测偏移量O。

[0064] $P(O=o|I) \propto N(\mu_o, \Sigma_o)$ (13)

[0065] $\mu_o = W_1\sigma(W_2 h_{fc6} + b_1) + b_2$ (14)

[0066] h_{fch}是F-RCNN的倒数第二个全连接层,μ_o是由全连接层和指令特征生成的高斯分布的中心(物体坐标(x,y,z))。b_1,b_2是偏置参数,W_1,W_2分别是指令和物体的权重矩阵。

[0067] 4.3预测目标位置:将目标位置定义为T=Offset+Reference,采用蒙特卡洛采样(MCMC)法确定物体放置点的坐标。

[0068] 对reference和offset进行分布采样,用一组$o-r$的序列$\{o_0, r_0, o_1, r_1, \ldots, o_n, r_n\}$表示采样样本集合,$t_n$是由$o-r$预测的位置。

[0069] $$o \sim P(O) \quad (15)$$

[0070] $$r \sim P(R) \quad (16)$$

[0071] $$L_{MCMC} = \mathbb{E}||t_{GT} - t_n|| \quad (17)$$

[0072] 将t_{Gi}-t_n真实位置与预测位置的距离作为负奖励,采用Reinforce Learning思想,通过蒙特卡罗方法拟合N个随机变量$o - r$的采样序列样本。具体方法如下:

[0073] $$L_{MCMC} = \frac{1}{N}\sum\left[\log P(R = r_n) - \frac{1}{2}(o_n - \mu_o)^{\top}\Sigma_o^{-1}(o_n - \mu_o) \cdot ||t_{GT} - t_n||\right] \quad (18)$$

[0074] 通过蒙特卡洛方法拟合确认指令所指定物体在环境中的放置点,并输出放置点坐标(x,y,z)。

[0075] 任务3.生成机器人代码

[0076] 步骤(5)、构建可编程控制器约束的数据库(PLC约束库)和CoBlox模块化编程。

[0077] 5.1 PLC约束库:实际生产环境中,机器人任务会受到PLC信号的约束。根据实际情况选择合适的PLC约束作为PLC约束库。PLC是一种可以与机器人交互并约束机器人行为的可编程控制器,由PLC工程师编写和定义。目前各生产机械臂厂家没有统一的PLC信号规定,存在机器同一个信号在编程平台中存在多种表示的问题。针对目前各家厂商规定不统一的问题,本实施例采用ABB码垛机器人的PLC信号表格(部分见附图2)。

[0078] 5.2 CoBlox模块化编程:采用CoBlox模块化编程的方式封装带有默认参数的函数体。例如Move<speed>to<somewhere>表示命令机械臂移动到固定位置,神经网络预测机械臂抓取的目标和放置位置(<somewhere>)以代替原始手动输入的位置参数,<speed>将分配默认参数并允许程序员手动更改;SET<what>则由分词工具截取PLC字段映射到相应信号。

[0079] 步骤(6)、采用StanfordNLP工具解析语言指令,在PLC约束库中匹配分词与PLC信号,随后采用BM25算法组合匹配CoBlox模块化编程,自动生成机器人程序框架。最后结合步骤(4)预测的目标位置坐标即放置点坐标,填充到机器人程序中,生成完整的机器人代码。(任务流程见附图3)

[0080] 本发明由工程师控制,允许语音和文本输入机器人任务指令,并且直接在终端输出任务执行代码,中间过程不可见。这种代码生成方法不仅支持灵活的输入方式,还采用模块化编程方法在简化编程任务的同时也能约束工程师编程行为,提高编程规范性。(任务流程见附图4)。

