SIMULAÇÃO E PROGRAMAÇÃO *OFFLINE* DE ROBÔS INDUSTRIAIS AFECTOS A TAREFAS DE LIXAGEM

Pedro Daniel Tavares Madaleno

Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

2011

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Disciplina de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Electrotécnica e de Computadores

Candidato: Pedro Daniel Tavares Madaleno, Nº 1030427, 1030427@isep.ipp.pt

Orientação científica: Manuel Fernando dos Santos Silva, mss@isep.ipp.pt

Empresa: Grohe Portugal – Componentes Sanitários, LDA.

Supervisão: Sérgio Costa, SCosta@grohe.pt

Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

24 de Outubro de 2011

Agradecimentos

Em primeiro lugar gostaria de agradecer a realização deste trabalho ao Eng.º Manuel Silva pela oportunidade e confiança depositada na minha pessoa para a realização do estágio facultado pela Grohe Portugal, bem como agradecer à Grohe por ter disponibilizado o referido estágio.

Gostaria ainda de mostrar o meu apreço por todo o apoio dedicado à minha pessoa pelo Eng.º Sérgio Costa, Eng.º Milton Rino e Eng.º Celso Maia ao longo do meu estágio na Grohe, bem como ao chefe dos afinadores Sérgio Amador por todos os ensinamentos e explicações cuidadas para a realização de um trabalho correcto e eficiente.

Desejo ainda dar um especial agradecimento ao Departamento de Engenharia Mecânica do ISEP por ter disponibilizado uma versão do SolidWorks para a realização da modelação 3D, bem como ao Eng.º Rui Fazenda, do referido departamento, pelo auxílio prestado na conversão dos ficheiros elaborados para a representação das células dos robôs.

Por último, um agradecimento a toda a minha família e amigos por todo o apoio e força depositada em mim ao longo da realização deste relatório, bem como ao longo do meu estágio.

Em suma, um enorme obrigado a todos os que directa ou indirectamente contribuíram para o meu sucesso escolar e para a realização deste trabalho.

Resumo

Este trabalho teve o intuito de testar a viabilidade da programação *offline* para tarefas de lixamento na empresa Grohe Portugal.

Para tal era necessário perceber o que é a programação *offline* e para isso foi efectuada uma pesquisa referente a essa temática, onde ficou evidente que a programação *offline* é em tudo semelhante à programação *online*, tendo apenas como principal diferença o facto de não usar o robô propriamente dito durante o desenvolvimento do programa.

Devido à ausência do robô, a programação *offline* exige que se conheça detalhadamente a célula de trabalho, bem como todas as entradas e saídas associadas à célula, sendo que o conhecimento das entradas e saídas pode ser contornada carregando um *backup* do robô ou carregando os módulos de sistema. No entanto os fabricantes habitualmente não fornecem informação detalhada sobre as células de trabalho, o que dificulta o processo de implementação da unidade no modelo 3D para a programação *offline*.

Após este estudo inicial, foi efectuado um estudo das características inerentes a cada uma das células existentes, com o objectivo de se obter uma melhor percepção de toda a envolvente relacionada com as tarefas de lixamento. Ao longo desse estudo efectuaram-se vários testes para validar os diversos programas desenvolvidos, bem como para testar a modelação 3D efectuada.

O projecto propriamente dito consistiu no desenvolvimento de programas *offline* de forma a minimizar o impacto (em especial o tempo de paragem) da programação de novos produtos. Todo o trabalho de programação era até então feito utilizando o robô, o que implicava tempos de paragem que podiam ser superiores a três dias. Com o desenvolvimento dos programas em modo *offline* conseguiu-se reduzir esse tempo de paragem dos robôs para pouco mais de um turno (8h), existindo apenas a necessidade de efectuar algumas afinações e correcções nos movimentos de entrada, saída e movimentações entre rotinas e unidades, uma vez que estes movimentos são essenciais ao bom acabamento da peça e convém que seja suaves.

Para a realização e conclusão deste projecto foram superadas diversas etapas, sendo que as mais relevantes foram:

- A correcta modelação 3D da célula, tendo em conta todo o cenário envolvente, para evitar colisões do robô com a célula;
- A adaptação da programação *offline* para uma linguagem mais usual aos afinadores, ou seja, efectuar a programação com *targets inline* e criar diferentes rotinas para cada uma das partes da peça, facilitando assim a afinação;
- A habituação à programação recorrendo apenas ao uso de módulos para transferir os programas para a célula, bem como a utilização de entradas, saídas e algumas rotinas e funcionalidades já existentes.

Palavras-Chave

Robôs, lixamento/polimento, programação *offline*, ABB RobotStudio.

Abstract

This work was developed to test the possibility of using offline programing in grinding tasks at Grohe Portugal.

To accomplish this task, it was necessary to understand what offline programing is and with this purpose a research about this theme was developed. During this research it became evident that offline and online programing are quite similar, being the main difference between both the lack of the physical robot during the offline programing.

Due to the absence of the robot during offline programing, the programmer must know every detail from the work cell and every input/output (IO) from the robot. The information on the IOs can be handled with no problem if the programmer is able to create a backup, or get the system files from the cell and upload those backup/files to the offline program. However the detailed knowledge from the cell is normally hard to have because manufacturers normally don't provide that information to protect their business.

After this initial study it was conducted a study to get to know all the characteristics of each existent cell, and to get all the possible knowledge about grinding. During this study there were conducted some tests to validate the 3D modeling and the developed programs.

The project consisted in developing offline programs to minimize the robots stop time during the programing of new products. All the programing done until that moment was made using the real cell, a process that could take more than three days. With the offline programing the robots stop time was reduced to a little more than a work shift (8h), being needed only some minor adjusts on some in/out movements and on the movements between routines and units.

To finish this project many stages were passed, being the most important ones:

- The correct 3D modeling of the robot world to avoid robot collisions with the cell;
- The adaptation of the offline programming to a programing language more common to the setters, because offline programing does not use inline targets and the use of routines for each part of the tap grinding process;

- The habituation to the offline programing using modules to transfer the program to the cell, and the use of some IOs, routines and functionalities already existent on the system.

Keywords

Robots, grinding/polishing, offline programing, ABB RobotStudio.

Índice

AGRADECIMENTOS ..... **I**

RESUMO ..... **III**

ABSTRACT ..... **V**

ÍNDICE ..... **VII**

ÍNDICE DE FIGURAS ..... **IX**

ÍNDICE DE TABELAS ..... **XI**

ACRÓNIMOS ..... **XIII**

1. INTRODUÇÃO **1**

1.1. CONTEXTUALIZAÇÃO 1
1.2. OBJECTIVOS 1
1.3. CALENDARIZAÇÃO 2
1.4. ORGANIZAÇÃO DO RELATÓRIO 3

2. ESTADO DA ARTE **5**

2.1. HISTÓRIA DOS ROBÔS INDUSTRIAIS 5
2.2. TIPOS DE ROBÔS 7
2.3. IMPACTO DOS ROBÔS INDUSTRIAIS NA SOCIEDADE 11
2.4. SIMULAÇÃO E PROGRAMAÇÃO *OFFLINE* DE ROBÔS INDUSTRIAIS 12
2.5. APLICAÇÕES PARA DESENVOLVIMENTO *OFFLINE* DE PROGRAMAS PARA TAREFAS DE LIXAMENTO COM ROBÔS 16

3. DEFINIÇÃO DO PROBLEMA **19**

3.1. GROHE PORTUGAL 19
3.2. DESCRIÇÃO DO PROCESSO DE FABRICO 20
3.3. PROBLEMAS NO LIXAMENTO DAS TORNEIRAS 28

4. MODELAÇÃO DAS CÉLULAS DE LIXAGEM **31**

4.1. AVALIAÇÃO DA CÉLULA DE TRABALHO 31
4.2. MODELAÇÃO DA CÉLULA 32
4.3. IMPORTAÇÃO E CONFIGURAÇÃO DOS MECANISMOS NO ROBOTSTUDIO 36
4.4. REALIZAÇÃO DA CALIBRAÇÃO DA CÉLULA SIMULADA 41
4.5. TESTES REALIZADOS COM PROGRAMAS DESENVOLVIDOS *OFFLINE* 42

5. DESENVOLVIMENTO DOS PROGRAMAS DOS ROBÔS **51**

5.1. CONFIGURAÇÃO DE ENTRADAS E SAÍDAS 51
5.2. DEFINIÇÃO DOS EVENTOS PARA EFEITOS DE SIMULAÇÃO DO PROGRAMA 53

5.3. DESENVOLVIMENTO DO PROGRAMA 56

6. CONCLUSÕES 65

REFERÊNCIAS DOCUMENTAIS 69

HISTÓRICO 71

Índice de Figuras

Figura 1 Elos e juntas de um robô com seis graus de liberdade [6] 7
Figura 2 Configuração cartesiana 8
Figura 3 Configuração articulada 8
Figura 4 Configuração cilíndrica 9
Figura 5 Configuração esférica 9
Figura 6 Configuração SCARA 10
Figura 7 Configuração paralela 10
Figura 8 Fornalha onde são fundidos os materiais 20
Figura 9 Fluxograma da fase de fundição 20
Figura 10 Fluxograma da fase de maquinagem 22
Figura 11 Triflex para maquinação das peças 22
Figura 12 Fluxograma da fase de lixamento/polimento 23
Figura 13 Processo de lixamento manual 24
Figura 14 Processo de polimento automático 25
Figura 15 Estrutura da fase de galvânica 26
Figura 16 Processo de lavagem, niquelagem, cromagem 27
Figura 17 Estrutura da fase de montagem 27
Figura 18 Célula de montagem manual 28
Figura 19 Unidades de lixagem presentes na célula do robô 32
Figura 20 Desenho do braço para a roda de 450 mm 33
Figura 21 Desenho da roda de 450 mm 33
Figura 22 *Assembly* do conjunto braço+roda de 450 mm 34
Figura 23 *Assembly* do conjunto braço+roda de 150 mm 34
Figura 24 Parte superior das unidades de lixagem 35
Figura 25 Unidades de lixagem da célula do robô 36
Figura 26 Criação da parte inferior das unidades de lixagem 37
Figura 27 Importação da parte superior da unidade de lixagem para o ABB RobotStudio 37
Figura 28 Posicionamento da parte superior da unidade de lixagem 38
Figura 29 Melhoramento do aspecto gráfico de uma geometria/mecanismo 38
Figura 30 Criação e configuração do mecanismo correspondente à unidade de lixagem 39
Figura 31 Continuação da configuração do mecanismo correspondente à unidade de lixagem 40
Figura 32 Definição das posições predefinidas do mecanismo correspondente à unidade de lixagem 40
Figura 33 Conclusão da criação do mecanismo e gravação como biblioteca 41

Figura 34 Ferramenta para definição dos *workobjects* das unidades 42
Figura 35 Ferramenta (*tool*) para definição dos *workobjects* 42
Figura 36 Parte superior da torneira após a realização do primeiro teste de programação *offline* 43
Figura 37 Parte de trás da torneira após a realização do primeiro teste de programação *offline* .. 44
Figura 38 Parte de baixo/frente da torneira após a realização do primeiro teste de programação *offline* 44
Figura 39 Parte de baixo da torneira após a realização do primeiro teste de programação *offline* 45
Figura 40 Lateral esquerda da torneira após a realização do primeiro teste de programação *offline* 45
Figura 41 Lateral direita da torneira após a realização do primeiro teste de programação *offline* 46
Figura 42 Unidades 1 e 3 com a representação da lixa, motor e esticador 47
Figura 43 Lateral direita da torneira utilizada no segundo teste de programação *offline* 48
Figura 44 Lateral esquerda da torneira utilizada no segundo teste de programação *offline* 48
Figura 45 Parte de baixo da torneira utilizada no segundo teste de programação *offline* 49
Figura 46 Parte de trás da torneira utilizada no segundo teste de programação *offline* 49
Figura 47 Parte de cima da torneira utilizada no segundo teste de programação *offline* 50
Figura 48 Modelo da célula de trabalho 52
Figura 49 Módulos de configuração dos robôs de lixagem 53
Figura 50 `Event Manager` 54
Figura 51 Janela do "`Event Manager`" e configuração do tipo de evento 54
Figura 52 Configuração do evento para avanço de unidades 55
Figura 53 Configuração do evento para recuo de unidades 55
Figura 54 Aspecto final da janela "`Event Manager`" com todos os eventos criados 56
Figura 55 Criação dos *targets* 58
Figura 56 Opções disponíveis para a criação e configuração dos *targets* 58
Figura 57 Menu para criação dos movimentos (definição dos *paths*) 59
Figura 58 Menu que permite efectuar a configuração dos *paths* 60
Figura 59 Configuração detalhada das instruções de movimento dos programas 60
Figura 60 Funcionalidade "`Auto Configuration`" 61
Figura 61 Configurações disponíveis para o robô executar o *path* 61
Figura 62 *Path* do robô após configurações 62
Figura 63 Exemplo de código RAPID de um programa para o robô ABB IRB4400 63
Figura 64 Custo/hora de um robô 67

Índice de Tabelas

Tabela 1 Calendarização das tarefas do trabalho desenvolvido 2

Acrónimos

3D	–	Três Dimensões
ABB	–	Asea Brown Boveri
ACIS	–	Allen, Charles, Ian's System
BP	–	Baixa Pressão
CAD		Computer-Aided Design
CAM		Computer-Aided Manufacturing
CNC	–	Computer Numerical Control
ISEP	–	Instituto Superior de Engenharia do Porto
JIRA	–	Japan Industrial Robot Association
PLC	–	Programmable Logic Controller
RUR	–	Rossum's Universal Robots
SCARA	–	Selectively Compliant Assembly Robot Arm ou Selectively Compliant Articulated Robot Arm

1. INTRODUÇÃO

Este documento pretende descrever o trabalho desenvolvido no âmbito da disciplina de Tese/Dissertação, realizado na empresa Grohe ao longo do presente ano curricular.

1.1. CONTEXTUALIZAÇÃO

Este trabalho surgiu do desejo por parte da empresa Grohe Portugal de optimizar os tempos de programação dos robôs para novos produtos. Devido à falta de conhecimento da aplicação fornecida pela Asea Brown Boveri (ABB) para esse efeito (RobotStudio), a empresa propôs ao Instituto Superior de Engenharia (ISEP) a realização de um estágio curricular com o intuito de testar a viabilidade do *software* tendo em conta o tipo de trabalhos realizados na empresa.

1.2. OBJECTIVOS

Este trabalho foi pensado e realizado com o intuito de minimizar o impacto na produção por motivos de programação, aquando da introdução de um novo produto na linha de produção sendo que, actualmente, devido à constante procura de produtos inovadores e apelativos, essa mudança é frequente.

No âmbito do desenvolvimento deste trabalho foi necessária a realização de uma pesquisa sobre o tema lixagem/polimento com utilização de robôs na execução dessas tarefas. Ao longo dessa pesquisa tornou-se evidente que está associada uma enorme complexidade à programação dos robôs afectos a estas tarefas, quer pela necessidade de realização de

movimentos complexos, quer pela necessidade de fluidez na realização desses movimentos.

No desenvolvimento de um programa para efectuar o lixamento de uma peça, o programador tem de ter em conta inúmeros factores aliados à execução da tarefa; esses factores variam, por exemplo, desde a pressão a efectuar no contacto da peça com a lixa para o correcto lixamento da superfície, à minimização de passagens no mesmo sentido aquando do lixamento evitando assim o excessivo desgaste, à complexidade e fluidez do movimento tendo em conta a geometria da peça, entre outros.

Todo o processo de programação e testes realizados ao longo deste trabalho foram efectuados recorrendo ao uso de um robô da marca ABB, sendo que, neste caso específico, foi utilizado um robô ABB IRB4400 com capacidade de carga de 45 kg e com um alcance de 1,96 m.

1.3. CALENDARIZAÇÃO

Neste ponto é apresentada a calendarização das tarefas desenvolvidas ao longo da realização deste trabalho.

Tabela 1 Calendarização das tarefas do trabalho desenvolvido

Nome das etapas	Inicio	Fim	Outubro					Novembro				Dezembro				Janeiro					Fevereiro				Março					Abril				Maio					Junho			
			3	10	17	24	31	7	14	21	28	5	12	19	26	2	9	16	23	30	6	13	20	27	6	13	20	27	3	10	17	24	1	8	15	22	29	5	12	19	26
Estudo da documentação	05-10-2010	02-12-2010																																							
Simulação *offline* de um programa já existente no robô	26-10-2010	18-11-2010																																							
Elaboração do estado da arte	07-12-2010	08-02-2011																																							
Desenvolvimento *offline* de um programa	23-11-2010	16-12-2010																																							
Teste com programa efectuado *offline*	14-12-2010	16-12-2010																																							
Modelação tridimensional da célula	21-12-2010	06-01-2011																																							
Desenvolvimento offline de um programa nunca efectuado	01-01-2011	31-01-2011																																							
Desenvolvimento de uma peça para definição de workobjetcs	27-01-2011	04-02-2011																																							
Definição dos workobjects referentes a cada unidade	17-02-2011	24-02-2011																																							
Desenvolvimento offline de um programa para um novo produto	21-02-2011	22-02-2011																																							
Desenvolvimento de alguns melhoramentos em programas já existentes	23-02-2011	11-04-2011																																							
Realização de testes com os programas efectuados	12-04-2011	12-05-2011																																							
Elaboração de melhorias em programas já existentes	12-05-2011	30-06-2011																																							
Elaboração do relatório Final	03-01-2011	30-06-2011																																							

Estudo e pesquisa sobre as diversas aplicações existentes

Trabalho realizado recorrendo ao RobotStudio ou a uma das células existentes

Elaboração do relatório

1.4. ORGANIZAÇÃO DO RELATÓRIO

Ao longo deste documento irão ser explicados todos os problemas encontrados e respectivas soluções desenvolvidas, sendo que cada um dos capítulos é referente a um dos desenvolvimentos requeridos. Como tal, este documento encontra-se estruturado da seguinte forma:

- Estado da Arte – Neste capítulo será feita uma breve introdução e explicação à temática da programação *offline* de robôs bem como à utilização de robôs para o lixamento/polimento;
- Definição do Problema – Com este capítulo pretende-se explicar os problemas que estão associados ao lixamento das peças com a utilização de robôs sendo que, neste caso específico, será explicado com detalhe o lixamento de torneiras;
- Modelação das Células de Lixagem – Neste capítulo serão explicadas todas as fases que constituíram a modelação da célula, ou seja, será explicado como foram conseguidos os modelos da célula, como foram configurados e quais os ajustes efectuados;
- Desenvolvimento dos Programas dos Robôs – No desenvolvimento dos programas será explicado todo o processo de programação utilizando o RoboStudio e posteriores ajustes que foram necessários efectuar no sistema real; serão também apresentadas algumas imagens de peças lixadas com programas desenvolvidos *offline*, após algumas afinações;
- Conclusões – Por fim serão tiradas algumas conclusões de todo o trabalho realizado e serão apontadas algumas possibilidades para desenvolvimentos futuros.

2. ESTADO DA ARTE

Neste capítulo será feita uma breve introdução aos robôs industriais e, posteriormente, será apresentada uma breve descrição das soluções existentes na área onde se insere este trabalho, dando especial atenção à simulação e programação *offline* de robôs industriais e posteriormente às aplicações de robôs industriais no lixagem/polimento.

2.1. HISTÓRIA DOS ROBÔS INDUSTRIAIS

A história da robótica industrial deverá sempre começar com a devida homenagem à ficção científica. O aparecimento das palavras robô e robótica deve-se a um dramaturgo e a um escritor de ficção científica. Em 1922 Karel Capek usou pela primeira vez a palavra robot na sua obra Rossum's Universal Robots (RUR) e no início dos anos quarenta Isaac Asimov utilizou a palavra robótica para descrever a arte e ciência em que nós, profissionais da robótica, nos encontramos actualmente a trabalhar.

Ambos os escritores viam os robôs como meios poderosos de efectuar tarefas mas Capek pensava nos robôs como potenciais ameaças ao mundo dizendo mesmo na sua obra que os robôs iriam tomar conta do mundo; por outro lado, Asimov dizia que com a introdução de circuitos nos robôs se conseguiria manter a humanidade em segurança, pois esses circuitos impediriam os robôs de atacar os humanos, tornando-os pacíficos e amigáveis.

George C. Devol apresentou em 1954 um proposta para patentear a sua aplicação para um manipulador programável, tendo esta sido patenteada em 1961. Esta patente estava destinada a ser seguida por outros que pretendiam estudar como deveriam ser desenvolvidos os primeiros robôs. Devol agrupou em 1956 os seus pensamentos acerca das rotinas fabris e o seu entendimento sobre a tecnologia disponível que poderia ser aplicada no desenvolvimento de um robô.

Em 1956 Devol e Joseph Engelberger conheceram-se num cocktail, começando assim uma longa relação que deu origem à Unimation Inc. Com o desenrolar dos anos e após visitas a trinta e cinco fábricas (15 fábricas automóveis e 20 fábricas de produções diversas) a Unimation implementa o seu protótipo na General Motors em 1961.

A Kawasaki Heavy Industries adquiriu em 1968 todas as licenças da Unimation Inc. e em 1971, com o divulgar da tecnologia, foi criada a primeira associação robótica do mundo, que contra o que seria esperado foi criada no Japão e intitula-se Japan Industrial Robot Association (JIRA) [1][2].

A partir de 1976 os robôs começaram a baixar de preços de uma forma extremamente acelerada, graças ao desenvolvimento da microelectrónica, uma vez que esta veio facilitar o processamento com baixos custos de produção [3].

Apesar de a microelectrónica ser tida como existente desde os anos 50 só na década de 70 é que se começou a tirar real proveito das suas capacidades, pois como em qualquer nova tecnologia a aceitação do mercado e os preços das primeiras peças eram muito elevados para as empresas comuns. Como tal, com o evoluir das tecnologias e com o baixar dos preços de produção da microelectrónica, o preço dos robôs começou também a tornar-se aceitável e rentável para as grandes empresas; já as pequenas e médias empresas só mais tarde é que começaram a decidir investir em robôs devido ao enorme investimento necessário para adquirir e preparar um robô para cada tarefa.

Por meados da década de 70 apareceram também os primeiros Programmable Logic Controllers (PLCs) [4]. A diferença destes sistemas para os anteriores é que as suas funcionalidades não são determinadas pelo *hardware* mas sim pelo *software*, o que facilitou a programação. Até então as funcionalidades dos equipamentos estavam restringidas pelo *hardware*, o que implicava desenvolvimento de novo *hardware* para se implementar novas funcionalidades.

