Filipe Correia Carrasquinho

Licenciado em Ciências de Engenharia

Electrotécnica e de Computadores

Ferramenta de Simulação para Robô Industrial

Dissertação para obtenção do Grau de Mestre em
Engenharia Electrotécnica e de Computadores

Orientador: João Almeida das Rosas, Professor Auxiliar, FCT-UNL

Júri:

Presidente: Luís Filipe Figueira de Brito Palma, Professor Auxiliar, FCT-UNL

Arguente: Tiago Oliveira Machado de Figueiredo Cardoso, Professor Auxiliar, FCT-UNL

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2015

Ferramenta de Simulação para Robô Industrial

Aos meus pais e irmão

Agradecimentos

A realização desta dissertação põe um fim a uma etapa significativa da minha vida, sendo a mais importante a nível académico até ao momento, que contou com apoios e incentivos aos quais estarei eternamente grato por tornarem possível todo este percurso.

Assim, dou os meus sinceros agradecimentos a todas as pessoas que de alguma forma contribuíram para a criação e conclusão do projecto que culminou nesta dissertação, mas também a todas as que marcaram certos momentos da minha vida durante toda esta fase académica.

Primeiramente, vou começar por agradecer ao Professor Doutor João Rosas, por me ter aceitado como seu orientando, assim como por todo o seu apoio, orientação e compreensão demonstrados durante a realização deste projecto, que consistiu numa total colaboração em solucionar dúvidas e problemas, mas também pelo incentivo e entusiasmo fornecido ao longo deste trabalho.

Agradeço também aos meus amigos e colegas de curso pela sua amizade, que preencheu esta etapa da minha vida com óptimos momentos, repletos de companheirismo, força e humor, e em termos académicos no seu total profissionalismo, agradecendo em particular: André Silva, Carlos Posse, Daniel Nunes, Daniela Silvestre, David Cantarinha, David Fernandes, Diogo Silva, Filipe Viegas, primo Filipe Carrasquinho, Francisco Coito, Frederico Monteiro, Gonçalo Maia, Henrique Ferreira, Iúri Rodrigues, João Alarcão, João Almeida, João Pecorelli, João Santos, José Franco, José Marques, José Simão, Micael Calado, Miguel Pereira, Paulo Martins, Pedro Carrasco, Pedro Choon, Pedro Pereira, Rúben Santinhos, Sérgio Coelho, Sérgio Saro, Sofia Ribeiro, Ricardo Bernardo e Ricardo Mexia.

Finalmente, um sincero e profundo agradecimento a toda a minha família, em especial aos meus pais, irmão e avó, no qual sem eles nada disto seria possível, agradecendo pela força e motivação fornecidos ao longo de todo este percurso.

Muito Obrigado!

Resumo

Actualmente, os meios de produção industrial estão em constante desenvolvimento, de forma a corresponderem a um mercado cada vez mais globalizado, caracterizado por uma forte competição, uma pressão na redução de custos, e a existência de produtos e serviços personalizados. Entretanto, ao nível de equipamento de robótica, também se tem notado esta pressão por parte do mercado, exigindo-se equipamento com uma complexidade crescente, tanto ao nível da sua morfologia, do seu controlo e da sua utilização. Ao nível da sua utilização, tornou-se então necessário adoptar abordagens que permitam um maior controlo quanto a falhas de manuseamento e respectiva modelação destes robôs. Uma destas abordagens é a simulação. A simulação é uma das áreas com maior crescimento a nível tecnológico, sendo utilizada em várias áreas de tecnologia e informação devido à sua capacidade de adaptação, tornando-se numa área bastante dinâmica e flexível. Dentro dum contexto de robôs industriais, as aplicações mais típicas consistem na programação "*offline*" de robôs e na aprendizagem de robótica, tendo como objectivo estudar e analisar os comportamentos de determinados sistemas, com o intuito de reduzir a necessidade de intervenção humana em tarefas repetitivas, perigosas e dispendiosas. Além disso, a simulação está também presente na computação informática nas salas de aulas, por forma a auxiliar o professor. Foi então idealizado um sistema de simulação capaz de imitar um robô industrial em que os alunos terão que conseguir controlar, para que os vários testes sejam primeiramente implementados no simulador, diminuindo assim, o risco de causar acidentes no seu congênere real durante a fase de testes. O sistema consiste numa interface em que possibilita ao utilizador fazer um controlo manual do robô simulado, assim como ser capaz de o controlar por comandos requisitados por uma aplicação exterior através de serviços WEB, permitindo assim, a simulação de vários testes no robô industrial em causa.