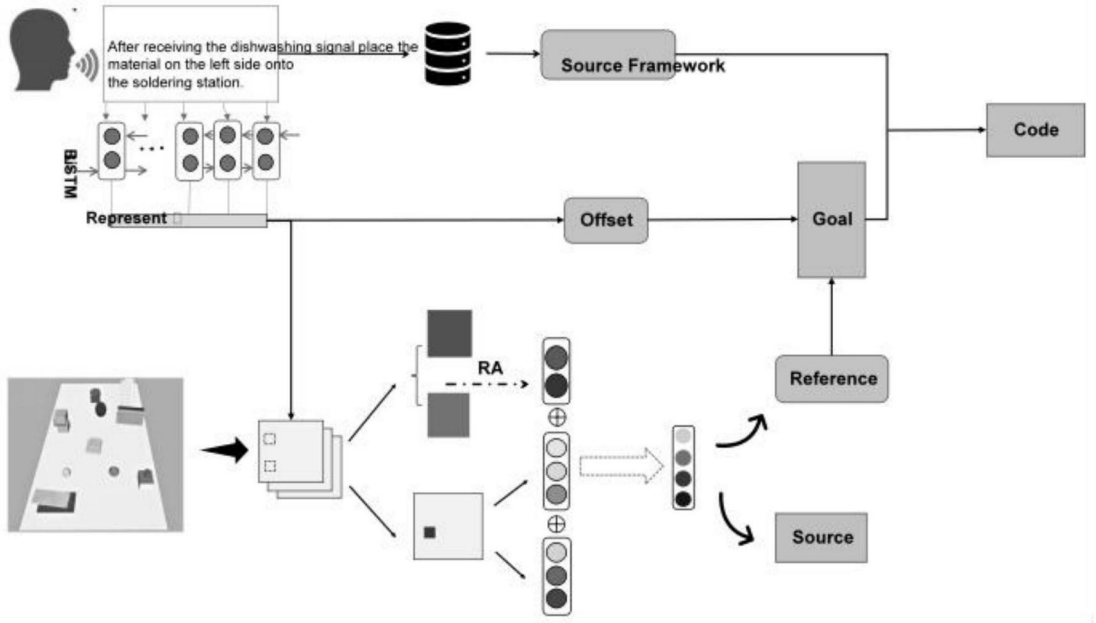


图1

Mapping	Name	Type	Note
…	…	…	…
200	IN_CAR_SELECTED	DO:200	Select a car
201	XiPan	DO:201	Executive sucker
202	XiPan_OK	DO:202	Executive sucker OK
203	FANG	DO:203	Put down the sucker
204	FANG_OK	DO:204	Put down the sucker OK
205	QU_OK	DO:205	Grab
206	IN_Task_Executing	DO:206	Task execution
207	IN_Unqualified_Flag	DO:207	Disqualification mark
208	DI_Offset_Width	DO:208	Attitude shift
…	…	…	…
230	OUT_Request_Data	DO:230	Output request data

图2

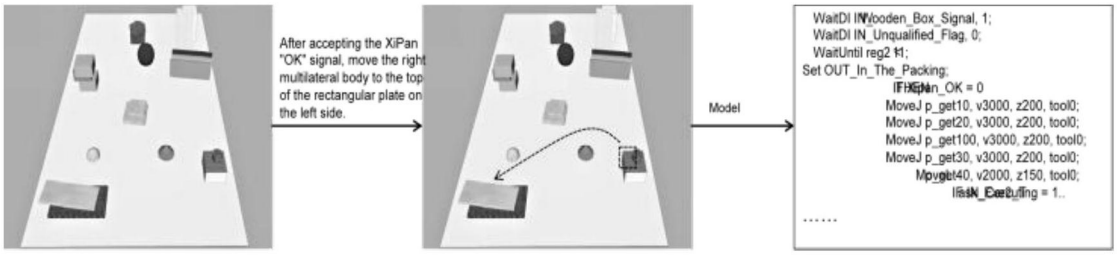

After accepting the XiPan "OK" signal, move the right multilateral body to the top of the rectangular plate on the left side.
Model
WaitDI IN_Wooden_Box_Signal, 1;
WaitDI IN_Unqualified_Flag, 0;
WaitUntil reg2 =1;
Set OUT_In_The_Packing;
MoveJ p_get10, v3000, z200, tool0;
MoveJ p_get20, v3000, z200, tool0;
MoveJ p_get100, v3000, z200, tool0;
MoveJ p_get30, v3000, z200, tool0;
.....

图3

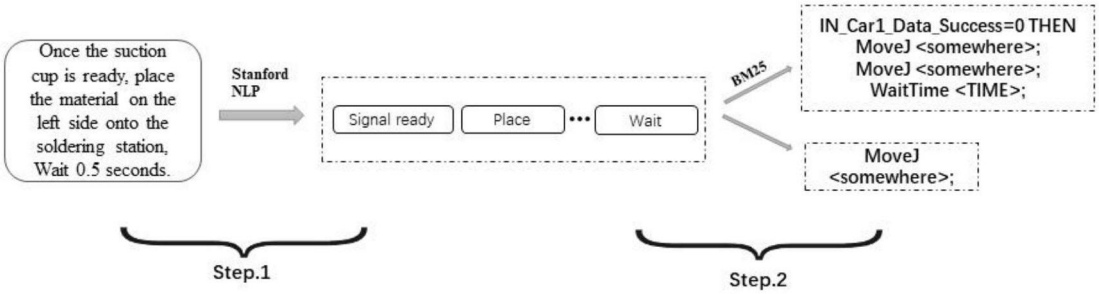

Once the suction cup is ready, place the material on the left side onto the soldering station, Wait 0.5 seconds.
Stanford NLP
Signal ready
Place
Wait
BM25
IN_Car1_Data_Success=0 THEN
MoveJ <somewhere>;
MoveJ <somewhere>;
WaitTime <TIME>;
MoveJ <somewhere>;
Step.1
Step.2

图4