Com a junção destas duas tecnologias, o preço dos robôs começou a baixar e cada vez mais empresas começaram a optar por escolher este tipo de máquinas tão versáteis para efectuar as suas produções. Deste modo, conseguiam diversificar as suas produções sem necessidade de novas máquinas [3][4][5].

2.2. TIPOS DE ROBÔS

Devido ao enorme número de robôs existentes nos dias de hoje é necessário existir um termo de comparação entre eles para se poder identificar correctamente quais as suas funcionalidades e mais-valias para uma determinada área.

A forma mais usual de classificação de robôs é pela sua estrutura ou seja pelos graus de liberdade que o robô detém.

Tal como pode ser visto na Figura 1, os robôs industriais possuem normalmente seis eixos, a que correspondem seis graus de liberdade. Os graus de liberdade definem-se como sendo o número total de movimentos independentes que um dispositivo pode efectuar, enquanto os graus de mobilidade são definidos como sendo o número de juntas que o sistema detém.

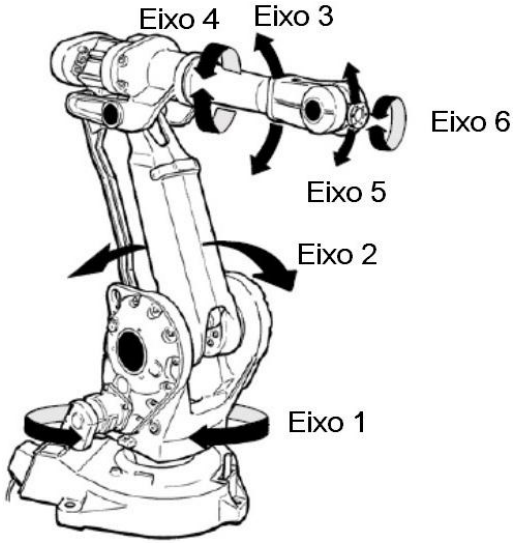

Figura 1 Elos e juntas de um robô com seis graus de liberdade [6]

Uma outra alternativa passa pela classificação dos robôs com base na sua estrutura. De acordo com esta classificação existem seis tipos principais de categorias, sendo estas:

- Cartesianos ou Gantry

  Os robôs cartesianos são constituídos por três juntas prismáticas. Na Figura 2 estão representados os movimentos do robô bem como o seu volume de trabalho;

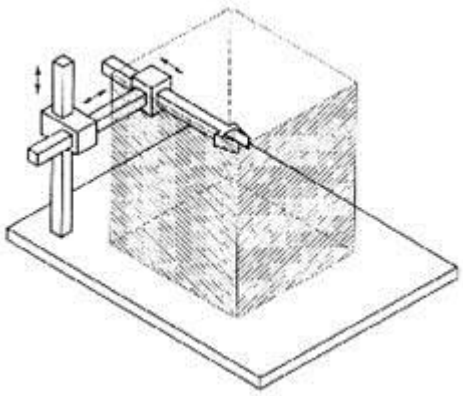

Figura 2 Configuração cartesiana

- Articulados

  Esta classificação é referente aos robôs mais utilizados no mercado, sendo estes normalmente constituídos por pelo menos três juntas rotativas. Na Figura 3 pode ver-se a representação deste tipo de configuração, e do correspondente espaço de trabalho;

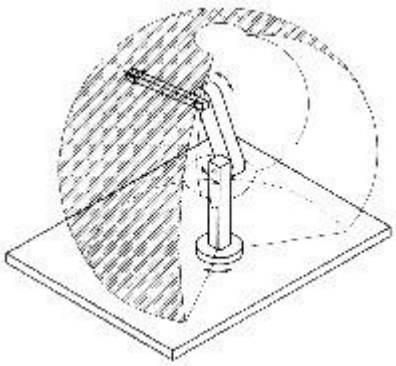

Figura 3 Configuração articulada

- Cilíndricos

  Os robôs cilíndricos utilizam duas juntas prismáticas e uma rotativa levando a que o seu volume de trabalho se assemelhe a um cilindro, tal como se pode ver na Figura 4, e daí o nome desta configuração;

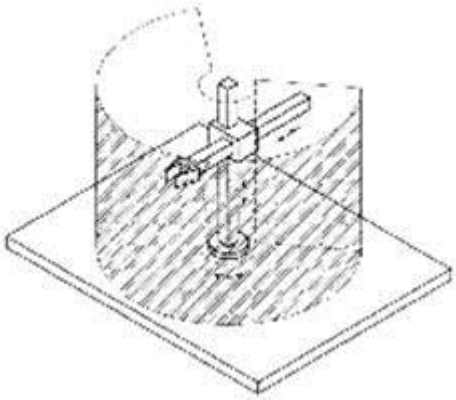

Figura 4 Configuração cilíndrica

- Esféricos

  A configuração esférica apresenta semelhanças com as configurações articulada e cilíndrica; neste caso, o tipo de juntas utilizadas por esta configuração difere ligeiramente das anteriores, uma vez que este tipo de configuração utiliza duas juntas rotativas e uma prismática, como se pode ver na Figura 5;

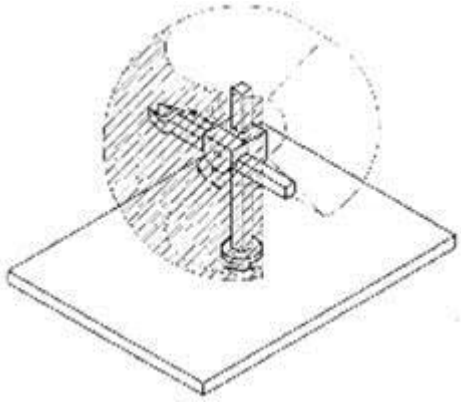

Figura 5 Configuração esférica

- SCARA

  A configuração Selectively Compliant Assembly Robot Arm ou Selectively Compliant Articulated Robot Arm (SCARA), é das configurações que permitem atingir velocidades de trabalho mais elevadas, mas com a ressalva que normalmente estes robôs são utilizados em aplicações do tipo *pick-and-place* em superfícies planas e horizontais, quer graças à sua velocidade, quer devido às restrições inerentes a esta configuração. Na Figura 6 pode ver-se um exemplo do volume de trabalho desta configuração.

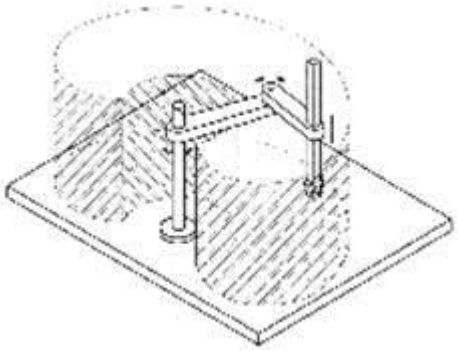

Figura 6 Configuração SCARA

- Paralelos

  Distinguem-se dos outros pela sua configuração diferente. Este tipo de robôs adopta geralmente duas configurações similares mas com objectivos diferentes, sendo estas:

  - *Tripod* (três braços), sendo que estes três braços interligam base, placa e *end-efector*;
  - *Hexapod* (seis braços) para conseguir usufruir de todo o espaço.

  Na Figura 7 pode-se ver um exemplo de cada uma das configurações paralelas mencionadas anteriormente.

Figura 7 Configuração paralela

Tendo por base estas definições, pode-se dizer que ao longo deste trabalho foram utilizados robôs com configuração articulada, pois para executar as tarefas pretendidas são estes que oferecem um maior volume de trabalho, o que facilita a correcta execução das tarefas pretendidas [1][7].

Alternativamente os robôs podem ser subdivididos nas duas categorias seguintes:

- Robôs controlados por computador: esta categoria pode ser dividida em duas subcategorias mas ambas se regem pela mesma base, ou seja, estes robôs repetem o que lhes foi programado podendo interagir com sensores ou com o operário para um melhor funcionamento. Quanto às duas subcategorias, podem ser classificadas como:

- Sem controlo-servo: neste tipo de robôs é o programa que controla o movimento dos diferentes componentes, e este movimento realiza-se através de posicionamento "ponto-a-ponto" no espaço;
- Com controlo-servo: este tipo de robôs permite duas formas distintas de trabalho: uma forma de controlo permite que os movimentos dos elos do robô se efectuem em função dos seus eixos, sendo que os movimentos podem ser realizados ponto-a-ponto ou com trajectória contínua e a outra forma de trabalho permite que os movimentos se realizem tendo em conta a posição dos eixos de coordenadas e a orientação da ferramenta (*end-efector*) do robô.

- Robôs Inteligentes: são controlados por sistemas multifuncionais, sendo capazes de interagir com seu ambiente através de sensores e de tomar decisões em tempo real. Actualmente o desenvolvimento destes robôs tem vindo a torná-los cada vez mais próximos do ser humano, ou seja, cada vez mais estes robôs são desenvolvidos para se parecer com humanos e ter reacções de humanos (sentimento, humor, dor, etc.) [3].

2.3. IMPACTO DOS ROBÔS INDUSTRIAIS NA SOCIEDADE

Os benefícios teóricos de utilizar robôs na indústria são numerosos e vão desde o aumento da produtividade, à melhoria e à consistência na qualidade final do produto (a qual também minimiza a necessidade de operações adicionais), à menor contratação de mão-de-obra especializada (difícil de encontrar), à fiabilidade do processo, à facilidade na programação e uso dos robôs, à operação em ambientes difíceis e perigosos ou em tarefas desagradáveis e repetitivas para o ser humano e à capacidade de trabalho sem interrupções por longos períodos de tempo [8].

Na prática existem alguns problemas com a aceitação total das vantagens anteriormente referidas num curto espaço de tempo. O investimento inicial para adquirir robôs é elevado e existe sempre o estigma do despedimento associado à compra de robôs, uma vez que existe a ideia formanda de que comprar um robot implica despedir funcionários o que não é totalmente verdade. Com a introdução de um robô para a produção de uma peça, apesar de isso implicar que vários funcionários deixem de efectuar essas funções, os postos que sucedem esse ponto da produção irão ficar com baixa resposta devido à elevada taxa de produção dos robôs face ao ser humano. Isto não é sempre obrigatório acontecer, mas normalmente é o caso mais frequente.

Em algumas operações industriais, tal como na manipulação de materiais, soldadura por resistência, soldadura por arco eléctrico e pintura, o sucesso da introdução de robôs deve-se à boa relação custo-benefício, pois para além de substituir a mão-de-obra humana num trabalho repetitivo, difícil e, muitas vezes, de alto risco, as taxas de produção e a qualidade dos produtos é constante e elevada.

Tendo em conta tudo o que foi referido anteriormente, pode dizer-se que a introdução dos robôs nos meios industriais é muita vezes uma situação delicada e de difícil aceitação pelas empresas, sendo que actualmente e com a massificação de produções (enormes quantidades de produtos são necessárias todos os dias) começa-se a ver cada vez mais a utilização de robôs nas diferentes áreas da indústria, e com especial atenção nas indústrias alimentares e nas indústrias farmacêuticas.

Nas indústrias alimentares essa implementação deve-se às enormes quantidades produzidas devido à constante procura e aumento da população. Na indústria farmacêutica o aumento da utilização de robôs deve-se à necessidade de produções massivas em ambientes esterilizados, o que é fácil recorrendo a robôs, uma vez que estes não respiram nem se deslocam, como acontece com as pessoas, levando ao aumento de partículas e micróbios no ar.

2.4. SIMULAÇÃO E PROGRAMAÇÃO *OFFLINE* DE ROBÔS INDUSTRIAIS

A programação *offline* de robôs industriais refere-se à capacidade de transferir programas criados em simuladores para a célula real, ou fazer o inverso, transferindo programas desenvolvidos no próprio robô para o simulador [9]. Como tal é necessário compreender o que é programação *offline*, como deve ser implementada e quais as suas vantagens e limitações.

Para a correcta implementação de um sistema de programação *offline* é necessário cumprir alguns requisitos sem os quais a programação não será eficaz. Para explicar melhor os principais requisitos, estes foram divididos por pontos, tal como apresentado de seguida.

- Uma correcta programação *offline* está directamente ligada e dependente de sistemas Computer-aided design (CAD)/Computer-aided manufacturing (CAM), pelo que é necessário que o sistema *offline* possua a capacidade de representação gráfica do modelo do "mundo";

- É necessário que o sistema detenha informações sobre o processo, ou tarefas, a serem programadas;
- O sistema deverá possuir uma boa representação da geometria, cinemática e dinâmica dos robôs;
- Após satisfeitos os requisitos dos três pontos anteriores, o sistema não será eficaz se não conseguir reconhecer e utilizar todos os dados de forma correcta;
- Terminada essa interligação, o sistema depara-se com outro problema, sendo este a verificação dos programas produzidos, uma vez que estes programas podem conter erros, tais como colisões;
- Realizado e confirmado o programa, é agora necessário que exista uma interface de comunicação adequada, quer ao controlador do robô, quer ao sistema de programação *offline*;
- Por último, existe a necessidade de uma interface homem-máquina adequada e amigável para facilitar ao programador a utilização de todos os seus conhecimentos aquando da utilização do sistema de programação *offline*.

Sendo conseguido tudo o que foi referido anteriormente consegue-se ter um bom sistema de programação *offline*; mas mesmo tendo um bom sistema, continuam a existir problemas com a realização de programações *offline*, podendo dividir-se estes problemas em três áreas distintas, a saber:

- Modelação e programação

  A área da modelação e programação pode ser subdividida em três sub-áreas diferentes que são:

  - Modelo geométrico;
  - Modelo do robô;
  - Método de programação.

  Estas três sub-áreas estão directamente ligadas à correcta modelação do sistema real (célula e robô), bem como à correcta utilização dos diferentes componentes da célula por parte do programador e posterior facilidade na transferência dos programas entre sistema real e sistema *offline* graças à enorme diversidade de robôs existentes (existem diversas marcas e modelos o que complica a transferência entre sistemas);
- Interface

  Mais uma vez a enorme variedade de linguagens de programação para robôs dificultam a construção de uma interface *standard* para o desenvolvimento *offline*; por isso, tal

como acontece com a modelação e programação, o trabalho conjunto entre fabricantes de robôs iria ajudar à normalização das interfaces e, para que isso fosse possível, podiam ser tidos em conta três pontos principais:

- Sistemas de programação;
- Sistemas de controlo;
- Formato dos programas.

Com a uniformização destes a criação de interfaces seria muito facilitada e, por sua vez, a aprendizagem e funcionamento seriam também facilitadas;

- Erros ou desvios no sistema real

  Por fim, os erros ou desvios no sistema real são os factores mais complicados de analisar e melhorar num sistema de programação *offline*. Estes dependem de inúmeros factores que vão desde o fabrico dos diversos componentes do robô até ao desgaste que estes componentes têm com as horas de funcionamento. Para uma melhor percepção destes problemas face a sistemas *offline*, dividiram-se estas discrepâncias em quatro pontos principais:

  - Robô

    O robô é um dos principais factores envolvidos nestes erros, pois a falta de rigidez da estrutura onde está assente, a falta de tolerâncias na construção do robô ou a diferença existente entre robôs (robôs da mesma marca e modelo podem ter diferenças significativas) podem levar ao aparecimento de desvios consideráveis quando se utilizam sistemas de programação *offline*;

  - Controlador do robô

    O controlador pode ter problemas com a resolução. A resolução representa o menor incremento de movimento que um robô pode fazer e aquando da programação *offline* esse incremento pode ser inferior ao comportado pelo controlador;

  - Célula de trabalho

    O principal problema deste ponto deve-se ao facto de ser extremamente complicado conseguir obter uma localização precisa de todos os componentes que constituem a célula (robô, máquinas, ferramentas), bem como aos efeitos secundários originados pelo funcionamento do robô na célula que podem alterar o desempenho do robô (a temperatura é um dos factores que pode alterar desempenho do robô);

  - Modelo e sistema de programação

Por fim, a correcta modelação da célula irá dar a precisão do programa; como tal, se a modelação não for correctamente desenvolvida ou se o sistema de programação *offline* não permitir obter muito detalhe com essa programação o programa desenvolvido terá desvios que podem ser consideráveis e causar problemas ou danos na célula.

Posto isto, conclui-se que para se conseguir bons sistemas de programação *offline* será necessário que no futuro exista mais interligação e comunicação entre os diferentes fabricantes de robôs e sistemas de programação *offline*. Como será perceptível no fim deste trabalho, a programação *offline* pode (e deve) ser uma mais-valia no desenvolvimento de novos programas [1].

Por tudo o que foi referido anteriormente, pode-se então dizer que se a célula real do robô e toda a modelação três dimensões (3D) estiverem em concordância, os programas podem ser testados no simulador e posteriormente podem entrar directamente em funcionamento na célula real, uma vez que o comportamento do robô será igual ao visualizado no simulador.

Com tudo isto, como será de prever, a grande vantagem da programação *offline* prende-se com a minimização dos tempos de paragem dos equipamentos produtivos para alteração dos programas e com a capacidade de testar a viabilidade da utilização do robô no fabrico de uma peça ainda antes de esta ser produzida; ou seja, na fase de desenvolvimento de novas peças, antes de se decidir se a peça irá ser produzida ou não, pode recorrer-se ao simulador para testar os eventuais problemas que a produção dessa peça trará.

Actualmente, com o aumento da diversidade de produtos, é preciso ter em conta os tempos de paragem para o desenvolvimento de um novo programa para a produção. Assim é necessário minimizar esses tempos de paragem, uma vez que a redução do tempo de paragem de um robô para o desenvolvimento de um novo programa representa o aumento de lucros e a continuidade da produção de uma linha - como tal todas as empresas têm interesse em desenvolver os seus programas *offline*. Graças a esta necessidade, têm aparecido diversas aplicações que facilitam a programação *offline*, sendo que estas são normalmente desenvolvidas para diferentes tipos de processos, podendo uma mesma aplicação ser utilizada em diferentes processos. Por exemplo, o programa disponibilizado pela ABB (ABB RoboStudio) vem já com diferentes bibliotecas, nas quais se podem

encontrar todos os modelos de robôs disponibilizados pela ABB, bem como uma série de outros equipamentos disponibilizados também pela marca. É de salientar que podem ser desenvolvidos mecanismos utilizando o programa da ABB, bem como carregar mecanismos desenvolvidos noutros simuladores 3D. Desta forma, no caso de já existir um modelo 3D de um mecanismo o programa permite que este seja carregado e posteriormente configurado (este processo irá ser explicado na secção 4.3).

2.5. APLICAÇÕES PARA DESENVOLVIMENTO *OFFLINE* DE PROGRAMAS PARA TAREFAS DE LIXAMENTO COM ROBÔS

Existem diferentes tipos de aplicações passiveis de serem utilizadas em áreas como o lixamento, mas é de salientar que das diferentes aplicações analisadas nenhuma detinha qualquer tipo de especificação para estas tarefas. O estudo incidiu sobre os *softwares* desenvolvidos por diferentes empresas e denotou-se que, para algumas aplicações, a oferta é vasta mas existem outras áreas ainda pouco exploradas e desenvolvidas no que toca à programação *offline*. Finda essa pesquisa decidiu-se apresentar uma explicação e alguns exemplos dos *softwares* estudados.

Os *softwares* para desenvolvimento de programas *offline* caracterizam-se normalmente de duas formas diferentes:

- Proprietários

  São os *softwares* dos fabricantes dos robôs. Normalmente os fabricantes detêm programas específicos para facilitar a programação *offline* dos seus robôs e cada fabricante fornece ainda duas hipóteses para utilização dos seus *softwares*, sendo estes dois tipos de *softwares*:

  - Genéricos: Este tipo de *software* permite a facilidade de utilização em diferentes aplicações, mas essa facilidade implica o aumento de dificuldade da utilização em alguns casos específicos;
  - Dedicados: São normalmente desenvolvidos para tarefas muito específicas e facilitam a programação dos robôs nessas tarefas. Os casos mais comuns da utilização deste tipo de *software* são a pintura e soldadura.

Relativamente aos *softwares* ditos proprietários podem-se referir, a título de exemplo, os que se encontram enumerados de seguida:

- A ABB dispõe do *software* ABB RobotStudio (utilizado na realização deste trabalho) [10];
- KUKA: A KUKA fornece o seu *software* próprio chamado de "KUKA SIM" [11];
- Fanuc: A Fanuc detém um *software* privado para a programação *offline* dos seus robôs, sendo este chamado de OlpcPRO [12];

- Abertos

  Os *softwares* abertos são aplicações com a capacidade de serem utilizadas em diferentes marcas de robôs, isto é, são aplicações com capacidade de compilar o código para ser compatível com os diferentes requisitos de cada uma das marcas e, tal como no caso dos *softwares* proprietários, podem ser subdivididos em dois tipos diferentes, sendo estes caracterizados da mesma forma que no caso dos *softwares* proprietários: genéricos ou dedicados.

  Relativamente aos *softwares* abertos podem-se enumerar alguns casos, a título de exemplo:

  - Camelot: O *software* disponibilizado por esta empresa é muito semelhante ao disponibilizado pela ABB, mas pode ser utilizado para diferentes robôs, sendo apenas necessária a correcta modelação 3D de cada um dos dipositivos utilizados [13];
  - Delfoi: A Delfoi fornece *softwares* semelhantes aos anteriores mas com a particularidade de desenvolver *softwares* mais específicos para determinadas tarefas, isto é fornece um *software* para lixamento, outro para soldadura, etc., e com isso consegue manter a especificidade e o aperfeiçoamento da programação para essas tarefas [14];
  - Dynalog: A Dynalog fornece essencialmente *softwares* para facilitar as multifuncionalidades de cada robô, ou seja fornece *software* com capacidade de avaliar correctamente os espaços e respectivas restrições para ser mais fácil alterar a programação de um robô quando existe a necessidade de efectuar alguma alteração [9];
  - Delmia: A Delmia oferece sistemas de "Virtual Commissioning", isto é, oferece sistemas com a capacidade de modelação 3D para testar produtos e novos *layouts* ou implementações dos componentes já existentes, podendo assim garantir o seu correcto funcionamento quando necessário. Com este tipo de *software* podem testar-se possíveis falhas e detectar possíveis problemas que poderão ocorrer quando se muda alguma particularidade de um sistema [15];

- Tecnomatix (Siemens PLM): É um *software* fornecido pela Siemens para simulação *offline* de programas, fornecendo diversas capacidades ao utilizador para testar, simular e validar possíveis implementações dos programas para as células reais [16].