Palavras-chave: Modelação, Motor de Jogo, Robô Industrial, Serviços WEB, Simulação.

Abstract

Currently, the industrial means of production are in constant development in order to correspond to an increasingly globalized market, characterized by strong competition, pressure to reduce costs, and the availability of customized products and services. In terms of robotic equipment, it has also been noticed this pressure from the market, requiring equipment with increasing complexity, both in terms of their morphology, their control and use. In terms of its use, it became necessary to adopt approaches that allow greater control on the handling of faults and the modelling of these robots. One such approach is the simulation. The simulation is becoming one of the fastest growing areas in technology, being used in various fields of technology and information from the market due to its adaptability, making it a very dynamic and flexible area. Within industrial robots, the most typical applications consist of offline programming on robots and learning robotics, aiming to study and analyse the behaviour of certain systems, in order to reduce the need for human intervention on repetitive, dangerous and costly tasks. In addition, the simulation is also present in computer computing in the classroom in order to help the teacher. It was then designed a simulation system that can mimic an industrial robot in which students have to be able to control so that the various tests are first implemented in the simulator, thereby decreasing the risk of causing accidents on its real counterpart during the testing phase. The system comprises an interface that enables the user to make a manual control of the simulated robot, as well as being able to control by commands requested by an external application through WEB services, thus allowing the simulation of several tests on the industrial robot concerned.

Keywords: Industrial Robot, Modulation, Game Engine, WEB Services, Simulation.

Índice de Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

2D	Espaço Bidimensional
3D	Espaço Tridimensional
CAD	Computer Aided Design
HTTP	HyperText Transfer Protocol
JSON	JavaScript Object Notation
REST	Representational State Transfer
TIC	Tecnologias de Informação e Comunicação
WEB	World Wide Web
WPF	Windows Presentation Foundation
XAML	eXtensible Application Markup Language
XML	eXtensible Markup Language
XNA	XNA's Not Acronymed

1

Introdução

Os meios de produção industrial necessitam dum contínuo desenvolvimento, de forma a poderem corresponder com as necessidades dum mercado globalizado e bastante competitivo, exercendo uma pressão para uma personalização de produtos, menor consumo de materiais e diminuição de custos. Os sistemas de manufactura actualmente aplicados nestes meios de produção têm de ser mais flexíveis e adaptativos a ponto de conseguirem acompanhar a constante evolução desse mercado.

Como consequência, para que estes sistemas de manufactura consigam acompanhar esse desenvolvimento, as máquinas implementadas vão ganhando uma complexidade crescente, tanto ao nível da sua estrutura e utilização, onde se podem destacar os robôs ou sistemas robóticos. Devido à sua complexidade, é necessário também adaptar formas mais adequadas para a respectiva modelação, controlo e utilização dos referidos robôs. Mais especificamente ao nível da sua utilização, os robôs são preparados para funcionarem em ambientes disruptivos para o operador humano e, portanto, a incorrecta utilização reveste-se de perigo para o operador humano e outros recursos que o rodeiam.

É assim necessário adoptar abordagens que permitam detectar eventuais falhas de utilização, podendo-se assim, minimizar os riscos da sua ocorrência, sendo que, as melhores abordagens são sempre aquelas que permitem evitar os problemas antes destes ocorrerem. A simulação é uma destas abordagens.