Para além destas empresas existem outras que normalmente trabalham no desenvolvimento de soluções para casos específicos, mas também fornecem soluções *standard* para aplicações amplamente utilizadas. Apenas foram referidas estas por se tratarem das mais conhecidas.

É de salientar que grande parte destas aplicações são indicadas para situações onde as superfícies são planas, isto é, são aplicações que facilitam a programação para superfícies a serem lixadas, superfícies estas que não podem ser muito complexas (no caso das torneiras, devido à irregularidade quer da superfície, quer à complexidade do próprio desenho, estes programas não são muito precisos), pois com o aumento da complexidade da peça aumenta também o número de pontos e reorientações necessárias, o que é difícil de programar e visualizar correctamente utilizando o simulador.

3. DEFINIÇÃO DO PROBLEMA

Neste capítulo será feita a descrição do processo de fabrico das torneiras e a identificação do problema adito à realização de tarefas de lixamento/polimento recorrendo à utilização de robôs.

3.1. GROHE PORTUGAL

A Grohe Portugal é uma parte significativa da organização fabril do grupo alemão Grohe e especializa a sua produção em torneiras de gama média, do tipo monocomando, para casa de banho. A especialização numa gama restrita de produtos, com o seu fabrico específico e estrutura de suporte, permite que a empresa seja bastante competitiva sem sacrificar qualquer dos requisitos de qualidade que tornam a Grohe a marca líder mundial.

O processo de produção, desde a fundição, maquinagem, acabamento superficial, cromagem e montagem formam a competência-chave da empresa. Estas competências são fortalecidas através de uma rede de fornecedores locais que, por sua vez, apresentam vantagens ao nível dos custos com alta performance de qualidade.

A empresa foi constituída em 1996, com a criação do Departamento Comercial do Porto e a sua equipa de suporte de vendas. Um ano mais tarde, mais precisamente em Outubro de

1997, é concretizada a construção da unidade de produção em Albergaria-a-Velha, sendo oficialmente inaugurada em 28 de Maio de 1998.

3.2. DESCRIÇÃO DO PROCESSO DE FABRICO

A empresa Grohe Portugal é responsável pela produção de alguns dos modelos pertences à marca. Para esta produção são necessárias cinco fases distintas, que se passam a listar e a descrever:

- Fundição

  É nesta fase que se inicia a produção de cada um dos modelos. Para tal é efectuada a fundição das matérias-primas necessárias à obtenção de uma liga de latão a ser utilizada para o fabrico das peças. Na Figura 8 pode ver-se uma imagem de um dos fornos utilizados para esse fim.

Figura 8 Fornalha onde são fundidos os materiais

Tal como se pode ver na Figura 9, o processo de fundição está dividido em quatro etapas sendo estas: fundição da liga de latão/criação dos "machos", vazamento, corte e grenalhadora.

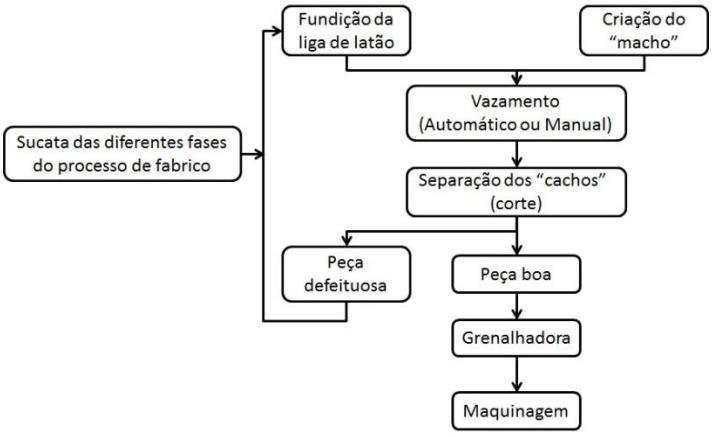

Figura 9 Fluxograma da fase de fundição

Na primeira etapa introduz-se matéria-prima num de dois fornos de fusão central, fornos esses que funcionam a temperaturas que rondam os 1100ºC. Após a fundição da matéria-prima o metal resultante é analisado para avaliar se as percentagens de materiais contidos estão correctas, ou seja, se as quantidades dos diferentes tipos de elementos de liga utilizados estão adequados aos requisitos exigidos pela marca, uma vez que todo o metal líquido só é considerado apto para passar à fase seguinte após essa análise num espectrómetro. Após a aprovação dessa solução, esta é distribuída pelas diferentes máquinas (coquilhadoras) onde será vazada a peça.

Para fazer a cavidade interior das torneiras são utilizados "machos" (postiço que dá a configuração interna da torneira). Os "machos" são produzidos através de uma caixa de "machos", sendo que para a sua produção é utilizada areia sílica, endurecedor e resinas.

Na segunda etapa são utilizados dois tipos de vazamento:

- Baixa Pressão (BP) – Estas máquinas funcionam por injecção de metal líquido na respectiva coquilha: o operador insere o "macho" na máquina e esta injecta o metal líquido;
- Manual (gravidade) – O operador insere inicialmente o macho na coquilha, tal como acontece no processo anterior, seguidamente recolhe metal líquido de um forno utilizando uma colher e, por fim, recorrendo ao efeito da gravidade, vaza o metal líquido para o interior do respectivo "macho".

Após o arrefecimento das peças passa-se à terceira etapa onde é necessário efectuar o corte do "cacho" (apoio central entre os dois moldes denominado como gitos de alimentação): como cada "macho" contém o molde de duas ou mais peças, estas têm de ser separadas do seu apoio no "macho". Finda esta etapa, passa-se então à quarta e última etapa, sendo que esta é efectuada recorrendo a máquinas denominadas Grenalhadoras. Estas máquinas são responsáveis pela libertação de toda a areia proveniente dos "machos" que fica retida no interior das peças. Para isso, as peças são introduzidas no interior da máquina e esta, recorrendo a um tambor rotativo associado à projecção de esferas metálicas, através da vibração retira a totalidade da areia existente no interior das peças. Estas máquinas também são responsáveis por preparar as peças para o posterior lixamento, pois o impacto das esferas na superfície da peça retira-lhe a superfície vidrada originária da fundição, superfície esta que dificulta o lixamento.

- Maquinagem

  Como se pode ver na Figura 10, esta fase está divida em três etapas, sendo estas: maquinação, desengorduramento e teste de estanquidade.

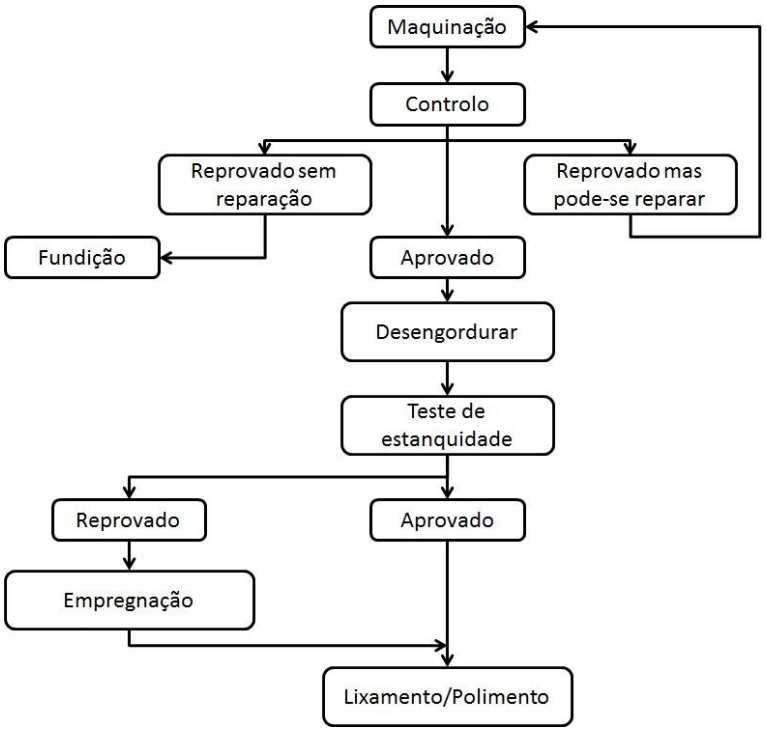

Figura 10 Fluxograma da fase de maquinagem

A secção de maquinagem, referente à primeira etapa, é responsável por efectuar todas as furações, criação de roscas em todas as peças e realizar ligeiros ajustes em alguns tipos de peças. Para isso, são utilizadas máquinas Computer Numerical Control (CNC).

Na Figura 11 é apresentada uma vista superior de uma das células utilizadas na maquinagem das peças com um pormenor de uma "cabeça" que efectua o processo.

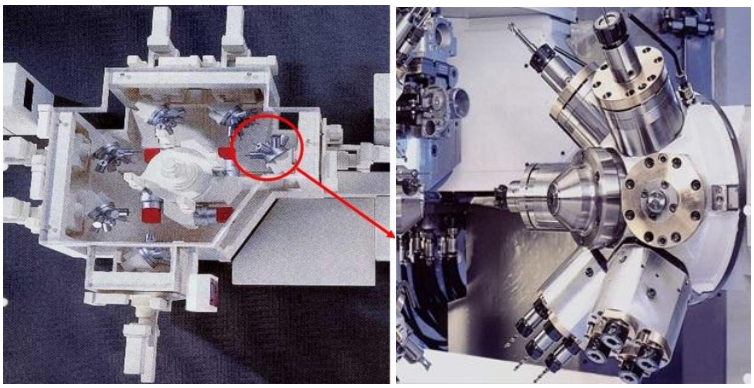

Figura 11 Triflex para maquinação das peças

Após este processo é efectuado um controlo visual e um controlo recorrendo a calibres passa-não-passa (efectuado pelo próprio operador da máquina). Findo o controlo das peças passa-se então à segunda etapa, onde as peças são lavadas e desengorduradas para facilitar o lixamento e para eliminar toda a gordura proveniente dos banhos de lubrificação aquando do processo de maquinação. Por fim chega-se à última das etapas, onde algumas das peças são submetidas a um teste de estanquidade.

- Lixamento/Polimento

  Após a maquinagem, passa-se para a fase de lixamento/polimento onde é efectuado o lixamento e o polimento das peças. Tal como se pode ver pela Figura 12, esta fase está dividida em duas etapas principais sendo elas o lixamento e o polimento.

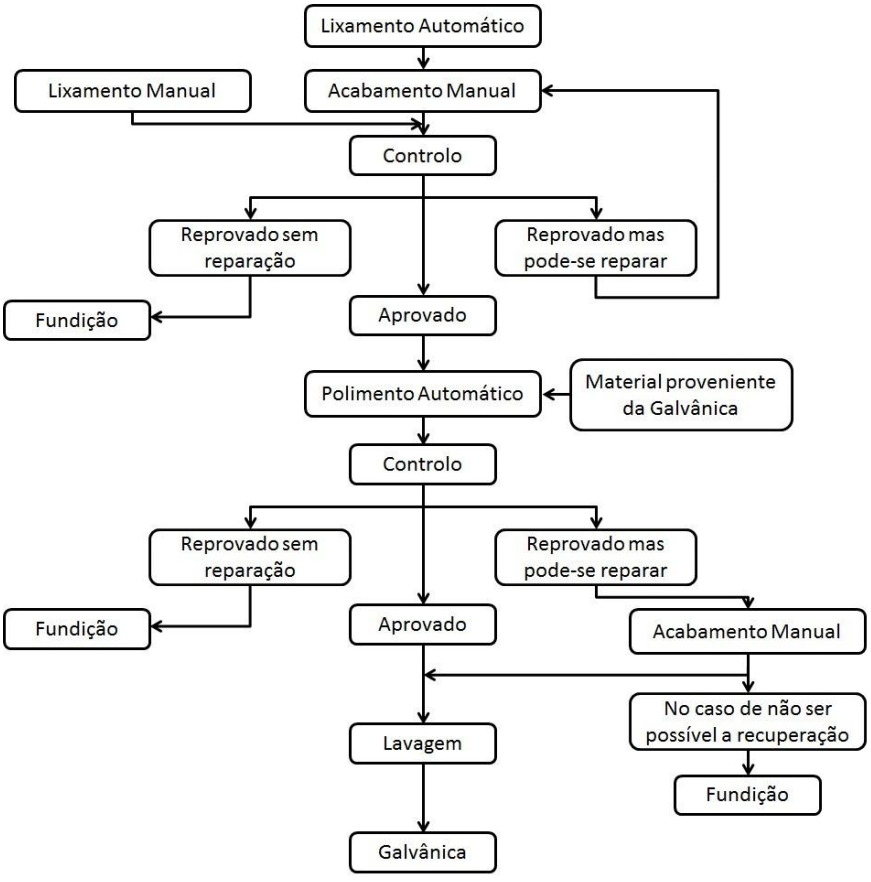

Figura 12 Fluxograma da fase de lixamento/polimento

Na primeira etapa executa-se o processo de lixamento, podendo este ser efectuado de duas formas diferentes:

- Lixamento automático – Neste caso recorre-se à utilização de robôs ABB IRB4400 com capacidade de carga de 45 kg.

Estes robôs podem efectuar qualquer tarefa, sendo que no modelo de funcionamento actual da Grohe Portugal os robôs são programados *online*, ou seja, pára-se a célula e programa-se o robô para a tarefa pretendida;

- Lixamento manual – No lixamento manual, tal como se pode depreender do nome, esta tarefa é efectuada por um operador experiente e utilizando máquinas de lixamento manual ou combis (máquinas que permitem lixar e/ou polir). A Figura 13 mostra um operador a trabalhar numa dessas máquinas. Este modo de lixamento é utilizado muitas vezes para dar o acabamento a peças que os robôs não conseguem realizar na totalidade ou no caso de ser mais vantajoso efectuar estas operações manualmente do que utilizando os robôs, quer por ser mais rápido, ou mesmo por ser mais eficiente.

Figura 13 Processo de lixamento manual

De notar que nesta etapa todas as peças passam por dois tipos diferentes de lixa para obter um melhor acabamento. Na primeira fase é retirada grande parte da película de fundição utilizando lixa de grão 80; na segunda fase é dado um acabamento final à peça utilizando lixa de grão 280. A utilização de dois tipos de lixa deve-se ao facto de se tentar obter o melhor acabamento, isto é, baixar a rugosidade superficial o mais possível. Para tal, tenta-se também cruzar o lixamento nas diferentes fases para evitar a criação de sulcos e para ser possível a eliminação de riscos criados na primeira fase de lixagem.

Na etapa de polimento, tal como acontece no lixamento, existem dois tipos diferentes de funcionamento:

- Polimento automático – No polimento automático são utilizadas células dedicadas a essa tarefa, ou seja, células projectadas para a tarefa de polimento. Estas células utilizam o polimento por imersão, isto é, as peças são polidas

recorrendo a uma escova (composta por dois tipos de tecido, sendo estes algodão e poliéster) embebida em pasta abrasiva. A Figura 14 mostra um dos modelos de máquinas de polimento automático utilizada para polir as peças. É de notar que a máquina se encontra aberta e parada;

Figura 14 Processo de polimento automático

- Polimento manual – O polimento manual apenas é utilizado para recuperar algumas peças, após ser efectuado o polimento automático (caso seja necessário), ou em algumas peças específicas.

Quando se termina o lixamento/polimento todas as peças passam por uma inspecção visual. No caso de conterem defeitos passíveis de serem reparados, estas reparações são efectuadas manualmente.

De notar que algumas das peças, devido à complexidade do seu desenho interior, requerem ser lavadas antes de passarem para a secção da galvânica, para que seja reduzida a possibilidade de contaminação dos banhos existentes nessa fase.

- Galvânica

  Chegados a esta fase, e como se pode ver pela Figura 15, as peças necessitam de ser desengorduradas para posteriormente serem niqueladas e cromadas.

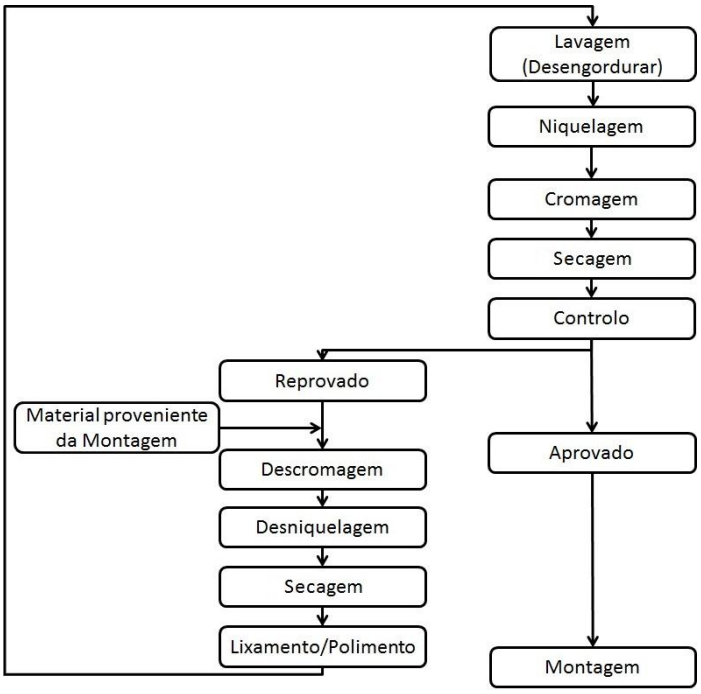

Figura 15 Estrutura da fase de galvânica

Esta fase pode ser dividida em três grandes etapas, sendo estas as seguintes:

- Lavagem (desengordurar)

  As peças são lavadas e desengorduradas para evitar a contaminação dos tratamentos que irão ser administrados posteriormente, sendo que nesta etapa as peças são activadas para receberem os tratamentos seguintes;

- Niquelagem

  Utiliza-se electrodeposição de níquel; este processo é efectuado com a imersão das peças num banho de níquel brilhante a uma temperatura de 60ºC, tendo esse banho uma duração variável entre 10-12 min. Este tratamento serve para dar um aspecto brilhante à peça;

- Cromagem

  Utiliza-se electrodeposição de crómio; este processo efectua-se com a imersão das peças num banho de crómio a uma temperatura de 40ºC, tendo esse banho uma duração variável entre 4-5 min. Este tratamento serve para dar resistência à corrosão nas peças não niqueladas e para garantir o brilho nas peças que são niqueladas.

De notar que os dois últimos processos necessitam da activação dos tratamentos através de corrente eléctrica, sendo que esta corrente varia entre 4-11 A/dm^2, dependendo do tratamento a ser aplicado.

Na Figura 16 pode ver-se as suspensões com as torneiras durante o processo efectuado na galvânica.

Figura 16 Processo de lavagem, niquelagem, cromagem

Estando estas etapas concluídas, as peças são novamente inspeccionadas. No caso de serem aprovadas passam para a fase de montagem; no caso de serem rejeitadas as peças são descromadas, desniqueladas e, caso seja necessário, passam novamente pela fase de lixamento/polimento, com o intuito de tentar corrigir os eventuais defeitos (caso seja possível).

De notar que apesar das verificações anteriores, nesta fase é normal aparecerem peças defeituosas, uma vez que alguns defeitos apenas são visíveis após os tratamentos levados a cabo nesta fase do processo de fabrico.

- Montagem

  Chegados a esta fase o processo subdivide-se em duas etapas, tal como se pode ver na Figura 17. No término da segunda etapa (Inspecção e teste) a peça é dada como finalizada quanto ao seu processo de produção.

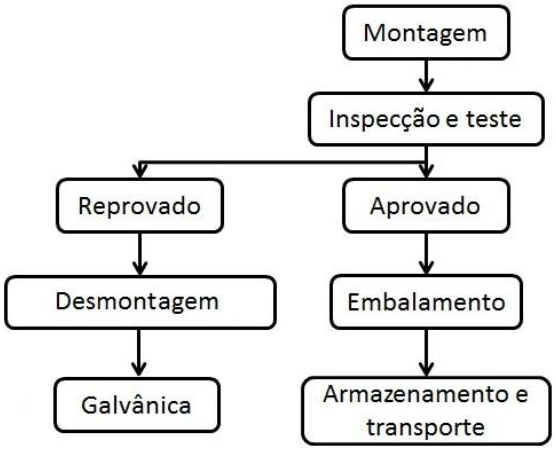

Figura 17 Estrutura da fase de montagem

Na fase de montagem é efectuada a montagem de todos os componentes da torneira e o posterior embalamento da mesma.

Na Figura 18 pode ver-se uma célula de montagem com as três operadoras a trabalhar.

Figura 18 Célula de montagem manual

Após esta fase, e apesar da existência de uma secção de armazenamento (que apenas existe para manter pequenos *stocks* e armazenar temporariamente algumas peças), as peças são carregadas em camiões logo após o término da sua produção para posterior transporte.

3.3. PROBLEMAS NO LIXAMENTO DAS TORNEIRAS

Tendo sido explicado todo o processo de fabrico de uma peça interessa agora analisar o problema que motivou a realização deste trabalho. Esse problema vai desde as longas paragens das células para efectuar a programação *online* para um novo produto, até à constante mudança de produto o que implica paragens para *setup*. Como tal, com este trabalho decidiu-se tentar melhorar (reduzir) os tempos de paragem associados à programação de novos produtos recorrendo à utilização de programação *offline*. Para conseguir esta redução efectuou-se um estudo para identificar os principais problemas adjacentes às tarefas de lixamento e agruparam-se os principais problemas como apresentado em baixo.

É de notar que os problemas adjacentes as estas tarefas, tal como se pode imaginar, devem-se essencialmente à complexidade das peças e à necessidade de efectuar movimentos coordenados em diferentes eixos mantendo, por exemplo, o paralelismo com a roda.