Recorrendo à simulação, a qual se tornou uma área inovadora capaz de auxiliar as mais diversas áreas de actividade humana, é possível utilizar o modelo que representa um qualquer sistema, um robô neste caso concreto, e simular o seu comportamento de forma a poder-se efectuar uma avaliação antecipada do seu desempenho.

No caso concreto dos robôs, pretende-se também verificar que os programas que controlam o seu funcionamento possam ser testados, verificando-se que funcionam de acordo com as especificações. Também aqui, a simulação permite aferir que os programas de controlo de um robô funcionam correctamente, minimizando-se o potencial dos erros que poderiam surgir depois do robô ter sido colocado em modo de produção. Por outro lado, a já referida complexidade dos robôs requer habilitações

adequadas por parte dos operadores que os vão utilizar. De facto, aprender a utilizar e programar os robôs de uma forma segura e produtiva é extremamente importante, pois eventuais falhas na sua programação podem ser bastante perigosas para pessoas e equipamentos. Uma forma de minimizar estes riscos consiste em recorrer a uma aprendizagem e também desenvolvimento de programas, seguindo uma abordagem baseada em simulação.

Seguindo então este paradigma de simulação, este trabalho consistiu no desenvolvimento duma ferramenta de simulação para um robô industrial. Este simulador proporciona um ambiente virtual em que um utilizador consegue interagir com um robô simulado e programar o seu funcionamento, permitindo também, desenvolver novas formas de programar um robô, de complexidade crescente, para testa-las de uma forma em diferido. Por exemplo, pode-se colocar o robô simulado em situações de mau funcionamento, e desenvolver código correspondente para uma recuperação preventiva.

Se utilizado num contexto educacional, os estudantes, numa fase inicial da sua formação, não incorrem em perigo durante as primeiras interacções com os robôs, permitindo uma aprendizagem mais rápida e eficaz. Em termos económicos, estas vantagens, sem os erros com efeitos nefastos em equipamento real, trazem benefícios financeiros, pois evita-se os consequentes custos de reparação e desperdícios de recursos.

Para o desenvolvimento da ferramenta de simulação, recorreu-se à utilização das recentes Tecnologias da Informação e Comunicação (TIC), nomeadamente, aplicações para modelação gráfica, motores de jogos e serviços WEB (*World Wide Web*), incluindo as programações em linguagens C++ e C#.

1.1 Contexto e Motivação

Nas empresas de manufactura actuais, a simulação assume um papel preponderante na forma como as empresas implementam os seus processos nos diferentes robôs industriais, por forma a obter sucesso nos seus produtos. Isto estabelece uma análise prévia nas soluções que se pretende alcançar, tornando a simulação algo que fornece à indústria aplicações de estratégia no mercado.

No entanto, nota-se ainda uma ausência de ferramentas que permitam modelar e simular problemas em robôs industriais no âmbito da educação, daí a motivação em criar uma ferramenta capaz de auxiliar os alunos a obterem um melhor conhecimento quanto às funcionalidades de determinados robôs que estejam a ser leccionados. Este resultado será realizado com uma aplicação integrada sobre um motor de jogos em que o utilizador será capaz de testar, manualmente, as diferentes acções possíveis

do modelo em questão, e também através de outros programas criados pelos próprios utilizadores, para uma melhor percepção das funcionalidades que o robô analisado é capaz de realizar.

Este sistema em concreto consiste num simulador do braço robótico das versões ROB3 / ROB3i (caracterizado no Capítulo 3), que substituirá o seu congênere real, de forma a serem feitos o menor número de testes possível no robô, para uma menor probabilidade de falhas e acidentes. Será um projecto aplicado, então, no âmbito da cadeira de Robótica na área de Electrotecnia da Faculdade de Ciências e Tecnologia, cujo resultado final, o simulador referido, proporcionará efeitos benéficos na forma de aprender os conceitos de Robótica na referida disciplina.

1.2 Metodologia de Trabalho

Para esta dissertação foi considerada e utilizada uma metodologia de trabalho inspirada pelo método científico descrito em (Schafersman, 1997). Neste processo, inicia-se por uma investigação teórica (onde se faz uma descoberta científica quanto à caracterização do problema), passado para uma fase mais prática de implementação e experimentação, concluindo com uma análise dos resultados obtidos (através de uma validação e posterior aceitação / rejeição da solução formulada como resposta ao problema). O processo de trabalho para esta dissertação é, então, representada na Figura 1.1.

Através da Figura 1.1, verifica-se que todo este processo de trabalho é baseado nos passos seguintes:

- **Identificação do Problema:** É uma etapa importante visto que consiste em encontrar e definir qual o problema e suas características. Desta forma, o problema identificado para esta dissertação é o da criação de uma ferramenta que consiga simular um determinado robô industrial, no âmbito da educação;

- **Investigação:** Este passo define a parte em que toda a informação relativa à caracterização do problema identificado anteriormente é adquirida, com base em artigos e projectos científicos existentes na área em causa. Para este caso específico, consiste na recolha de informação e dados importantes quanto aos vários tipos de simulador de robôs industriais já existentes, e quanto aos tipos de sistemas computacionais (CAD, motores de jogo) utilizados que permitam a criação de um simulador;

- **Formulação da Hipótese:** Nesta etapa, efectua-se a criação de uma hipótese de solução para o problema proposto, com base em toda a investigação previamente feita, de maneira a especificar e a focar melhor os resultados desejáveis. Para esta dissertação, a hipótese imposta é a da criação de um simulador de um específico robô industrial;

- **Implementação:** Tendo em vista a anterior caraterização do problema e identificação da hipótese, segue-se para a realização do desenvolvimento de ferramentas que vão permitir, posteriormente, a um teste, de forma a ser possível rectificar eventuais problemas existentes no decorrer do projecto;

- **Teste da Hipótese:** Fase em que são aplicados diversos testes para que se consiga obter resultados viáveis que possam ser utilizados para a validação da hipótese gerada. Neste caso, os resultados esperados coincidem com o bom funcionamento do robô simulado, em relação às suas possíveis acções de trabalho, assim como ao correcto uso do protocolo de comunicação no simulador, de forma similar ao robô real;

- **Validação da Hipótese:** Após realizados os testes feitos anteriormente, passa-se para a fase em que se vai impor uma análise e interpretação dos dados obtidos, sendo depois verificada a validação da hipótese proposta. Assim sendo, caso os resultados gerados sejam satisfatórios e correspondam às previsões previamente estabelecidas, responde-se ao problema imposto, permitindo validar a hipótese; caso os resultados obtidos não sejam os esperados, deverá ser reformulada uma nova hipótese, por forma a aperfeiçoar a abordagem inicial do projecto. Para o caso em questão, consiste nos vários testes de movimentação e orientação do robô, assim como da comunicação entre o simulador e outra aplicação exterior;

- **Publicação dos Resultados:** Para este passo final, de acordo com o resultado da experimentação do trabalho de investigação, uma publicação de resultados e conclusões são efectuadas, sendo posteriormente finalizado por um documento sobre a hipótese, tal como esta dissertação.