Posto isto, enumeram-se aqui alguns dos principais problemas que surgem no lixamento robótico das torneiras:

- Limitações de eixos dos robôs

  Em todos os processos de lixamento efectuados por robôs é necessário ter sempre em mente que existem diversas limitações de movimentos e também a possibilidade de se atingir os limites dos eixos do robô. Assim, mesmo conseguindo efectuar determinado movimento, apesar da sua complexidade, pode não ser possível efectuar a totalidade do movimento devido a um dos eixos do robô exceder o seu limite máximo. No caso do lixamento de uma determinada peça as entradas e saídas da lixa devem ser feitas suavemente (por entradas e saídas da lixa entenda-se que se fala do momento quando se encosta ou desencosta a peça à lixa). Com este intuito existe muitas vezes a necessidade de fazer rotações de 180º, rotações estas que acabam por não ser possíveis por se atingir o limite de algum dos eixos do robô. Este entrave aconteceu diversas vezes em alguns dos programas desenvolvidos ao longo deste trabalho no qual os robôs atingiram enumeras vezes o limite do eixo cinco, obrigando assim à restruturação do movimento para ser possível executar o correcto lixamento da peça.

- Cumprimento das concordâncias das linhas das peças

  Os problemas ligados à concordância das linhas da peça que nem sempre são respeitadas, ou seja algumas peças contêm formas distintas e alguns vincos que devem ser mantidos o que quando se efectua uma programação *online* é complicado conseguir, pois para manter essas linhas e vincos está-se dependente do "olho" humano que nem sempre avalia a situação da melhor forma.

- Falhas nos processos anteriores ao lixamento

  Outro dos problemas do lixamento é o problema que reside nas duas fases anteriores (fundição e maquinação). Como o robô não tem qualquer tipo de controlo visual para verificar a concentricidade das peças ou a existência de mais ou menos poros[1], o lixamento da peça defeituosa origina uma peça para sucata e, por vezes, pode também danificar algum componente da célula.

[1] Poros são as irregularidades provenientes do processo de fundição, onde a existência destes "buracos" pode ser melhorada mas raramente pode ser anulada.

Como tal, é necessário avaliar com detalhe cada uma das peças bem como a forma como será feito o programa para essa peça, de forma a conseguir retirar o máximo partido da célula, com tempos de produção reduzidos e com a máxima fiabilidade no processo.

4. MODELAÇÃO DAS CÉLULAS DE LIXAGEM

Neste capítulo será explicado o processo utilizado aquando da modelação das células de trabalho. Esta modelação é essencial para a correcta programação dos movimentos dos robôs e dos restantes equipamentos que os constituem sem que ocorram colisões.

4.1. AVALIAÇÃO DA CÉLULA DE TRABALHO

Para a realização de um correcto modelo a 3D foi necessário avaliar a célula existente e determinar quais seriam os componentes da referida célula necessários à boa representação no RobotStudio.

Como se pode ver na Figura 19, cada célula de fabrico contém quatro unidades de lixamento, sendo que as duas unidades centrais usam rodas de 150 mm de diâmetro e as duas unidades das extremidades usam rodas de 450 mm de diâmetro (esta é a configuração habitual das células podendo existir variações em casos específicos).

Figura 19 Unidades de lixagem presentes na célula do robô

Após a análise das respectivas unidades iniciou-se a modelação 3D de cada uma das unidades. Foi também necessário decidir como seria feita a modelação e que componentes seriam inseridos na respectiva modelação.

Efectuada essa análise decidiu-se que apenas seria necessário a representação gráfica das rodas, braços e "corpo" da parte superior da unidade, tal como se poderá verificar no ponto seguinte.

4.2. MODELAÇÃO DA CÉLULA

Para efectuar a modelação da célula recorreu-se à utilização de um programa de desenho 3D, especificamente o SolidWorks.

Recorrendo a este programa, efectuou-se o desenho de cada uma das peças necessárias para a correcta modelação da célula. Iniciou-se essa modelação pelos braços que sustêm as rodas, pois se verificou que seria uma parte fulcral para o correcto posicionamento e representação de cada uma das unidades.

De notar que os braços que suportam as rodas de 450 mm são diferentes dos que suportam as rodas de 150 mm, sendo que apenas serão demonstrados aqui, a titulo exemplificativo, alguns dos modelos 3D.

Tal como se pode ver pela Figura 20, a concentricidade das rodas está dependente do correcto alinhamento do braço, uma vez que o braço é responsável pela estabilidade da roda.

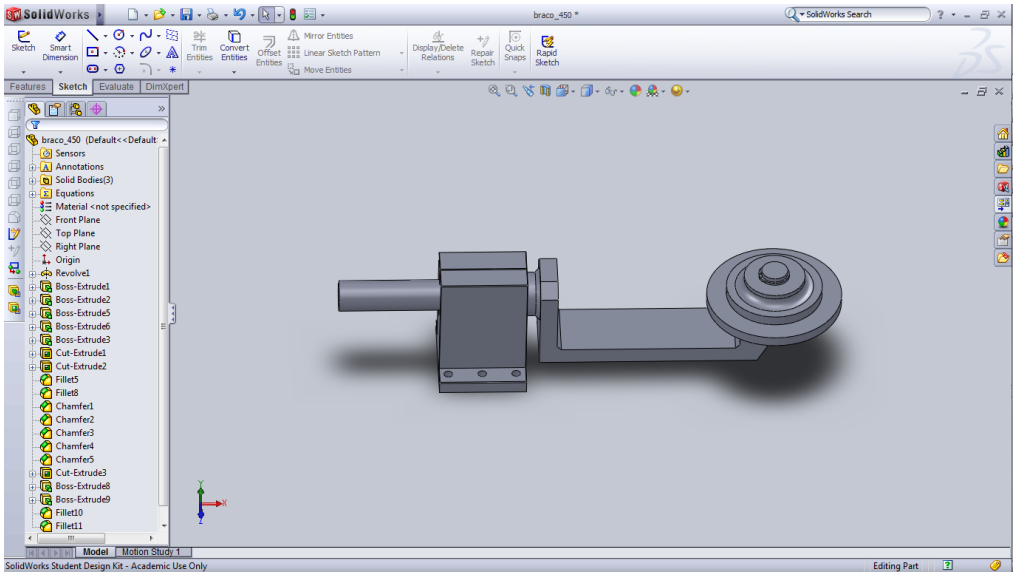

Figura 20 Desenho do braço para a roda de 450 mm

Após o desenho do braço, desenhou-se a respectiva roda, tal como se pode ver na Figura 21.

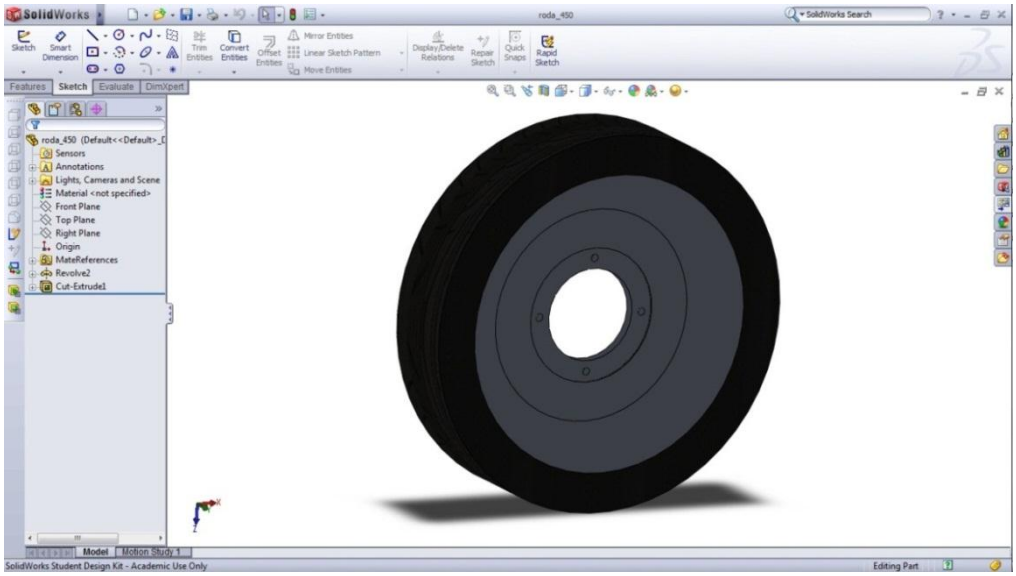

Figura 21 Desenho da roda de 450 mm

Após a obtenção do desenho de ambos os componentes, passou-se à montagem do conjunto, ou seja, criou-se um ficheiro *assembly* onde se posicionaram correctamente as duas peças. Tal como se pode ver na Figura 22, as peças foram acopladas utilizando a funcionalidade "`Mates`", evidenciada na figura.

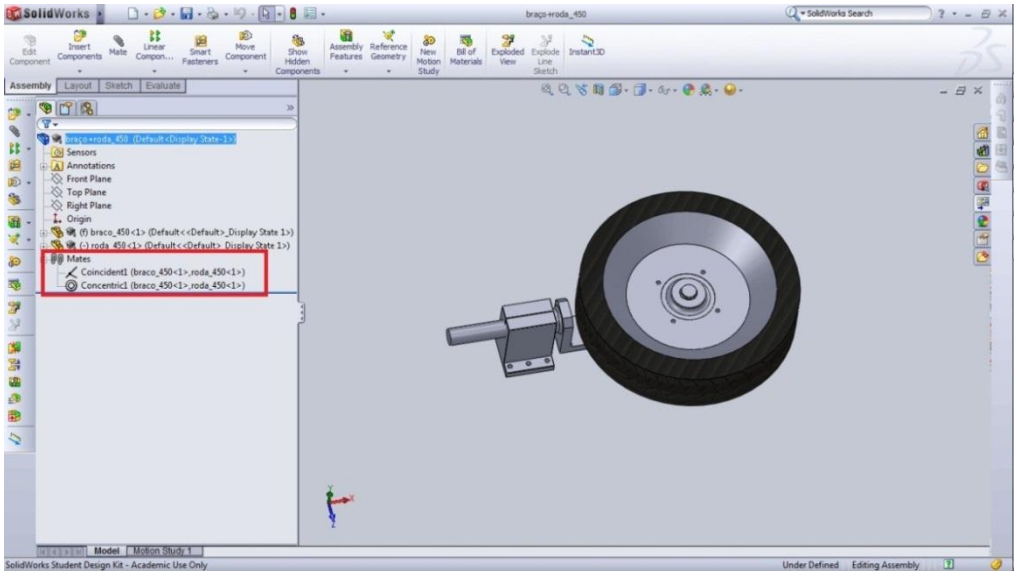

Figura 22 *Assembly* do conjunto braço+roda de 450 mm

Tendo efectuado os desenhos e o *assembly* do braço e roda de 450 mm, efectuaram-se os mesmos passos para o braço e roda de 150 mm, como se pode ver na Figura 23. Este processo representa apenas uma parte do modelo de toda a célula.

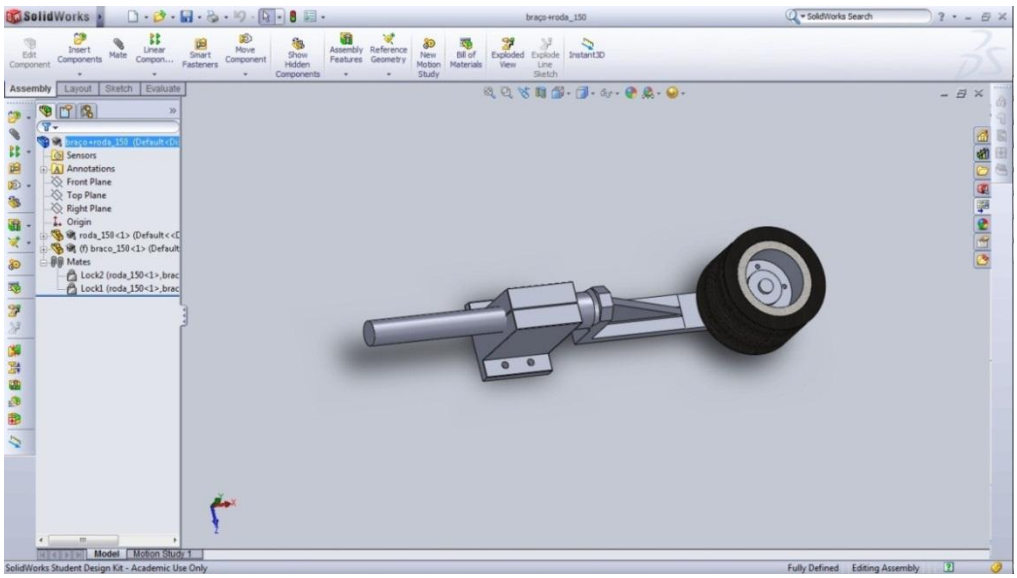

Figura 23 *Assembly* do conjunto braço+roda de 150 mm

Como se pode ver na Figura 24, as unidades foram desenhadas de forma "rudimentar" e pouco exacta no que toca a pormenores, pois apenas é necessário contemplar alguns pormenores existentes na mesma (não foram consideradas furações e cablagem por não se considerar relevante para a modelação da célula).

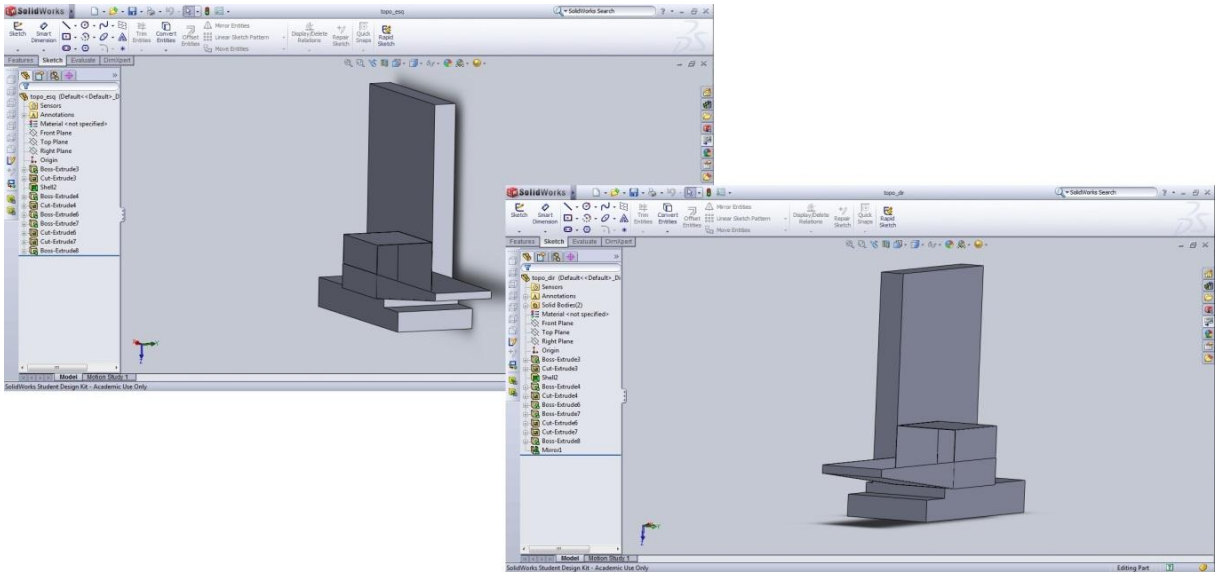

Figura 24 Parte superior das unidades de lixagem

Após a obtenção das unidades restava apenas criar os ficheiros *assembly* que vão representar de forma aproximada cada uma das unidades. Para se executar este passo recorreu-se aos *assemblys* criados anteriormente para cada um dos braços e efectuou-se a montagem de cada uma das unidades. No canto superior esquerdo da Figura 25 encontra-se representada a unidade um e no canto superior direito a unidade quatro; na parte inferior da imagem vê-se a unidade dois do lado esquerdo e a unidade três do lado direito.

O posicionamento correcto das diferentes unidades apenas será efectuado aquando da criação do modelo final utilizando o RobotStudio, uma vez que estas unidades aqui desenhadas irão corresponder a mecanismos no modelo final.

Com todas as unidades criadas, converteram-se os ficheiros para extensões passíveis de serem interpretadas pelo RobotStudio; neste caso utilizou-se a extensão ".sat", sendo esta relativa a ficheiros Allen, Charles, Ian's System (ACIS).

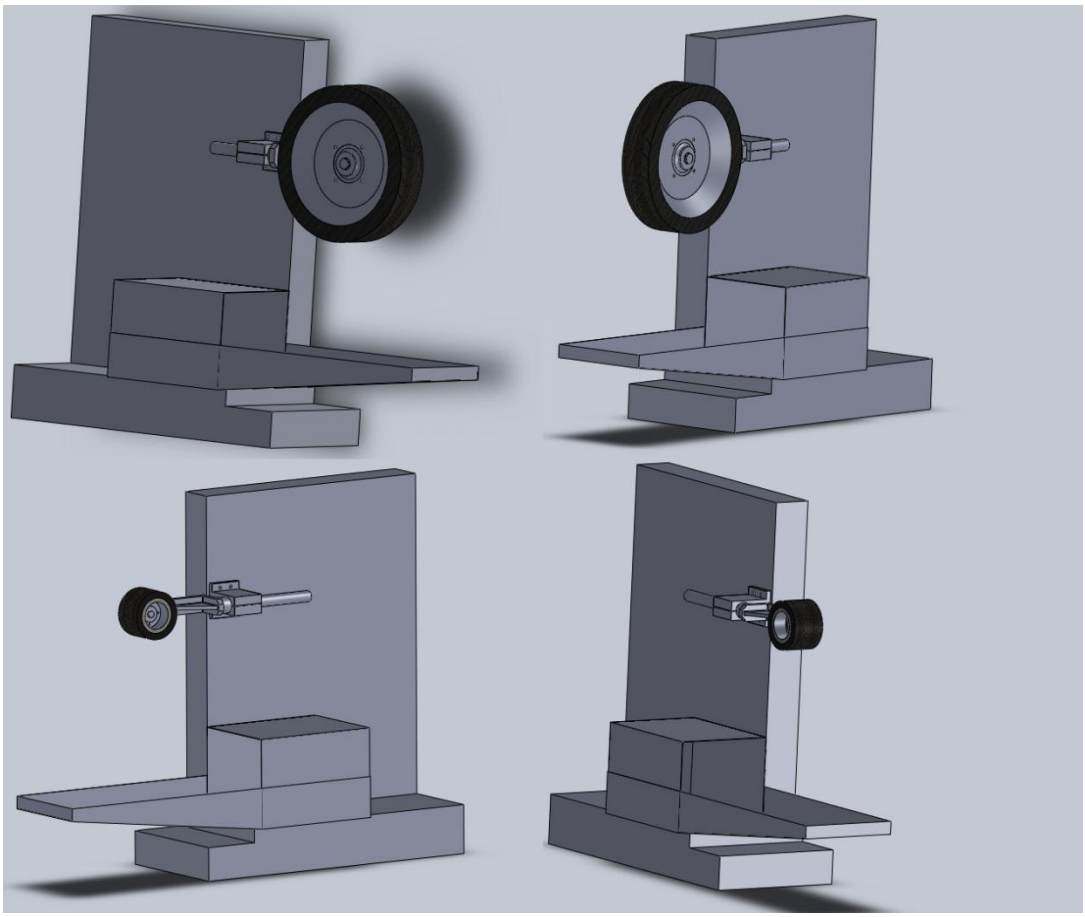

Figura 25 Unidades de lixagem da célula do robô

É de salientar que apenas foram desenhadas duas das unidades (neste caso as unidades um e três), pois como estas se encontram agrupadas duas a duas no que toca à sua configuração, optou-se por espelhar as duas unidades desenhadas para facilitar e acelerar o processo de modelação, mantendo assim constantes todas as características de cada unidade. É ainda de notar que para o desenho de cada unidade se optou por apenas desenhar a sua parte superior, visto que a parte inferior é fixa e pode ser facilmente desenhada no RobotStudio aquando da criação do modelo final da célula, tal como será explicado posteriormente.

4.3. IMPORTAÇÃO E CONFIGURAÇÃO DOS MECANISMOS NO ROBOTSTUDIO

Após a criação de todos os componentes descritos no ponto anterior, passou-se à configuração de cada um dos mecanismos necessários. Neste ponto, utilizam-se os ficheiros ACIS criados para cada um dos componentes e, recorrendo ao RoboStudio, configuram-se os mecanismos e acaba-se a modelação 3D de cada unidade. Com a

configuração dos mecanismos pretende-se que cada unidade efectue as diferentes acções representativas da realidade aquando da realização do programa, para a obtenção de uma simulação mais aproximada da realidade.

Para configurar os mecanismos existia a necessidade de completar a modelação 3D. Para isso foi utilizada a funcionalidade de criação de objectos fornecida pelo RobotStudio e, tal como se pode ver na Figura 26, foi criada a base da unidade com as medidas exactas, tal como era pretendido.

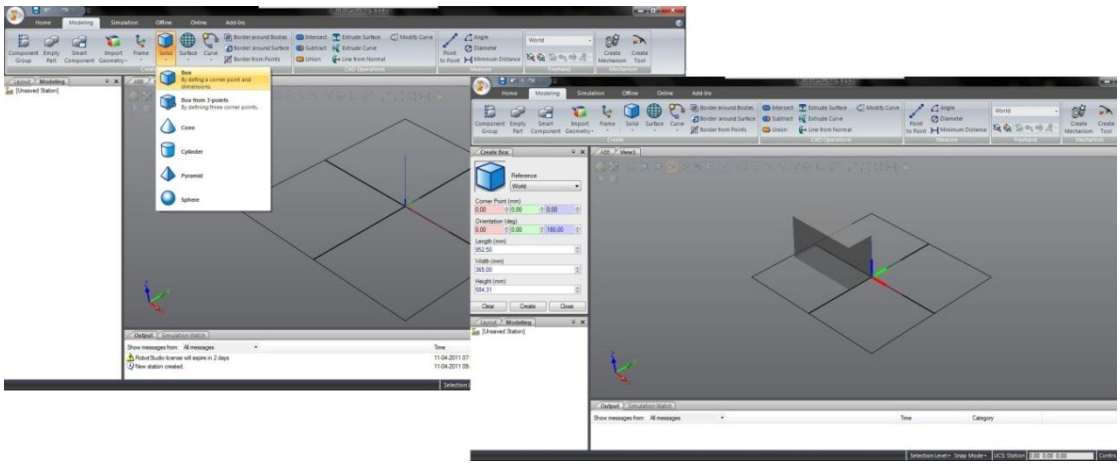

Figura 26 Criação da parte inferior das unidades de lixagem

Após a criação da parte inferior da unidade passou-se à construção da totalidade da unidade. Para isso foi necessário importar a parte superior previamente modelada no SolidWorks.