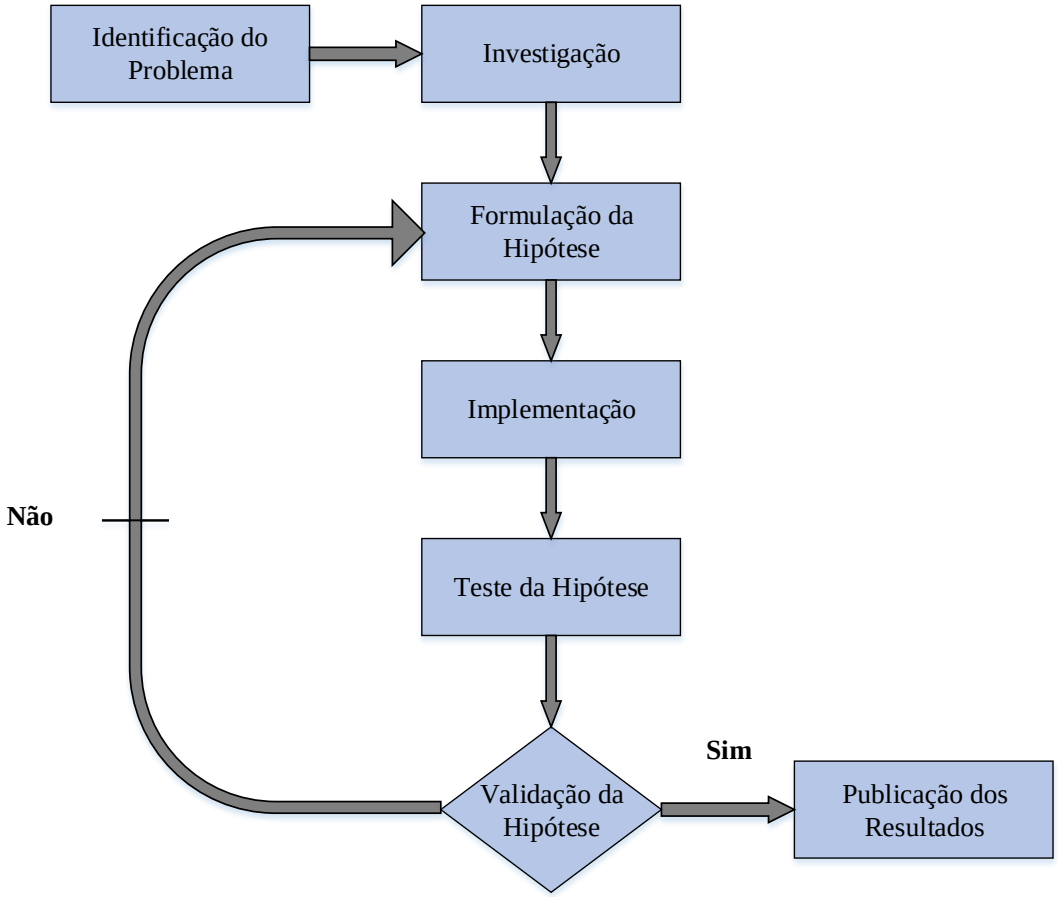

Figura 1.1: Diagrama de esquema da dissertação.

1.3 Formulação da Questão de Pesquisa

Dados os aspectos anteriormente mencionados em relação à segurança de materiais e pessoas no decorrer da aprendizagem, a questão de pesquisa que se coloca é:

"Qual a forma adequada para a aprendizagem de robótica, de forma a salvaguardar a segurança dos alunos e a integridade do equipamento robótico?"

Uma hipótese de resposta para esta questão pode basear-se na utilização da simulação. Para obter-se essa resposta vai-se proceder à criação de um simulador para um robô industrial que irá permitir o ensino de robótica baseada no paradigma da simulação. A concepção do simulador baseia-se na tecnologia dos motores de jogos, de modo a poder-se beneficiar dos inúmeros recursos de computação gráfica disponibilizados.

1.4 Esquema da Dissertação

Esta dissertação foi estruturada em cinco diferentes capítulos a seguir caracterizados:

- **Introdução:** Capítulo inicial desta dissertação, onde é apresentado uma contextualização do projecto, através de uma caracterização do problema e motivação para a sua realização. Além disso, também é representada a abordagem utilizada de metodologia de trabalho para a hipótese de solução formulada;

- **Estado de Arte:** Para este capítulo em específico, é apresentada toda a investigação efectuada quanto às áreas de interesse, nomeadamente as características principais quanto a robôs industriais, em relação aos simuladores existentes destes robôs, e também acerca das tecnologias que possibilitam a implementação destes simuladores;

- **Desenvolvimento da Ferramenta Proposta:** Esta secção da dissertação é focada na identificação do problema, sendo feito um levantamento de requisitos e especificações ao qual o sistema em causa deverá obedecer para a fase de implementação, sendo também realizada uma descrição da arquitectura geral do sistema e de todo o modelo do simulador que servirá como teste da solução proposta. Além disso, é feita uma descrição dos passos mais relevantes para o desenvolvimento deste projecto, por forma a originar a ferramenta de simulação proposta;

- **Validação:** Neste capítulo é onde se verifica que o objectivo principal é garantir que o robô simulado possui o mesmo comportamento do seu congênere real. Além disso, verifica-se nesta secção que a ferramenta desenvolvida satisfaz adequadamente os requisitos funcionais estabelecidos;

- **Conclusões:** Neste capítulo final são sumarizados os aspectos mais relevantes do projecto realizado e apresentado possíveis caminhos de pesquisas e implementações para trabalho futuro.

2

Estado de Arte

Neste capítulo é realizada uma investigação e pesquisa quanto ao estado actual dos simuladores de robôs industriais existentes, de modo a identificar a vantagem da presença de um simulador no âmbito de controlo de certos robôs industriais para a área da educação.

São também identificadas as ferramentas de desenvolvimento de um simulador, incluindo primeiramente, os sistemas computacionais de CAD existentes para a criação de modelos, assim como os motores de renderização e jogos para a definição de animações dos modelos efectuados, e também para a realização de cenários e esquemas de interface.

Foram identificados e analisados vários sistemas computacionais possíveis, de forma a identificar quais os que oferecem mais e melhores vantagens para o problema desta dissertação.

2.1 Sistemas Robóticos

Actualmente, o mercado dos robôs industriais tem vindo a evoluir bastante, isto porque robôs cada vez mais sofisticados e complexos têm vindo a ser desenvolvidos por forma a conseguir exercer um maior número de funções, de maneira às empresas conseguirem uma maior eficiência, produtividade e menor consumo de materiais.

Este mercado iniciou-se em 1961, onde Joseph Engelberger e George Devoe foram os primeiros a desenvolver robôs industriais, aplicando o primeiro robô industrial, o PUMA (*Programmable Universal Manipulator Arm*), na sua empresa, *Unimation* (Barata, 2012). Embora os robôs industriais sejam aplicados nas mais diversas áreas, desde montagem, soldadura, pintura, entre outras, as suas características principais mantêm-se semelhantes, sendo a seguir representados alguns conceitos e aspectos essenciais destes robôs.

Existem vários tipos de classificações quanto a robôs industriais, desde os robôs paralelos (que são formados por cadeias cinemáticas fechadas e são geralmente caracterizados por não possuírem

actuadores nos membros móveis), os *gantry* (que consistem em robôs construídos em forma de ponte, que usam carris para se mover ao longo do eixo horizontal), e os articulados (que simulam as características de um braço robótico e onde todas as juntas são de revolução).

Para cada robô industrial são definidos uma quantidade de *links* e juntas, sendo que a primeira consiste nas partes sólidas de um robô, e a segunda nos acoplamentos móveis entre os *links*. As juntas podem ser divididas em duas frentes: as prismáticas, no qual o movimento é feito apenas linearmente, sendo geralmente actuados por motores eléctricos; as de revolução, que consistem em juntas de rotação accionadas por motores eléctricos e transmissões, ou então por cilindros hidráulicos (Barata, 2012). Uma ilustração de ambos os tipos de juntas está representada na Figura 2.1.