Para que isso fosse efectuado, e como se pode ver na Figura 27, utilizou-se a funcionalidade "`Import Geometry`" do RobotStudio, obtendo-se assim na área de trabalho as duas partes necessárias para a criação do mecanismo.

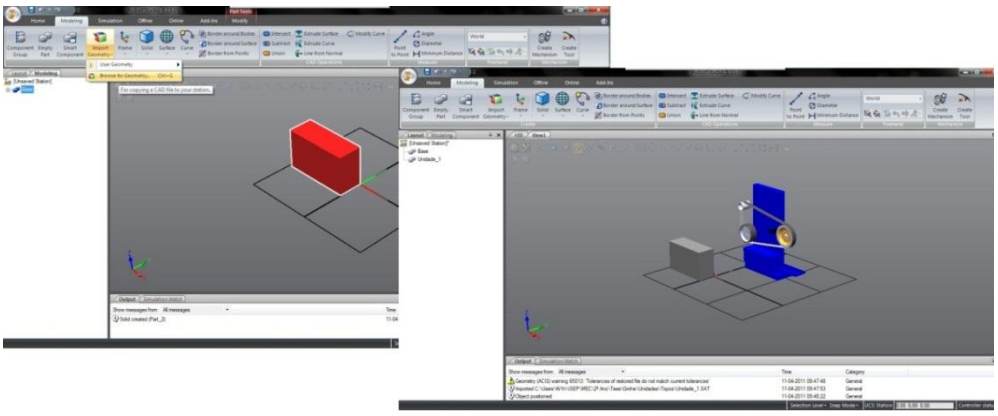

Figura 27 Importação da parte superior da unidade de lixagem para o ABB RobotStudio

Com a importação efectuada decidiu-se alterar a origem da parte superior para que o seu posicionamento fosse facilitado e permitisse mais precisão (base e parte superior devem estar correctamente alinhadas para facilitar o posicionamento na célula). Todo o processo de alteração da origem da geometria e posterior posicionamento pode ser visto na Figura 28.

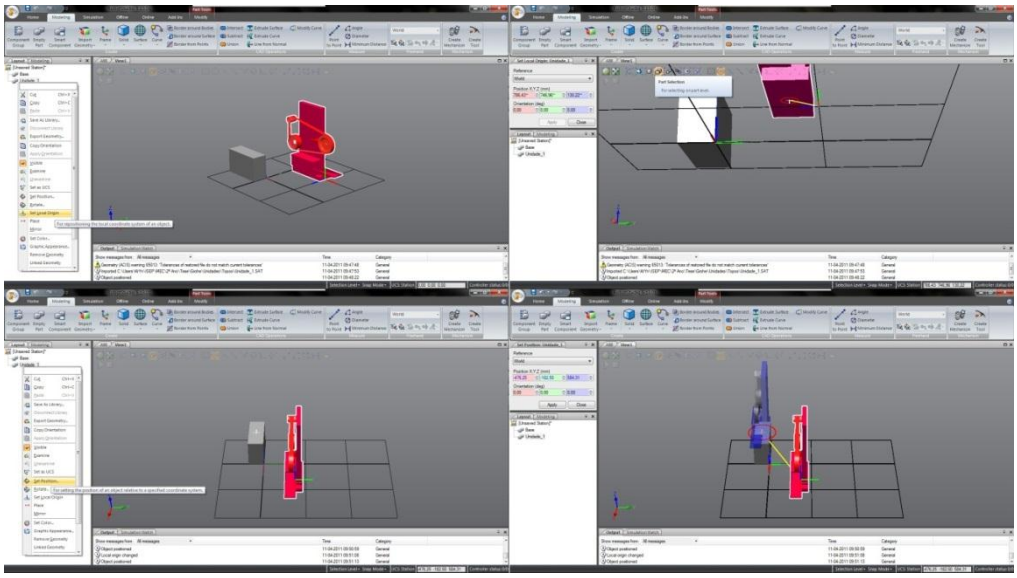

Figura 28 Posicionamento da parte superior da unidade de lixagem

Com a posição definida, e apenas para melhorar o realismo da simulação, alterou-se a cor da parte inferior da unidade para se assemelhar à existente na realidade. Tal como se pode ver na Figura 29, para alterar a cor de uma geometria basta utilizar a funcionalidade "`Graphic Appearance`".

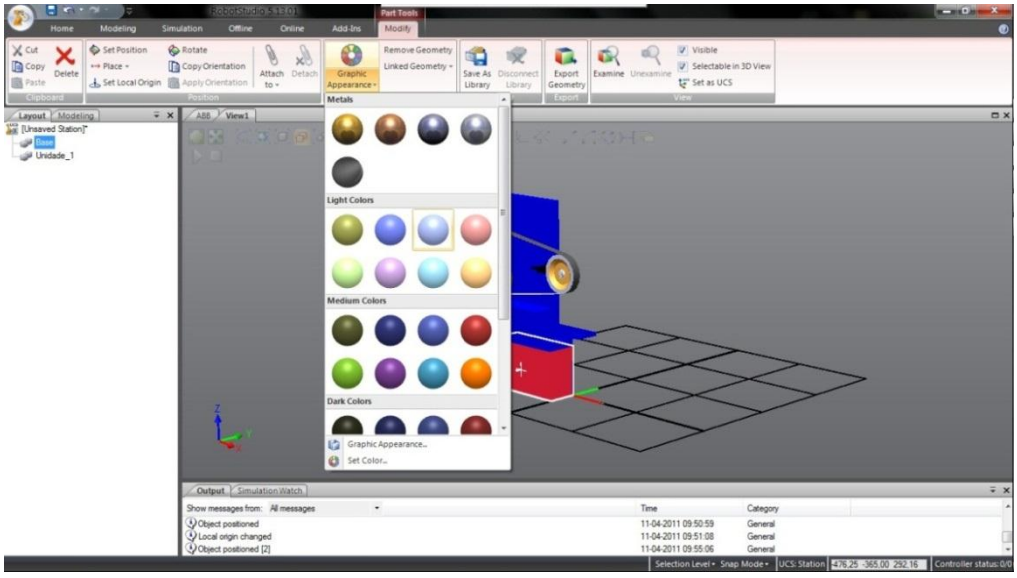

Figura 29 Melhoramento do aspecto gráfico de uma geometria/mecanismo

Terminado o processo de posicionamento e melhoramento gráfico da unidade, passou-se à criação e configuração do mecanismo.

Todos os passos para a conclusão deste processo estão representados na Figura 30 e Figura 31. No canto superior esquerdo pode-se ver o ambiente gráfico na fase inicial do processo, posteriormente, no canto superior direito e inferior esquerdo encontra-se a atribuição de cada uma das peças a um "`link`". É nesta fase que se identificam quais as partes que se irão mexer e qual será a base desse movimento. Por fim tem-se no canto inferior direito a configuração do movimento propriamente dito, definindo distância, orientações e limites.

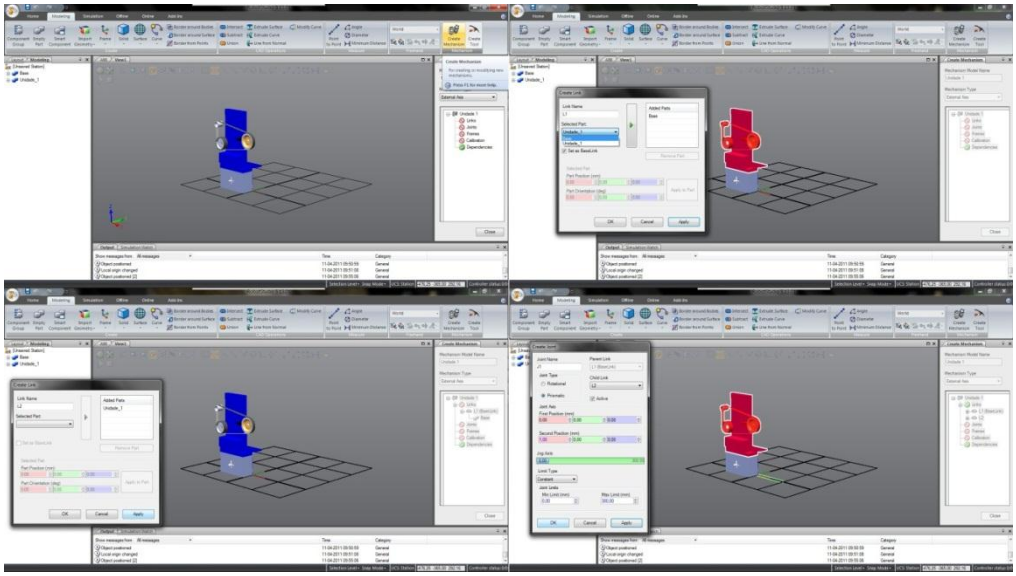

Figura 30 Criação e configuração do mecanismo correspondente à unidade de lixagem

No canto inferior esquerdo da Figura 31 está representado o mecanismo no ponto que antecede a sua compilação e no canto inferior direito é demonstrado o mecanismo já compilado.

Após a criação do mecanismo foram criadas duas posições pré-definidas e existentes no sistema real (ver Figura 32). O sistema real contém a posição "`UNIDAD_1_DELANTE`", que representa a unidade activada e deslocada para a frente, e a posição "`UNIDAD_1_DETRAS`", que representa a unidade em repouso e recuada.

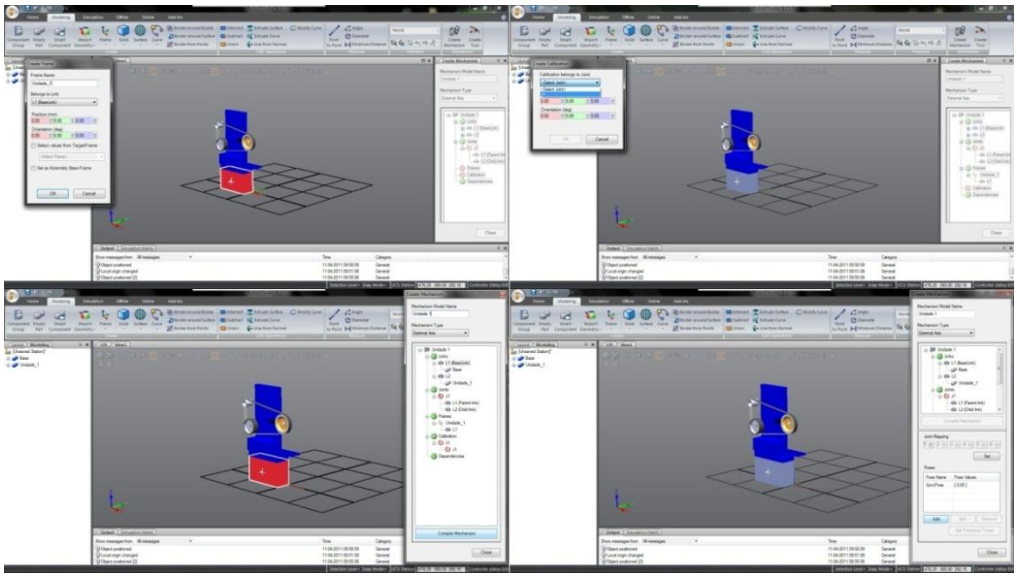

Figura 31 Continuação da configuração do mecanismo correspondente à unidade de lixagem

Após a criação destas posições, e para ser concluída a criação do mecanismo, apenas é necessário fechar a funcionalidade "`Create Mechanism`", como pode ser visto na Figura 32.

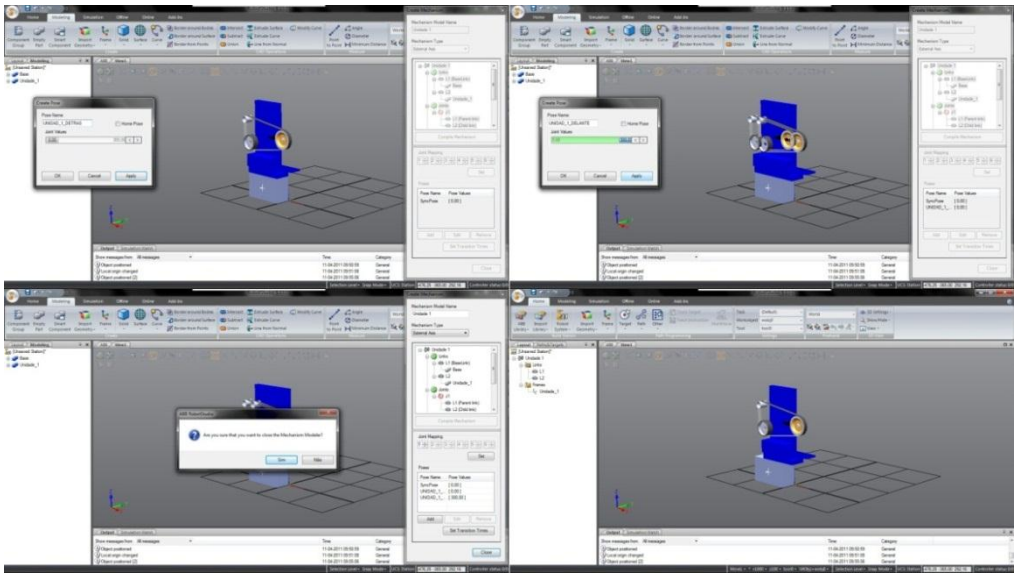

Figura 32 Definição das posições predefinidas do mecanismo correspondente à unidade de lixagem

Por fim, e como se pode ver na Figura 33, todo o mecanismo é gravado como biblioteca para posterior utilização aquando da realização dos respectivos programas.

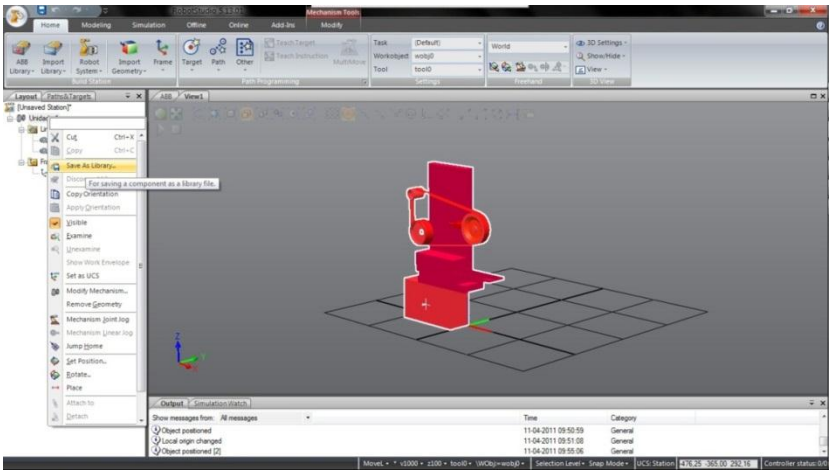

Figura 33 Conclusão da criação do mecanismo e gravação como biblioteca

4.4. REALIZAÇÃO DA CALIBRAÇÃO DA CÉLULA SIMULADA

Para a correcta definição da modelação 3D foi necessário a definição de novos *workobjects* referentes a cada uma das unidades, uma vez que devido à configuração actual das unidades de trabalho ser diferente da original os *workobjects* existentes não coincidem com o centro das rodas. Isto deve-se ao facto de quando as células foram adquiridas pela Grohe a sua configuração ser diferente, sendo posteriormente alterada para a configuração actual com o objectivo de melhorar o aproveitamento da célula e facilitar o correcto lixamento das torneiras.

Para a definição dos *workobjects*, existia a necessidade de desenvolver um mecanismo (ou uma aplicação) que fosse capaz de garantir essa concentricidade (manter os *workobjects* no centro das rodas). Para o desenvolvimento dessa ferramenta foi utilizado mais uma vez o SolidWorks. Foram desenhadas nesta aplicação algumas peças que, quando acopladas, formam um dispositivo capaz de garantir a concentricidade dos *workobjects* face às rodas. O mesmo dispositivo pode ser utilizado como forma de confirmação de possíveis desvios no caso de existir alguma colisão ou mudança de algum braço.

Tal como se pode ver na Figura 34, foram desenvolvidas três peças, sendo que a ferramenta necessita de três pinos (imagem do canto inferior direito) para estar completa.

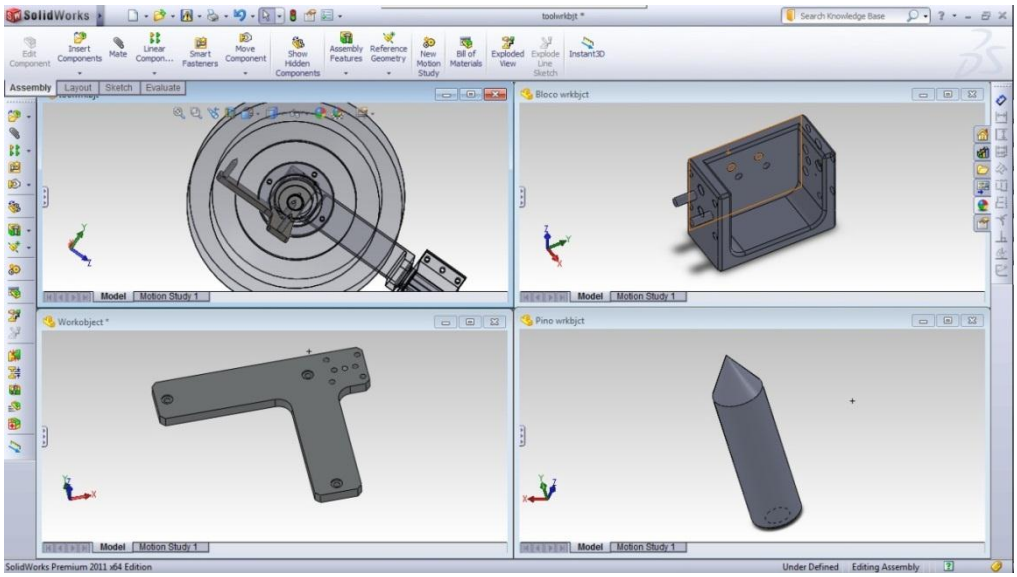

Figura 34 Ferramenta para definição dos *workobjects* das unidades

Pode ver-se na Figura 35 a aparência da ferramenta para a definição dos *workobjects* quando se encontra totalmente montada. Após a montagem é necessário acoplar o conjunto ao braço de cada uma das unidades e, posteriormente, configurar ou confirmar o posicionamento dos *workobjects*.

A definição dos *workobjects* será explicada no ponto seguinte, relativo ao desenvolvimento dos programas dos robôs.

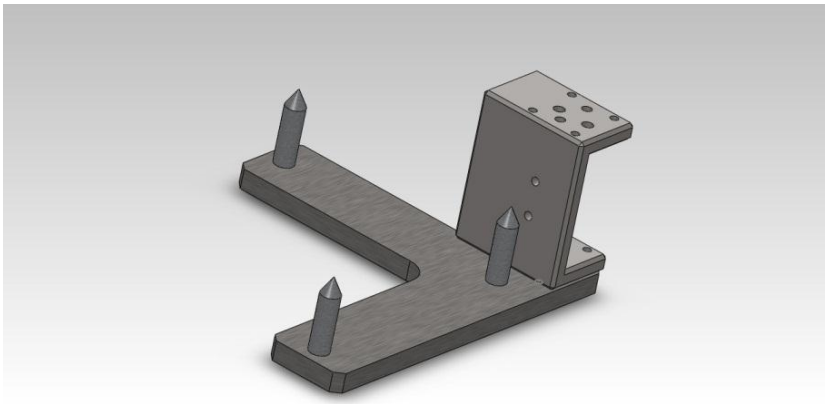

Figura 35 Ferramenta (*tool*) para definição dos *workobjects*

4.5. TESTES REALIZADOS COM PROGRAMAS DESENVOLVIDOS *OFFLINE*

Estando concluída a modelação 3D das unidades, e após serem inseridas no modelo virtual, foram efectuados alguns testes para validar a viabilidade da programação *offline* aquando da transferência do programa para a célula real. Como tal são apresentadas algumas

imagens das peças obtidas após a realização do primeiro teste e programação *offline*, mas é de salientar que não foi desenvolvida a totalidade do programa, porque apenas se pretendia verificar se o posicionamento das unidades na célula virtual estava correcto. Em cada uma das figuras seguintes é apresentado em primeiro lugar a peça após ser maquinada, em seguida a peça obtida com o programa desenvolvido *offline* e, por último, o resultado pretendido (peça lixada com o programa desenvolvido *online* no robô). A Figura 36 representa uma vista superior da peça, sendo que neste caso esta rotina não foi efectuada *offline* como se pode reparar pela inexistência de marcas de lixa.

Figura 36 Parte superior da torneira após a realização do primeiro teste de programação *offline*

Na Figura 37 pode ver-se uma fotografia da parte de trás da peça e, tal como se pode verificar, o resultado obtido foi satisfatório na lixagem da parte posterior da peça. No entanto, como é visível na Figura 38, existem alguns problemas na concentricidade da peça, sendo que este teste foi realizado antes de se adicionar a modelação das lixas às unidades.

Na Figura 38 pode visualizar-se a parte inferior da peça. Através da análise das imagens pode verificar-se que o resultado obtido na lixagem desta parte da peça não é satisfatório, mas é preciso ter em conta que a parte do meio da peça não foi efectuada *offline*. Isto deveu-se ao facto de ainda não existir a modelação da roda necessária à realização da peça, uma vez que esta peça é realizada com recurso a uma roda de 450 mm de diâmetro e 50 mm de largura na unidade 2 em vez da normalmente utilizada que tem apenas 100 mm de diâmetro e largura.

Figura 37 Parte de trás da torneira após a realização do primeiro teste de programação *offline*

Figura 38 Parte de baixo/frente da torneira após a realização do primeiro teste de programação *offline*

A Figura 39 apresenta também uma vista superior da peça mas com especial enfâse no corpo desta. Como se pode ver (e foi referido anteriormente) existem problemas de concentricidade no corpo da peça. Este problema deveu-se à falta das lixas no modelo das unidades. Verificou-se que aquando da programação *offline* os pontos foram posicionados no centro da roda e, como a lixa tem uma ligeira inclinação do centro da roda, quando se lixou a peça, esta forçou mais a parte inferior nessa inclinação, obtendo-se assim esta

deformação da circunferência (foi esta avaliação que levou à adição da modelação das lixas à célula virtual).

Figura 39 Parte de baixo da torneira após a realização do primeiro teste de programação *offline*

Na Figura 40 e Figura 41 pode-se ver a peça lateralmente, sendo que neste caso o resultado do facejamento da parte superior da peça foi muito satisfatório obtendo-se uma linha perfeitamente definida, mas a parte inferior teve pouco ou nenhum contacto com a lixa o que levou à conclusão que as unidades não estariam bem definidas no espaço.

Figura 40 Lateral esquerda da torneira após a realização do primeiro teste de programação *offline*

Figura 41 Lateral direita da torneira após a realização do primeiro teste de programação *offline*

É de notar que a realização destes testes não incluíram nenhuma afinação manual do programa do robô, sendo que as únicas alterações efectuadas recorrendo ao robô foram a nível de movimentações entre rotinas para evitar colisões e a peça apenas foi lixada recorrendo a uma unidade de lixa grossa (neste caso utilizou-se a unidade um).

Graças a estas conclusões avançou-se então para o desenvolvimento de uma ferramenta para a definição dos *workobjects* das unidades como foi demonstrado no ponto 4.4.

Após efectuados os testes com a configuração anteriormente descrita denotou-se a existência de algumas falhas e a necessidade da representação de mais alguns componentes. Como tal, voltou-se a avaliar quais os componentes necessários e decidiu-se representar o motor, o esticador e a lixa. Apenas a lixa seria importante mas para uma melhor percepção da célula optou-se por desenhar os três componentes, como se mostra na Figura 42. Foram também adicionadas algumas definições de aparência para melhorar o realismo gráfico da respectiva unidade.

Mais uma vez, após efectuada a modelação e respectiva aplicação das unidades na célula virtual efectuou-se mais um teste.

Este teste foi realizado recorrendo a um programa totalmente realizado *offline* e, na realidade, foram realizados dois testes com a mesma peça sendo um efectuado sem afinação e o outro com afinação. Os resultados destes dois testes podem ser vistos nas Figura 43, Figura 44, Figura 45, Figura 46, Figura 47.

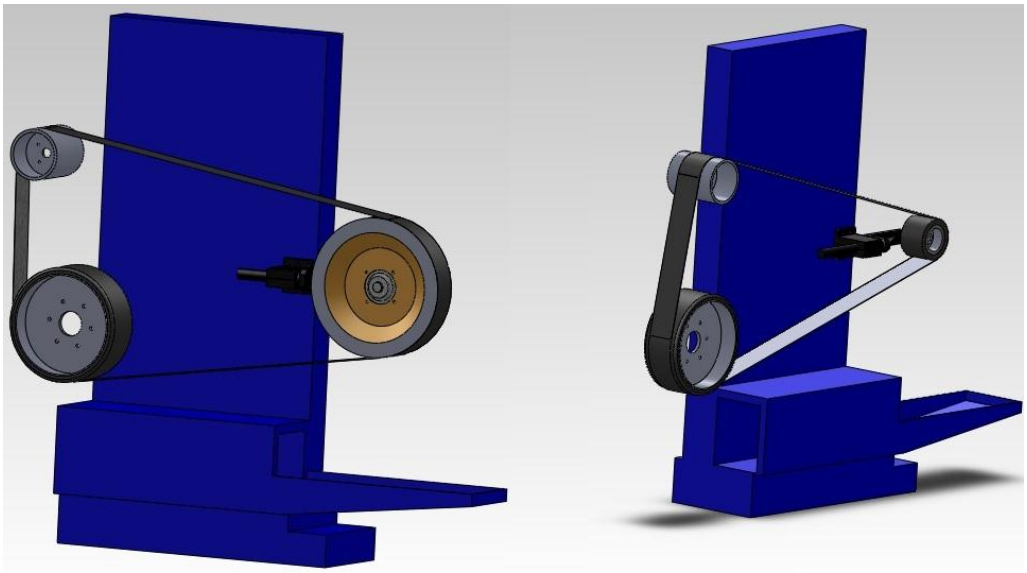

Figura 42 Unidades 1 e 3 com a representação da lixa, motor e esticador

Na imagem à esquerda pode-se ver a peça após ser maquinada e pronta para ser lixada; no centro encontra-se a peça lixada com o programa original sem qualquer tipo de afinações. Nesta imagem pode-se ver que nem toda a torneira foi lixada e que o acabamento do lixamento na parte lateral não é o melhor, para além de se notar uma ligeira deformação na parte de trás da torneira. Os factores associados a estes problemas são facilmente explicados, pois o problema reside na pressão efectuada sobre a roda quando se lixa a peça. Tem-se em contacto duas superfícies redondas (roda e torneira) e para se efectuar o contacto a torneira deve estar paralela (ou quase paralela) à roda e o movimento deve ser todo efectuado com a mesma pressão e isso não aconteceu neste caso (daí a deformação). Por último, a imagem da direita demonstra uma peça lixada já com o programa totalmente afinado e pronto para começar a produzir.

Na Figura 44 à esquerda pode ver-se a peça após ser maquinada e pronta para ser lixada e no centro a peça lixada com o programa original sem qualquer tipo de afinações. Mais uma vez, e tal como explicado na imagem anterior, esta lateral apresenta os mesmos problemas apesar de estar ligeiramente melhor na parte da bica. Por fim, a imagem da direita mostra uma peça lixada já com o programa totalmente afinado e pronto para começar a produzir.

Figura 43 Lateral direita da torneira utilizada no segundo teste de programação *offline*

Figura 44 Lateral esquerda da torneira utilizada no segundo teste de programação *offline*

Na Figura 45 à esquerda pode ver-se a peça após ser maquinada e pronta para ser lixada e no centro a peça lixada com o programa original sem qualquer tipo de afinações. Nesta parte da torneira o lixamento foi aceitável, existindo apenas a necessidade de aumentar ligeiramente a amplitude dos movimentos para se conseguir chegar à parte inferior da bica que, como se vê na imagem central, não está lixada. Por fim a imagem da direita apresenta uma peça lixada já com o programa totalmente afinado e pronto para começar a produzir.

Figura 45 Parte de baixo da torneira utilizada no segundo teste de programação *offline*

Na Figura 46 pode ver-se à esquerda a peça após ser maquinada e pronta para ser lixada e ao centro a peça lixada com o programa original sem qualquer tipo de afinações. Como se pode ver o lixamento obtido está aceitável, sendo que com o problema proveniente da lixa grossa que deformou a peça no final do processo de lixamento da peça o acabamento não é bom. Por fim a imagem da direita mostra uma peça lixada já com o programa totalmente afinado e pronto para começar a produzir.

Figura 46 Parte de trás da torneira utilizada no segundo teste de programação *offline*

Na Figura 47 pode ver-se à esquerda a peça após ser maquinada e pronta para ser lixada e ao centro a peça lixada com o programa original sem qualquer tipo de afinações. Esta rotina do programa foi a melhor conseguida através da programação *offline*, pois como se pode ver (e comparando a imagem central com a imagem do lado direito) o lixamento está muito aproximado, faltando apenas reorientar ligeiramente os pontos mais próximos da bica, quer para retirar a colisão (com o núcleo de alumínio da roda) que causou os dois riscos visíveis na imagem central, quer para lixar melhor a ligação da bica com o corpo. A imagem da direita ilustra uma peça lixada já com o programa totalmente afinado e pronto para começar a produzir.

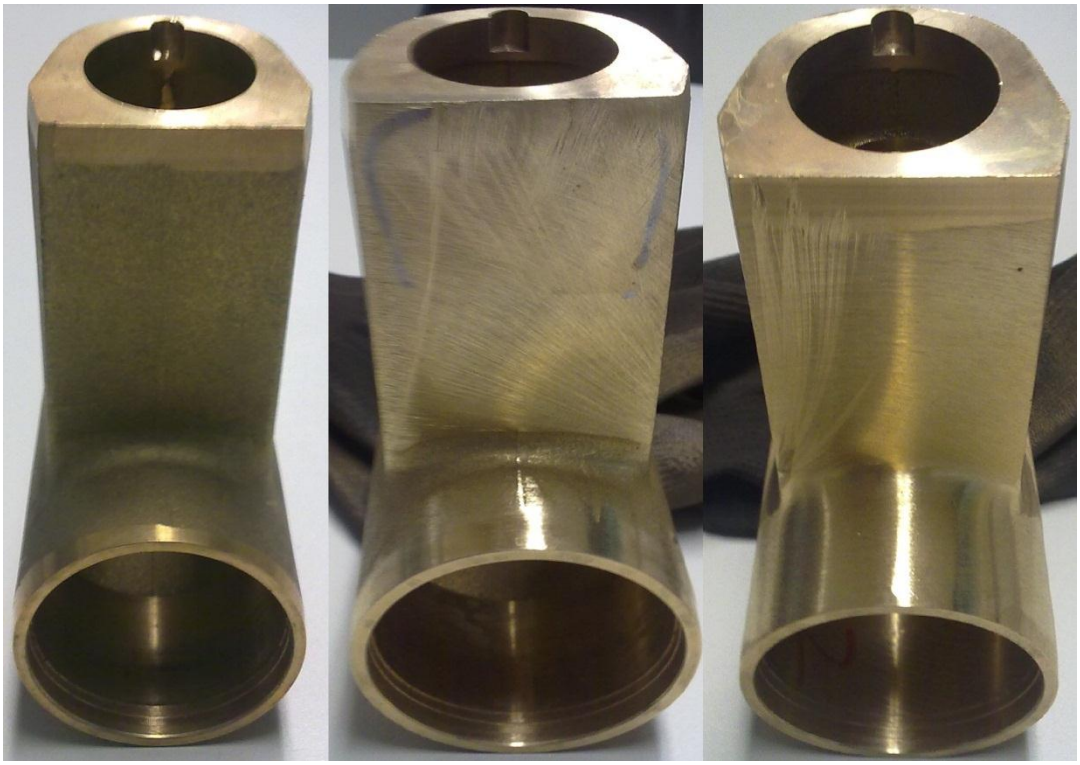

Figura 47 Parte de cima da torneira utilizada no segundo teste de programação *offline*

Como foi possível visualizar pelas imagens anteriores a realização deste segundo teste foi muito mais satisfatório que o primeiro. Com todos os ajustes efectuados ao longo do desenvolvimento dos programas, conseguiu-se obter uma boa precisão no que toca à modelação 3D da célula e, como consequência, obteve-se um programa mais fiável e próximo do pretendido.

5. DESENVOLVIMENTO DOS PROGRAMAS DOS ROBÔS

Neste capítulo irão ser explicados os métodos de programação utilizados, bem como alguns pormenores tidos em conta para uma melhor representação gráfica e simulação do funcionamento das unidades.

5.1. CONFIGURAÇÃO DE ENTRADAS E SAÍDAS

Resolvidos todos os problemas com a correcta modelação 3D da célula de trabalho, começou-se então a realização dos primeiros programas para o lixamento das peças.

Como se pode ver na Figura 48, foram também adicionados dois painéis, para além das quatro unidades, para ser possível a detecção da ocorrência de possíveis colisões com as paredes das células aquando da execução dos diferentes movimentos.

Para que seja possível descarregar um programa para o robô existe a necessidade de se ter no RobotStudio todas as configurações existentes no robô, isto é, cada robô e respectivo controlador têm entradas, sinais de comunicação e variáveis próprias (neste caso existe uma uniformização destas configurações em todos os robôs da ABB existentes no departamento de Lixamento/Polimento). Desta forma, para se poder começar a

programação, foi necessário carregar essas configurações para a célula de trabalho simulada no ABB RobotStudio.

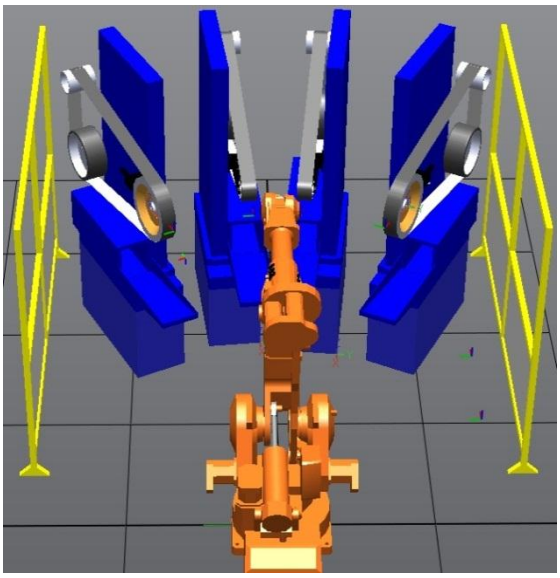

Figura 48 Modelo da célula de trabalho

Estas configurações podem ser carregadas de duas maneiras diferentes:

- Recorrendo a um *backup* previamente feito, ou seja fazer um *backup* da célula de trabalho e carregar esse *backup* para o ABB RoboStudio;
- Ou utilizando os ficheiros de parâmetros existentes em cada robô; neste caso copiam-se os ficheiros de configuração do robô ("`EIO.cfg`", "`MMC.cfg`","`MOC.cfg`","`SIO.cfg`" e "`SYS.cfg`") e carregam-se no ABB RobotStudio.

Após a carga destes parâmetros pode-se então começar a programação. Deve ter-se em atenção os módulos já existentes que são transversais a todos os robôs existentes no departamento (no caso dos robôs são, na generalidade, os módulos "`USER`" e "`BASE`"; existe ainda o módulo "`MEPSA`"/"`BULA`", que muda de nome dependendo do fabricante da célula mas em que a base é a mesma).

Estes módulos normalmente permanecem sempre constantes, uma vez que a política existente na empresa para as mudanças de programa assenta na alteração do módulo de programa e não na criação de *backups*. Assim, em vez de se criar um *backup* para cada programa existente apenas se guarda o módulo do respectivo programa. Deste modo consegue-se poupar tempo nas mudanças de produtos/programas, pois o tempo de carga de

um módulo é muito inferior ao tempo de carregamento de um *backup* e com esta filosofia é também possível carregar o mesmo programa em robôs diferentes sem interferir com as calibrações e configurações de cada robô.

Tal como se pode ver na Figura 49, os módulos que estão marcados foram adicionados para efectuar as configurações (configurações de variáveis e rotinas pré-definidas) e tornar possível a programação *offline* com capacidade de posteriormente levar o programa para a célula real.

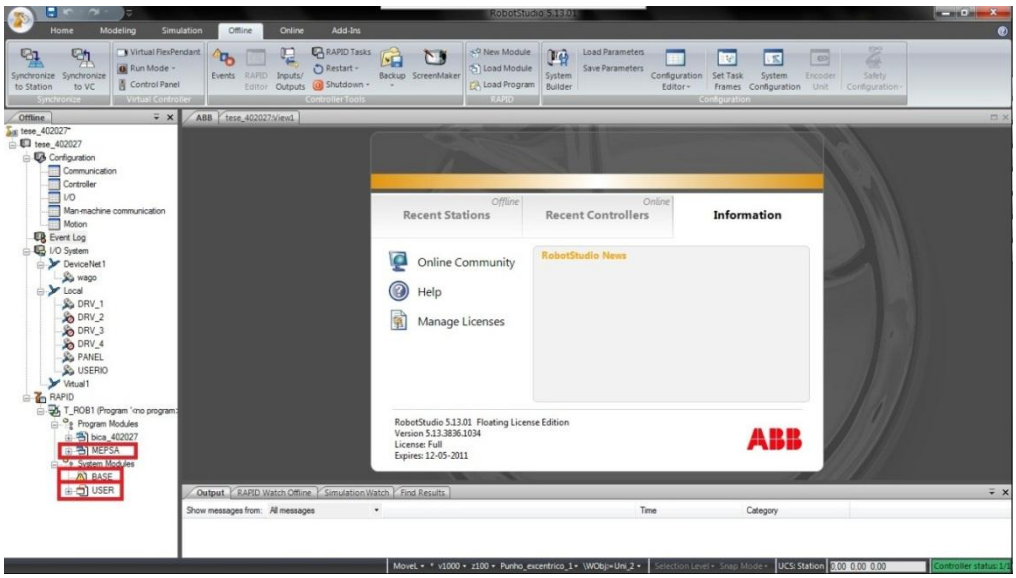

Figura 49 Módulos de configuração dos robôs de lixagem

Após o término das configurações, e tendo os módulos carregados, pode então passar-se à realização do programa para efectuar o lixamento da peça. Neste momento estão reunidas todas as condições para efectuar uma correcta programação tendo em consideração todas as entradas e variáveis existentes e definidas nos robôs.

5.2. DEFINIÇÃO DOS EVENTOS PARA EFEITOS DE SIMULAÇÃO DO PROGRAMA

Para melhorar a percepção dos movimentos do robô e das unidades ao longo do desenvolvimento do programa foram adicionados eventos para simular a movimentação das unidades.

Para se adicionar esses eventos devem-se ter previamente definidos todos os mecanismos ou funções de forma a facilitar o processo de configuração do evento. Tal como se pode

ver na Figura 50, para adicionar eventos é necessário abrir a *tab* "Simulation", seleccionar o menu "Simulation Logic" e posteriormente "Event Manager".

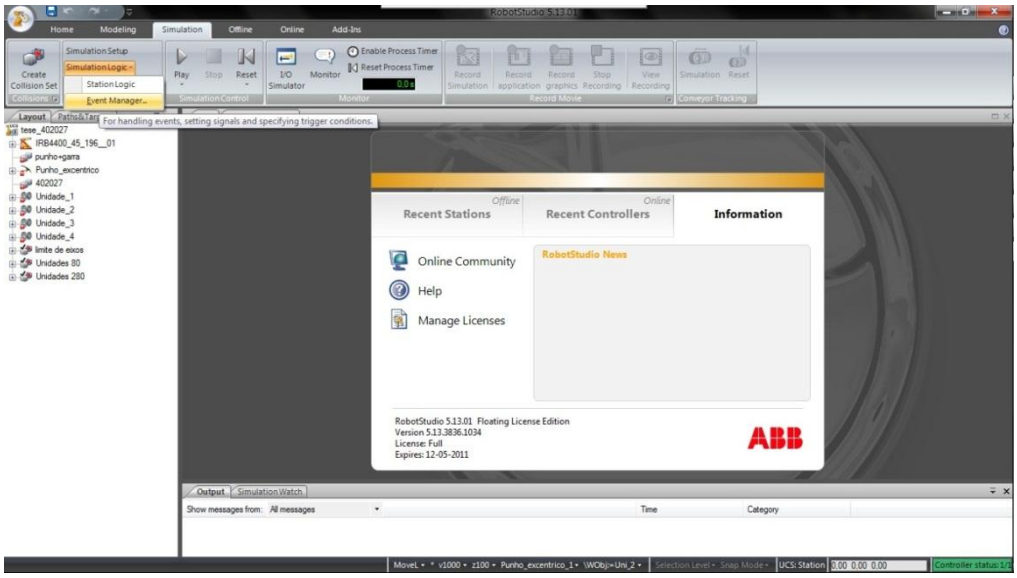

Figura 50 Event Manager

Após a selecção desse menu irá aparecer a janela apresentada no lado esquerdo da Figura 51. Nesta janela, utilizando o botão "ADD", podem acrescentar-se todos os eventos desejados. Neste caso apenas foram acrescentados eventos para o avanço e recuo das unidades, bem como para o avanço e recuo das paletes, aquando da simulação do programa. Na Figura 51 (do lado direito) pode ver-se ainda a configuração inicial para a criação dos eventos referidos anteriormente (o tipo de evento é igual no avanço e recuo da unidades).

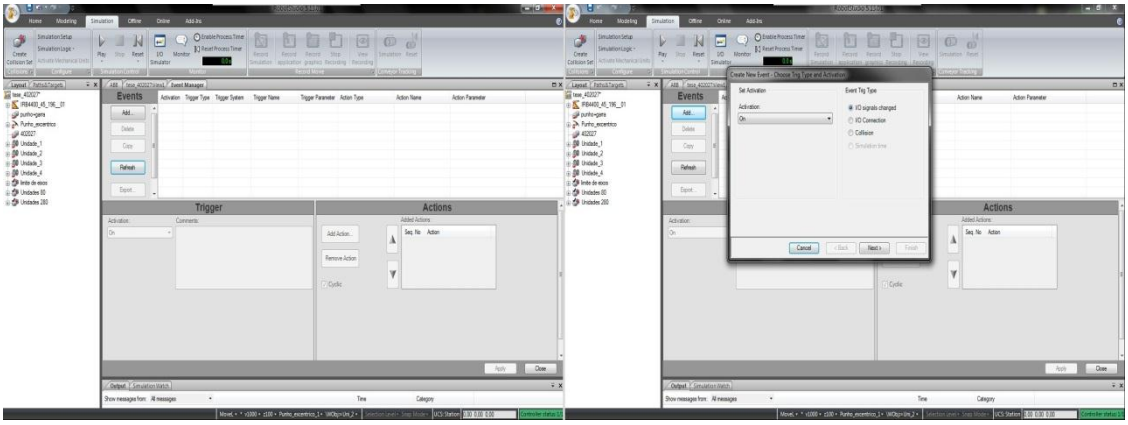

Figura 51 Janela do "Event Manager" e configuração do tipo de evento

Efectuado este passo começa-se então a configuração específica de cada uma das unidades e dos seus movimentos. Para tal é necessário saber quais as variáveis que irão influenciar o

evento e como será a resposta quando o evento é activado. Assim, ao longo da configuração do evento ter-se-á de configurar qual o sinal que activa o evento, qual será o evento, e que mecanismo será activado (neste caso especifico trata-se de um mecanismo) e como será activado. A título de exemplo são apresentadas duas imagens com os diferentes passos para duas configurações diferentes dos mecanismos. Na Figura 52 pode ver-se como é configurado o avanço das unidades e na Figura 53 como é efectuada a configuração do recuo das unidades.