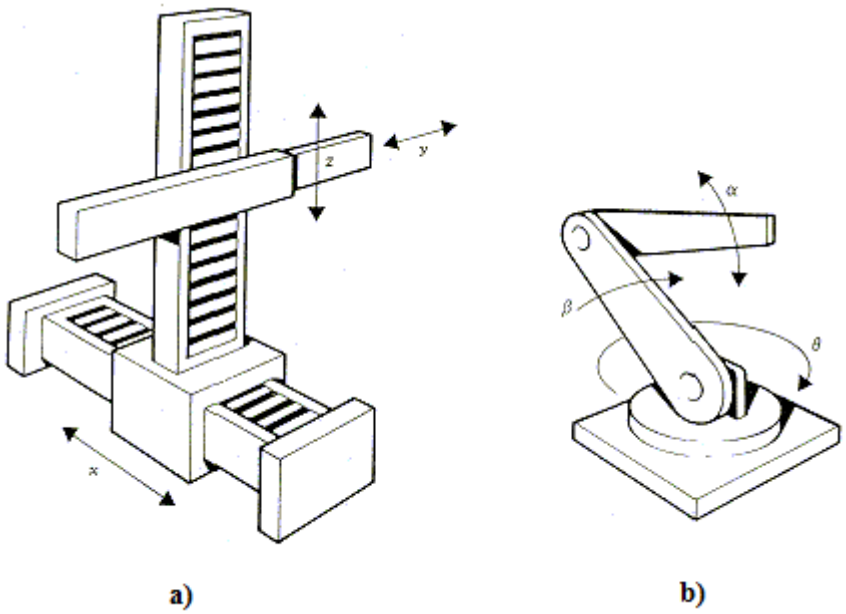

Figura 2.1: a) Representação de juntas prismáticas. **b)** Representação de juntas de revolução. (Barata, 2012).

Além disso, são as juntas que introduzem os graus de liberdade nos robôs, pois cada junta introduz um grau de liberdade. Desta forma, um robô contém *X* graus de liberdade, onde *X* consiste no número de movimentos independentes que consegue realizar; normalmente, os robôs industriais contêm 5 ou 6 graus de liberdade, sendo que três desses são utilizados para posicionamento no espaço 3D, e os restantes dois ou três utilizados para orientação, definindo os ângulos.

Os robôs industriais definem-se também pela sua resolução ou precisão, sendo que consiste no menor dos incrementos de movimento em que um robô pode dividir a sua área de trabalho, dependendo assim, de dois factores: a resolução do sistema de controlo e das imprecisões mecânicas. Assim sendo, é um conceito bastante importante para a área de simulação pois a ferramenta terá de possuir as características do robô a ser simulada o mais próximo possível do real, de modo que os resultados adquiridos pelos vários testes aplicados sejam plausíveis. Os erros de resolução podem ocorrer em diversas situações (Barata, 2012):

- **Erros de Calibração:** A posição determinada na calibração pode estar ligeiramente desajustada, resultando em erros da posição previamente calculada;

- **Erros de Utilização:** Ocorrem devido à elevada utilização do robô específico, normalmente resultado do aquecimento ou gasto do material;

- **Erros de Modelação:** Visto que o modelo cinemático não representa exactamente o robô pretendido, faz com que certos cálculos dos ângulos de junções contenham um certo erro.

Além disto, existe ainda um outro termo importante quanto a robôs industriais, a de repetibilidade. Isto é, o quão perto um dado robô é capaz de repetir determinadas posições previamente estabelecidas, ou seja, consiste no erro de posicionamento quando o robô desloca-se repetidamente para uma mesma posição. Desta maneira, a repetibilidade é apenas um resultado de erros aleatórios, erros estes geralmente inferiores à da resolução ou precisão. De forma a ilustrar melhor uma comparação quanto a estes dois últimos conceitos, apresenta-se a Figura 2.2.

Um último aspecto que importa referir são os modelos de cinemáticas utilizados para movimentar as juntas dos robôs respectivos. A cinemática é a ciência que trata da descrição dos movimentos dos corpos, sem se preocupar com a análise das suas causas. Assim sendo, apenas se restringe quanto à movimentação dos corpos em termos de posicionamento, orientação, velocidade e aceleração (Barata, 2012). A cinemática é, então, dividida em duas frentes: a directa e a inversa.