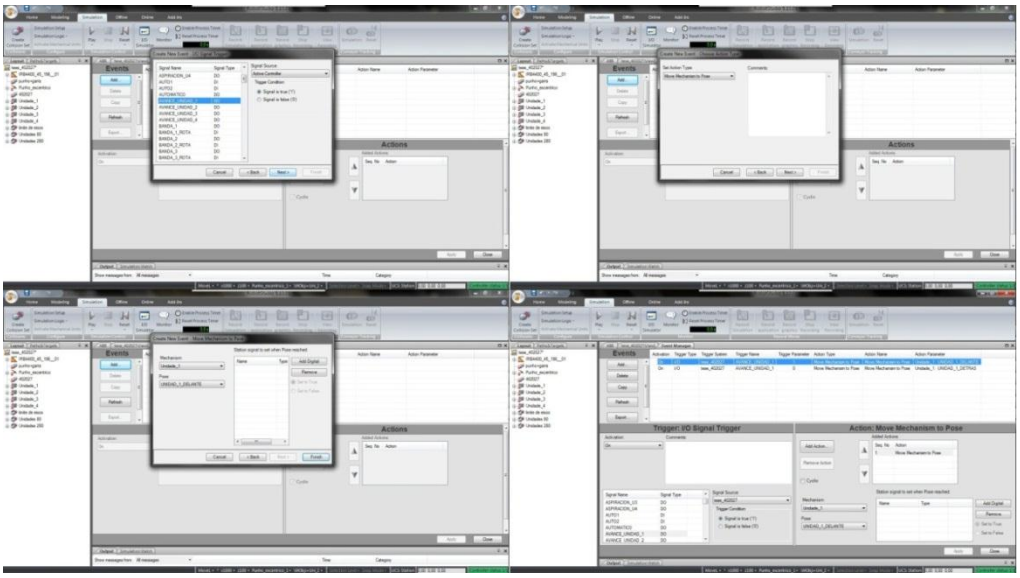

Figura 52 Configuração do evento para avanço de unidades

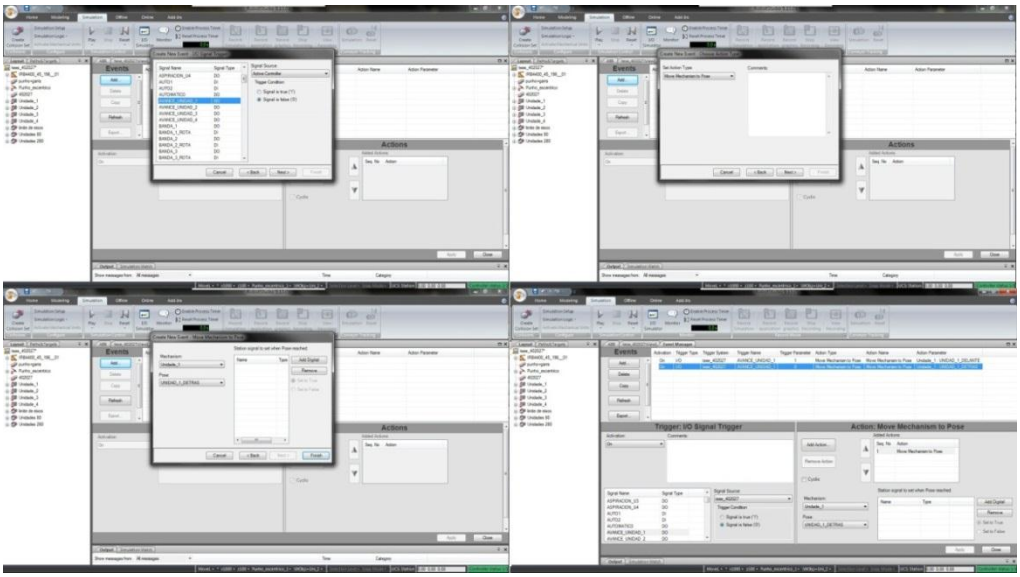

Figura 53 Configuração do evento para recuo de unidades

É de notar que, tal como foi referido anteriormente, estas imagens apenas servem para exemplificar os diferentes passos para a configuração de cada unidade. As imagens

representam apenas a configuração de uma unidade e têm de ser repetidos todos os passos descritos para todas as unidades, pois cada unidade responde a um sinal diferente. A Figura 54 é exemplificativa do número de eventos necessários e suas configurações, aquando do término das configurações de eventos para todas as unidades.

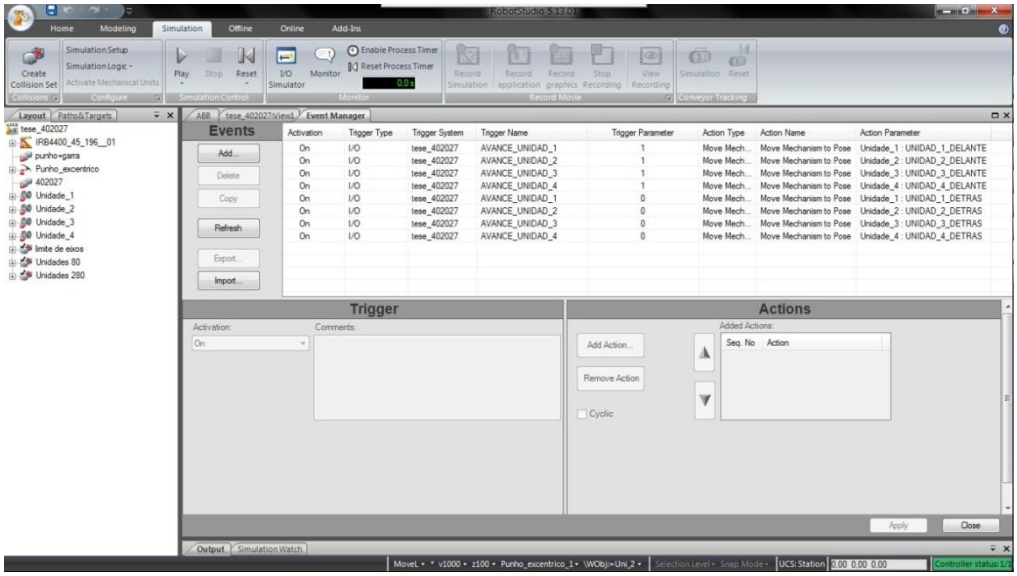

Figura 54 Aspecto final da janela "`Event Manager`" com todos os eventos criados

Terminado o processo de configuração dos eventos o modelo desenvolvido está aproximado da realidade, sendo possível começar a programação propriamente dita.

5.3. DESENVOLVIMENTO DO PROGRAMA

Para se programar os movimentos pretendidos do robô utilizando o ABB RobotStudio existem duas formas:

- Programação em código RAPID

  No caso da programação com código é extremamente complexo conseguir-se uma boa precisão, uma vez que a visualização da célula não é actualizada a cada linha escrita, ou seja ao desenvolver em código é necessário carregar o respectivo código para a estação, sendo que quanto mais código e pontos existirem mais lento se torna esse carregamento.

- Programação em ambiente gráfico

  Na programação em ambiente gráfico a obtenção de movimentos muito próximos do pretendido é extremamente fácil, desde que a modelação esteja correcta e seja fidedigna à realidade. Neste tipo de programação podem-se utilizar as geometrias criadas para ajudar à criação de *targets*; para criar um *target* pode seleccionar-se a superfície e o

target é criado na face da mesma (existem diversas opções para selecção de superfícies ou pontos contidos na mesma).

Após a criação de todos os *targets*, e antes de ser possível passar o programa para a célula real, existe a necessidade de carregar tudo o que foi feito em ambiente gráfico para o controlador virtual. Ao longo deste processo o RobotStudio gera automaticamente o código RAPID necessário.

Para o desenvolvimento deste trabalho foi utilizada uma relação entre a programação em código RAPID e a programação em ambiente gráfico, porque existia a necessidade de adicionar algumas condições para melhor funcionamento dos programas e dos movimentos. Esta relação entre ambos os desenvolvimentos deve-se ao facto de que para efectuar um correcto lixamento de uma peça é necessário ter em conta a relação existente entre velocidade da lixa, pressão exercida sobre a lixa e o movimento da peça face à lixa, pois todos estes factores influenciam a qualidade do lixamento efectuado.

Para o desenvolvimento de cada um dos programas foi primeiro necessário pedir a um afinador/lixador experiente, e com conhecimento dos processos de lixamento, que avaliasse a peça e indicasse quais os movimentos e como estes deveriam ser efectuados (deve ter-se em conta que cada peça tem as suas particularidades e a ordem dos movimentos executados é muito importante para a obtenção de resultados satisfatórios). Após deter as indicações do afinador/lixador pode-se iniciar a programação utilizando a célula simulada previamente criada.

Para a criação dos *targets* necessários para a definição dos movimentos, e como se pode ver na Figura 55, basta seleccionar o botão "`Target`" e posteriormente configurar e posicionar os *targets* pretendidos.

Como se pode ver na Figura 56, após a abertura desta funcionalidade existem diversos campos que podem ou não ser preenchidos facilitando assim a programação.

A título de exemplo, na imagem da esquerda podem ver-se quais as referências disponíveis para a criação dos *targets* (neste caso foi escolhida a referência *workobject*, para ser mais fácil e precisa a colocação dos *targets*); na imagem da direita pode ver-se o aspecto da janela do simulador após a colocação de um *target* com a origem no *workobject*, tal como assinalado na figura.

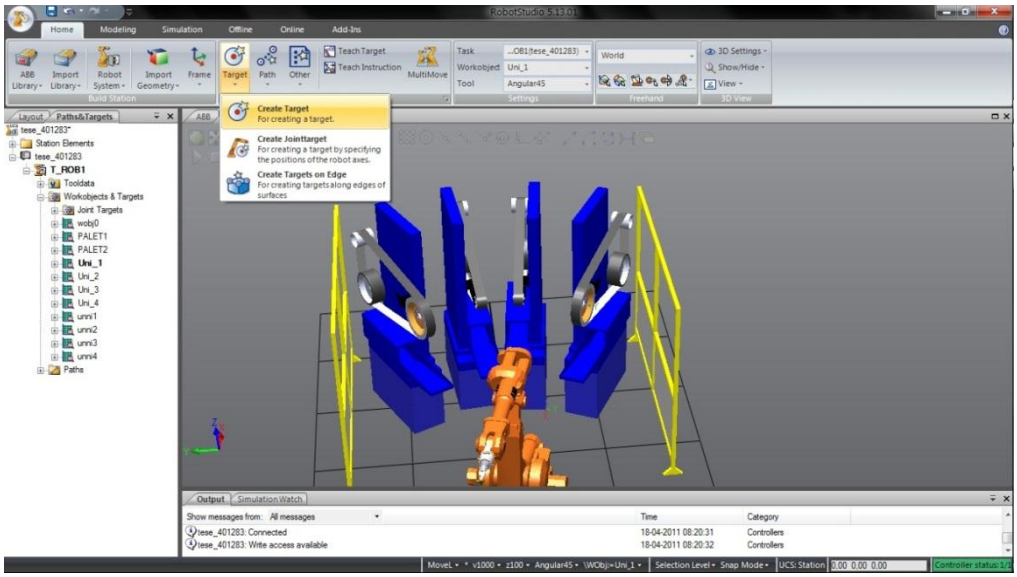

Figura 55 Criação dos *targets*

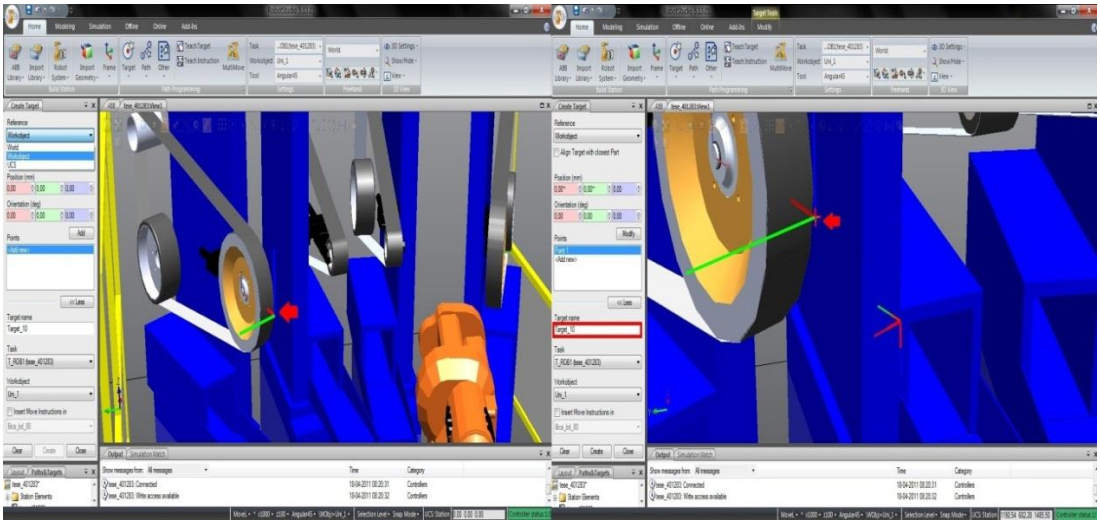

Figura 56 Opções disponíveis para a criação e configuração dos *targets*

Posteriormente procede-se à alteração do nome do *target*, apenas para facilitar a programação enquanto se utiliza o ambiente gráfico, pois quando se trabalha com pontos *inline*[2] (como é o caso) os nomes dos *targets* desaparecem vendo-se apenas as suas coordenadas e configurações, tal como se pode ver no extracto de código apresentado em baixo. É de referir que no caso da utilização de *targets* sem estarem definidos como *inline* irá aparecer no módulo do programa a definição de cada um dos *targets*.

[2] *Targets inline* é uma funcionalidade existente no ABB RobotStudio que faz com o código gerado seja igual ao código gerado pelo robô. Quando se programa no robô normalmente programa-se um ponto agregado a um movimento. No ABB RobotStudio, e por omissão, faz-se precisamente o contrário, uma vez que os pontos são programados antes de se decidir os movimentos.

```
!Movimento com target sem definição inline
CONST robtarget Bica_80_10:=[[-246.949920593432,-
68.0000000000001,8.94090999015162E-
05],[0.0122866716294339,0.703907507182188,-
0.710077204760816,-0.0123944468167853],[0,0,-
2,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

MoveL Bica_80_10,v50,z10,Angular45\WObj:=Uni_1;

!Movimento com target com definição inline
MoveL [[-246.949920593432,-
68.0000000000001,8.94090999015162E-
05],[0.0122866716294339,0.703907507182188,-
0.710077204760816,-0.0123944468167853],[0,0,-
2,0],[9E9,9E9,9E9,9E9,9E9,9E9]],v50,z10,Angular45
\WObj:=Uni_1;
```

Após terminar a criação dos *targets* (pode criar-se um determinado número de *targets* e posteriormente acrescentar mais ou retirar alguns, dependendo das necessidades) pode então começar-se a criar os movimentos (*paths*).

Para iniciar a criação dos movimentos basta seleccionar um ou mais *targets* e, com o botão direito do rato, abrir o menu apresentado na Figura 57, seleccionando a criação de um novo *path* ou acrescentado os *targets* a um *path* já existente.

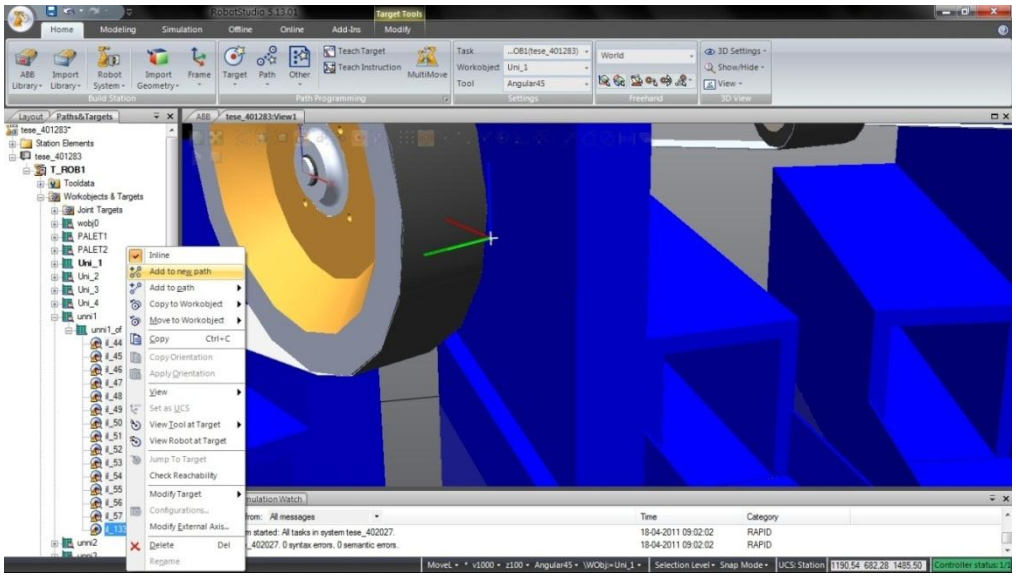

Figura 57 Menu para criação dos movimentos (definição dos *paths*)

Tendo os *paths* criados, existem diferentes opções associadas aos movimentos. Tal como se pode ver na Figura 58, podem efectuar-se várias alterações ao *paths* tais como mudar--lhes o nome, as configurações, entre outras.

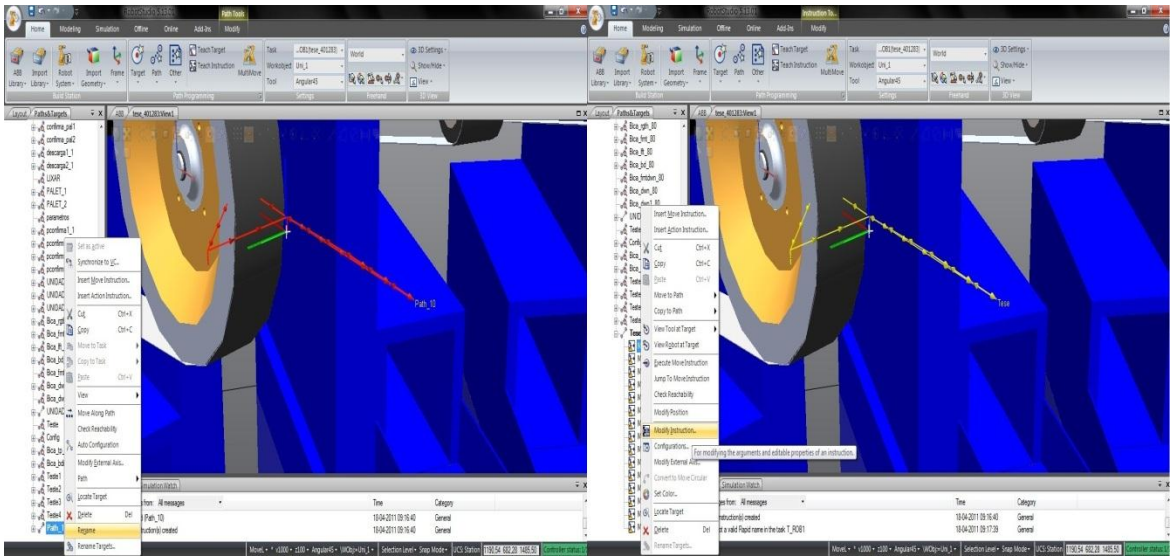

Figura 58 Menu que permite efectuar a configuração dos *paths*

Outros factores importantes nas tarefas de lixamento e no movimento dos robôs são a velocidade a que estes trabalham, o tipo de movimentos que efectuam, a precisão com que se posicionam sobre um determinado ponto e relativamente a que *tool* estão a trabalhar. Todas estas opções podem ser modificadas recorrendo a uma das funcionalidades ("`Modify Instruction`") existentes no menu apresentado na figura anterior (como se pode ver na Figura 59 existem diversas opções para cada uma dessas características).

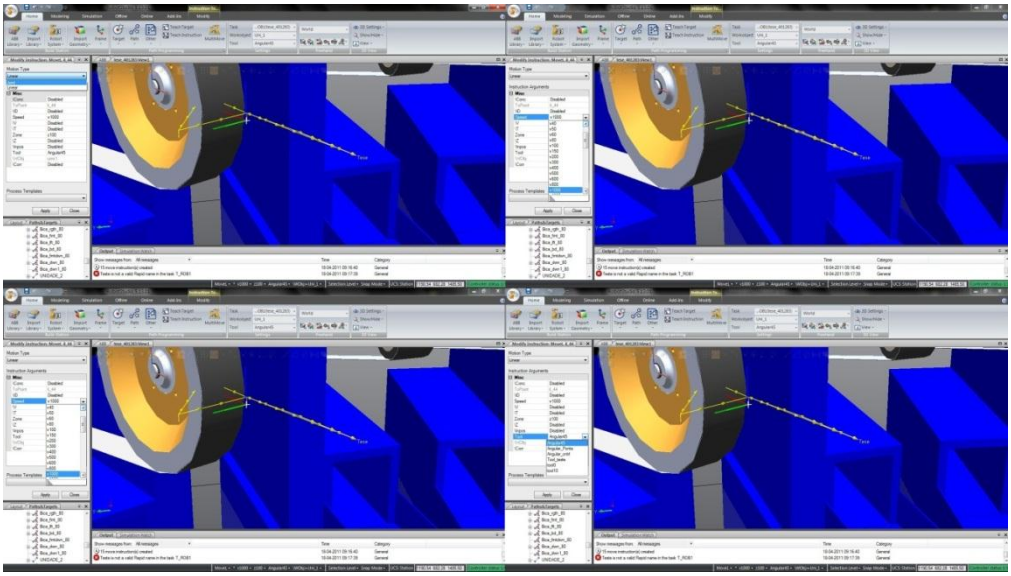

Figura 59 Configuração detalhada das instruções de movimento dos programas

Estando os movimentos configurados como pretendido, pode então passar-se à configuração das posições do robô. Isto significa que para o robô efectuar um determinado movimento, este pode mover-se com diferentes soluções da cinemática inversa (para cada

ponto existem diferentes soluções de cinemática inversa) e esta configuração serve para definir qual das soluções deve ser adoptada pelo robô. Para isso utiliza-se a funcionalidade "`Auto Configuration`", disponível no menu do *path*, que é acessível com o botão direito do rato. Esse menu é apresentado no lado direito da Figura 60, bem como o aspecto do *path* (aspecto do *path* refere-se à linha indicativa do *path* e seu sentido) que antecede a configuração.

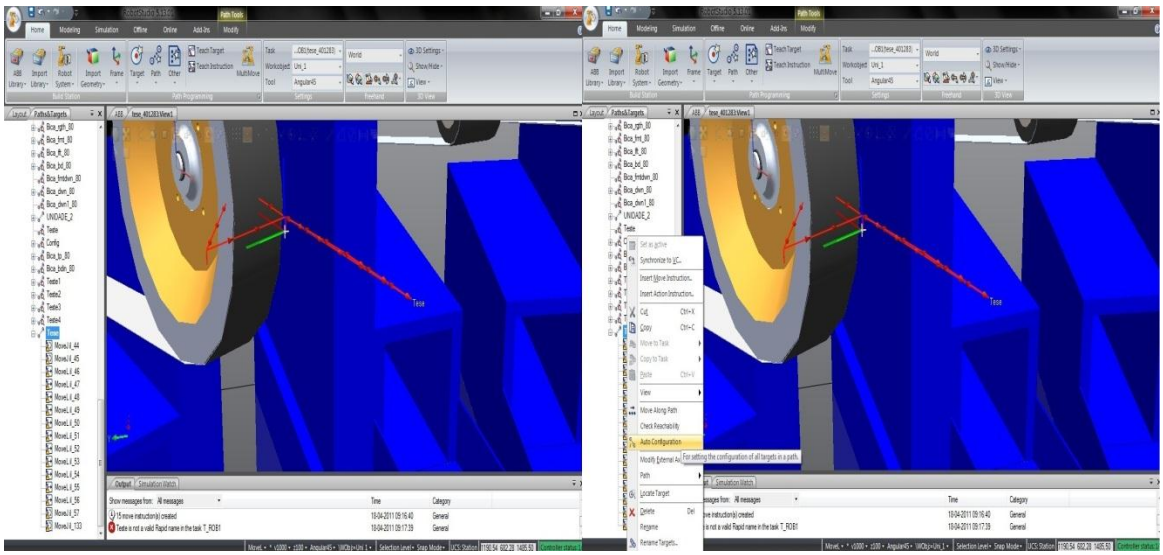

Figura 60 Funcionalidade "`Auto Configuration`"

Ao seleccionar esta funcionalidade aparece uma janela com as várias opções existentes, como se pode ver na Figura 61. Tal como foi referido anteriormente este processo serve para escolher a solução de cinemática inversa pretendida.

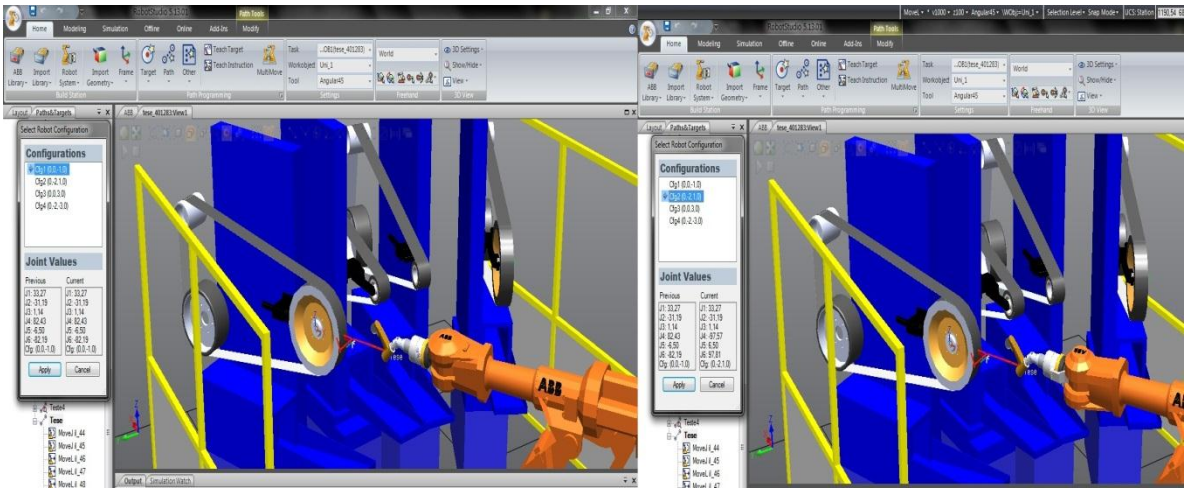

Figura 61 Configurações disponíveis para o robô executar o *path*

Com a conclusão deste passo o *path* está criado e definido, estando assim pronto para ser utilizado quando necessário. Caso não exista nenhum problema, a sua aparência deverá corresponder à apresentada na Figura 62.

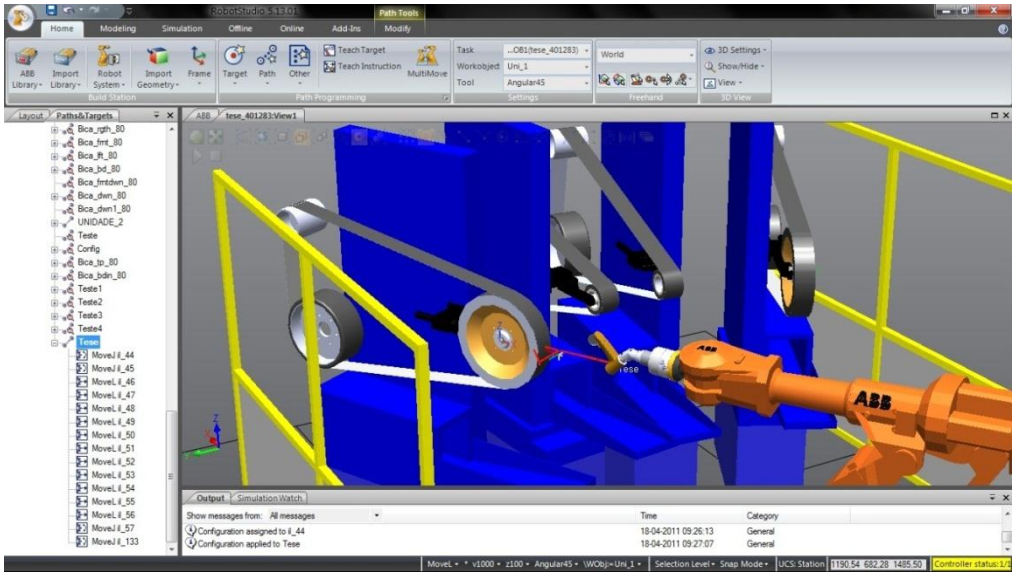

Figura 62 *Path* do robô após configurações

Todo o processo descrito ao longo deste capítulo é repetido para cada um dos programas criados, sendo que é necessário criar os *targets* e respectivos *paths* por diversas vezes, dependendo do número de rotinas necessárias.

Após todas as rotinas criadas pode-se carregar a programação desenvolvida no simulador gráfico para o controlador, passando assim todo o programa para uma versão código. A partir do momento em que se carrega o programa para o controlador, este faz as conversões para RAPID e, posteriormente, pode-se alterar ou acrescentar código consoante as necessidades.

Tendo concluído o desenvolvimento do código, o aspecto final do programa assemelha-se ao apresentado na Figura 63. De notar que apenas as rotinas referentes às quatro unidades são alteradas pois o restante código deste módulo é semelhante a todos os robôs. As rotinas não são totalmente apresentadas por se tratar apenas de um exemplo da totalidade do programa.

```
%%%
  VERSION:1
  LANGUAGE:ENGLISH
%%%

MODULE M8_65867
!***************************"************************
!*********************Tooll's*********************
!***************************"************************
  !!tool lado esquerdo da torneira
  PERS tooldata ponta_1:=[TRUE,[[-159.493,-75.3655,280.3],[1,-6.12303E-17,1.87458E-33,-3.06152E-17]],[1,[0,0,1],[1,0,0,0],0,0,0]];
  !!tool lado direito da torneira
  PERS tooldata pona_2:=[TRUE,[[-159.493,75.365,280.3],[1,6.12303E-17,9.71615E-17,-3.06152E-17]],[1,[0,0,1],[1,0,0,0],0,0,0]];
  !!tool meio da torneira
  PERS tooldata inversor:=[TRUE,[[-159.493,-1.29679E-20,280.3],[1,-6.12303E-17,1.87458E-33,-3.06152E-17]],[7,[-4.86,0,61.55],[1,0,0,0],0.4,0.4,0.4]];
  !!tool ponta da bica
  PERS tooldata ponta_bica:=[TRUE,[[-167.493,-2.88658E-12,360.3],[1,-2.77217E-17,3.68629E-18,-3.55618E-17]],[7,[-4.86,0,61.55],[1,0,0,0],0.4,0.4,0.4]];
!*************************"************************
!**************Pontos carga e descarga**************
!*************************"************************
  CONST robtarget P2c1_2:=[[-469.13,57,73.45],[0.001804,0.704642,0.709537,-0.00578],[-1,-1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P2c1_3:=[[-469.1,57.02,58.95],[0.001815,0.70464,0.709539,-0.005786],[-1,-1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P2c1_1:=[[-469.41,56.84,206.19],[0.001805,0.704636,0.709544,-0.005786],[-1,-1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P2c1_4:=[[-469.2,56.96,112.88],[0.00181,0.704642,0.709538,-0.005774],[-1,-1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P2c1_5:=[[-469.38,56.84,242.58],[0.001809,0.70464,0.709539,-0.005792],[-1,-1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P2c1_6:=[[-508.04,34.31,120.26],[0.000953,0.00852,-0.999919,0.009532],[-2,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P1c1_6:=[[-469.3,54.63,200.25],[0.003474,0.708192,0.705983,-0.006419],[-2,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P1c1_5:=[[-469.15,54.91,95.6],[0.003473,0.708187,0.705988,-0.006412],[-2,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P1c1_4:=[[-469.08,55.01,60.85],[0.003463,0.708174,0.706001,-0.006407],[-2,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P1c1_3:=[[-469.12,54.94,86.8],[0.003469,0.708182,0.705993,-0.006405],[-2,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P1c1_2:=[[-469.19,54.79,133.11],[0.003474,0.708177,0.705998,-0.00642],[-2,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget P1c1_1:=[[-469.31,54.61,210.43],[0.003475,0.708191,0.705984,-0.006404],[-2,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  PERS num R1U1_prog:=450;
  PERS num R1U2_prog:=150;
  PERS num R1U3_prog:=150;
  PERS num R1U4_prog:=450;
  PERS num R1U1_act_prg:=450;
  PERS num R1U2_act_prg:=148;
  PERS num R1U3_act_prg:=149;
  PERS num R1U4_act_prg:=406;
  PERS num R2U1_prog:=450;
  PERS num R2U2_prog:=150;
  PERS num R2U3_prog:=150;
  PERS num R2U4_prog:=450;
  PERS num R2U1_act_prg:=450;
  PERS num R2U2_act_prg:=148;
  PERS num R2U3_act_prg:=149;
  PERS num R2U4_act_prg:=406;
  PERS num incvel1:=0.08;
  PERS num incvel2:=0.04;
  PERS num incvel3:=0.06;
  PERS num incvel4:=0.08;
  PERS num limbanda1:=110;
  PERS num limbanda3:=110;
  PERS num limbanda2:=110;
  PERS num limbanda4:=110;
  PERS num Num_palet:=2;
  VAR bool flag1:=FALSE;

  PROC velbanda

  PROC Cod_program

  PROC parametros

  PROC lixar

  PROC UNIDADE_1

  PROC UNIDADE_2

  PROC UNIDADE_3

  PROC UNIDADE_4

 !***************************"************************
 !*********************Unidade 1**********************
 !***************************"************************
  PROC corpo

  PROC frente

  PROC baixo

  PROC ponta

 !***************************"************************
 !*********************Unidade 2**********************
 !***************************"************************
  PROC costas

  PROC cima

 !***************************"************************
 !*********************Unidade 3**********************
 !***************************"************************
  PROC costas_U3

  PROC cima_U3

 !***************************"************************
 !*********************Unidade 4**********************
 !***************************"************************
  PROC corpo_U4

  PROC frente_U4

  PROC baixo_U4

  PROC ponta_U4

 !***************************"************************
 !*********************Palete 1***********************
 !***************************"************************
  PROC PALET_1

  PROC carga1_1

  PROC descarga1_1

 !***************************"************************
 !*********************Palete 2***********************
 !***************************"************************
  PROC PALET_2

  PROC carga2_1

  PROC descarga2_1

  PROC Conf_Pieza

ENDMODULE
```

Figura 63 Exemplo de código RAPID de um programa para o robô ABB IRB4400

Relativamente ao código apresentado nesta figura, realça-se que os *targets* definidos no início do módulo são referentes às cargas e descargas e são ajustados consoante o programa em questão; as variáveis numéricas "`R1U1_prog`" até "`R2U4_act_prog`" são variáveis nas quais estão definidos os diâmetros das rodas para questões de afinação; as variáveis "`incvel`" definem os incrementos de velocidade em cada unidade (estes incrementos são efectuados automaticamente ao longo da duração de vida de uma lixa, sendo estes incrementos controlados na rotina "`velbanda`"); as variáveis "`limbanda`" definem o limite máximo de peças que podem ser lixadas por unidade antes de existir a necessidade de trocar a lixa; por último, a variável "`Num_palet`" define a versão das paletes que estão a ser utilizadas (esta funcionalidade encontra-se actualmente em desuso devido a problemas com a comunicação com os computadores, uma vez que estes tendem a avariar com muita facilidade devido às elevadas temperaturas e ao excesso de pó).

6. CONCLUSÕES

Ao longo deste trabalho foi perceptível que associadas às tarefas que foram sendo realizadas ter-se-iam de superar inúmeras dificuldades para a correcta modelação e programação, mas que com o desenvolvimento de algumas ferramentas seria possível conseguir uma modelação suficientemente detalhada para alcançar bons resultados.

O primeiro problema surgiu com a pesquisa bibliográfica para a aquisição de conhecimentos sobre o estado da arte relativo à lixagem robotizada. Foi extremamente complicado descobrir documentação explicativa e fidedigna sobre a temática da utilização de robôs associados ao lixamento, pois a informação existente era pouco explicativa sendo, muitas vezes, apenas referidas algumas funcionalidades da globalidade das células. Como tal o desenvolvimento desta parte do trabalho foi essencialmente baseado nos conhecimentos e na experiência dos funcionários da Grohe, com os quais tive o prazer de trabalhar e aprender as diversas particularidades do processo de lixamento e do funcionamento das células.

Com a obtenção de algum conhecimento acerca do funcionamento geral do processo e das células decidiu-se então passar à fase de modelação da célula. Nesta fase o trabalho foi extremamente aliciante e motivador, com a necessidade de desenvolver peças específicas (por exemplo o dispositivo para definição dos *workobjects* das unidades) para conseguir

obter bons resultados, apareceu a obrigatoriedade de pôr em prática todos os conhecimentos adquiridos, quer no ISEP, quer com os funcionários da Grohe, o que facilitou e impulsionou o desenvolvimento rápido e eficaz das peças necessárias.

Ao longo do processo de desenvolvimento dos referidos dispositivos e configuração das unidades foram sendo efectuados alguns testes para verificar o estado em que se encontrava a modelação, bem como para verificar se os programas que estavam a ser desenvolvidos podiam ou não ser viáveis. Com esses testes foram notórias as diferentes variáveis que influíam no processo de lixamento. No primeiro teste efectuado, o programa, apesar de estar correcto e com pressões constantes, deformou em demasia a peça, devido à primeira modelação não ter contemplado os ângulos entre a lixa e a roda. Este problema foi rapidamente ultrapassado com a introdução da representação da lixa na modelação da célula o que se veio a comprovar ser uma mais-valia, quer para a correcta programação, quer na correcta percepção da colocação da peça face à roda.

Após o término da modelação foi realizado um segundo teste (teste final) no qual se obtiveram resultados muito satisfatórios face ao que era pretendido. É de relembrar que o objectivo da realização deste trabalho era o desenvolvimento de programas *offline* para minimizar os tempos de paragem para programação e, com a utilização do programa desenvolvido *offline*, esse tempo de paragem foi drasticamente reduzido. Antes da utilização da programação *offline* a programação de um produto podia envolver paragens superiores a três dias (considerando a programação em turnos de 8 horas, isto representa mais de 24h horas de paragem). Com a utilização da programação *offline* conseguiu-se diminuir esse tempo pondo o robô em funcionamento ao fim de aproximadamente 8 horas o que representa uma redução nunca inferior a 66,5%. Tendo em conta que a média de produção de uma célula de trabalho ronda as 120 peças por turno são evidentes os ganhos obtidos com esta mudança de estratégia.

Como se pode ver na Figura 64 o custo/hora de um robô é elevado e com os tempos de paragens inerentes à programação *online* os gastos para programar um novo produto são muito elevados. Sendo necessário pelo menos três dias (24 h) para programar um novo produto, o custo inerente a esta paragem é de 325,92 €, o que é financeiramente insustentável para uma empresa com a constante criação de novos produtos. Quando em comparação com os resultados obtidos com a programação *offline*, este custo passa apenas a ser de 108,64 €, representando uma redução dos gastos em cerca de 217,28 €.

Custo Médio (h)	Ganhos por Produto (€)
13,58	217,28

	Tempo de Paragem (h)	Gastos (€)
Programação *Online*	24	325,92
Programação O*ffline*	8	108,64

Figura 64 Custo/hora de um robô

Um dos factores relevantes no tempo de paragem quando num cenário de programação *online* é a complexidade das peças e como a Grohe é uma empresa que procura sempre a inovação, existe a necessidade de inovar nos formatos das torneiras. Graças a essa inovação, as peças cada vez mais são complexas e com formas inconstantes, o que dificulta a programação aumentando o seu tempo de implementação. Mais um dos motivos pelos quais a implementação da programação *offline* foi uma mais-valia.

Ao longo da realização deste trabalho estive também envolvido em alguns projectos de melhoramento e optimização do processo de lixamento de algumas peças, bem como pude aprender e ajudar a efectuar diversas tarefas de manutenção e reparação de robôs, incluindo troca de motores, calibrações ou mesmo a anulação de folgas em determinados eixos. Tendo em conta todo o conhecimento e experiência que adquiri ao longo deste estágio, sou da opinião que a minha decisão de concorrer a este foi a mais acertada e é extremamente gratificante poder ter pertencido a um grupo tão coeso e importante internacionalmente como é a Grohe.

Como conclusão do trabalho em si pode dizer-se que a utilização de *softwares* como o ABB RobotStudio traz grandes vantagens na redução dos tempos de paragem dos robôs e, ao mesmo tempo, tornam o processo de desenvolvimento e avaliação de um novo produto mais fácil, apesar de todas as dificuldades inerentes à correcta modelação do mundo. Após a correcta modelação da célula apenas existe a necessidade de voltar a alterar o modelo da célula caso se faça alguma alteração na célula real. Caso contrário pode utilizar-se sempre o mesmo modelo para todos os desenvolvimentos relativos a essa célula.

Para finalizar o trabalho destacam-se, de seguida, alguns desenvolvimentos futuros possíveis de serem implementados:

- Desenvolvimento de um dispositivo para definição dos *workobjects* das paletes (este dispositivo já está projectado e pedido mas ainda não tinha sido entregue até a data de entrega deste relatório);
- Utilização de sistemas de visão artificial com recurso a câmaras para detectar defeitos nas peças, efectuando auto ajustes no programa;
- Possível alteração do sistema de funcionamento das células trocando os papéis actualmente existentes, isto é, o robô passar a segurar o material abrasivo e a peça ficar em unidades de suporte fixas.

Resumindo, esta foi uma oportunidade de excelência para entrar em contacto com o mundo do trabalho, com a utilização dos sistemas robóticos na indústria, para adquirir experiência profissional e para melhorar e reforçar os conhecimentos sobre a utilização de robôs em diversas tarefas e quais as suas principais limitações.

Referências Documentais

[1] Nof, Shimon Y. - Handbook of Industrial Robotics; John Wiley & Sons, Inc; 1999. ISBN 0-471-17783-0

[2] Siciliano, Bruno; Khatib, Oussama - Springer Handbook of Robotics; Springer+Business Media; 2008. ISBN 978-3-540-23957-4, e-ISBN 978-3-540-30301-5

[3] Mektronic (2001) – Robótica; Acedido em: 10 de Outubro de 2010; em: http://mektronica.vilabol.uol.com.br/robotica.htm

[4] Siemens AG (2005) - History of Industrial Automation; Acedido em: 10 de Outubro de 2010; em: http://www.siemens.com/innovation/en/publikationen/publications_pof/pof_spring_2005/history_of_industrial_automation.htm

[5] Pinto, Jim (2007, Abril) - A short history of Automation growth; Acedido em: 10 de Outubro de 2010; em: http://www.jimpinto.com/writings/automationhistory.html

[6] Vitorino, José; Introdução ao IRC5.0; Versão 2.0; 2007

[7] Craig, John J. – Introduction to Robotics Mechanics and Control; Addison-Wesley Publishing Company, Inc; 1989. ISBN 0-201-09528-9

[8] Filho, Teodiano - Aplicação de Robôs nas Indústrias, 2000; Acedido em: 10 de Outubro de 2010, em: http://www2.ele.ufes.br/~tfbastos/RobMov/robosindustriais.pdf

[9] Dynalog, Solutions – Off-line programing; Acedido em: 24 de Maio de 2011; em: http://www.dynalog-us.com/Solutions/?CategoryID=2&OvID=24

[10] ABB; Acedido em: 24 de Maio de 2011; em: http://www.abb.pt/product/pt/9AAC111580.aspx?country=PT

[11] Kuka; Acedido em: 24 de Maio de 2011; em: http://www.kuka-robotics.com/en/pressevents/productnews/NN_040630_KUKASim.htm

[12] Fanuc Robotics; Acedido em: 24 de Maio de 2011; em: http://www.fanucrobotics.com/Products/vision-software/AtoZ.aspx

[13] Camelot IT for robots; Acedido em: 24 de Maio de 2011; em: http://www.camelot.dk/pages/robottechpapers/robotprogramming.aspx

[14] Delfoi; Acedido em: 24 de Maio de 2011; em: http://www.delfoi.com/web/solutions/production/en_GB/robotics/

[15] Delmia; Acedido em: 24 de Maio de 2011; em: http://www.3ds.com/products/

[16] Tecnomatix; Acedido em: 24 de Maio de 2011; em: http://www.plm.automation.siemens.com/en_us/products/tecnomatix/

Histórico

- 31 de Fevereiro de 2010, Versão 1.0, mailto:1030427@isep.ipp.pt
- 28 de Abril de 2011, Versão 2.0, mailto:1030427@isep.ipp.pt
- 1 de Junho de 2011, Versão 3.0, mailto:1030427@isep.ipp.pt
- 13 de Junho de 2011, Versão 3.1, mailto:1030427@isep.ipp.pt
- 21 de Junho de 2011, Versão 3.2, mailto:1030427@isep.ipp.pt
- 1 de Julho de 2011, Versão 3.3, mailto:1030427@isep.ipp.pt