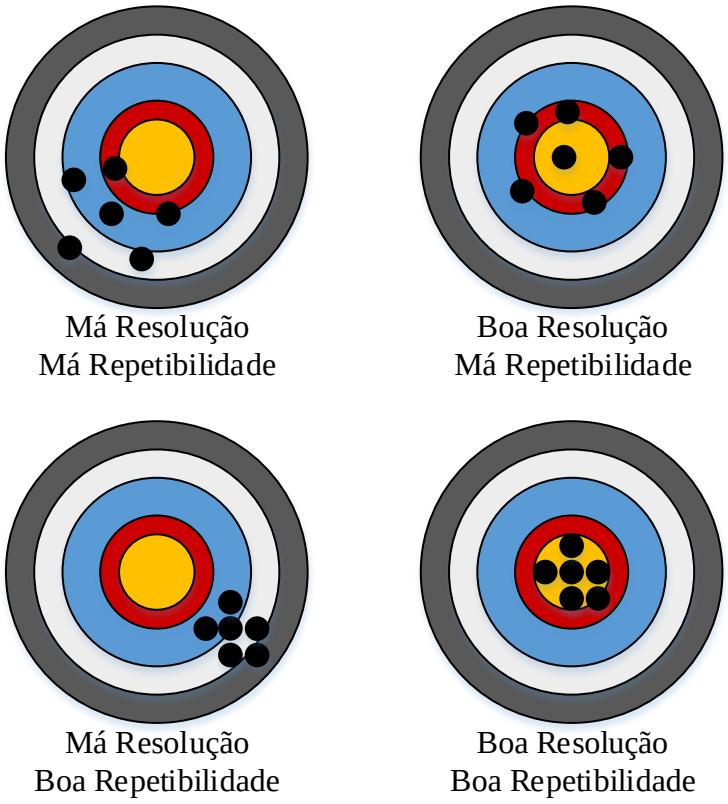

Figura 2.2: Ilustração comparativa entre resolução e repetibilidade.

Começando pela directa, esta consiste no cálculo do posicionamento e orientação de partes articulares, as juntas ou *links* mencionadas anteriormente, de um certo modelo a partir dos ângulos e comprimentos do objecto em questão. Ou seja, a partir das suas posições angulares ($\theta_1, \theta_2, \dots, \theta_n$), chega-se às suas posições identificadas pelas coordenadas cartesianas (x, y, z,) e suas orientações identificadas por coordenadas rotacionais independentes (*roll*, *pitch*, *yaw* - α, β, γ), que são necessárias para descrever qualquer tipo de orientação de um corpo no espaço. Por outro lado, a cinemática inversa consiste no cálculo dos ângulos e comprimentos de partes articuladas de um modelo a partir do posicionamento e orientação desejado em certos pontos desse objecto, resultando numa conversão inversa das coordenadas mencionadas (Kay & Jennifer, 2005). Ou seja, dadas as coordenadas e orientação desejadas, a cinemática inversa permite obter os valores de cada junta, para que o robô consiga movimentar-se para essas coordenadas.

Todos estes cálculos são realizados a partir das chamadas transformadas homogéneas, que consistem em descrições matemáticas onde se calcula o posicionamento e a orientação de uma determinada posição no espaço relativamente a uma outra posição a partir das diversas rotações e translações dos modelos. São, então, utilizados dois esquemas de coordenadas: juntas e mundo. Para uma melhor percepção deste conceito de cinemáticas, fica a seguir representado a Figura 2.3.

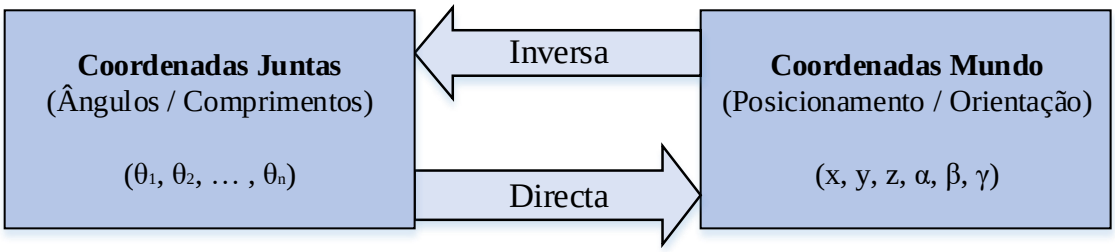

Figura 2.3: Transformação entre coordenadas através de cinemáticas.

2.2 Conceitos de Simulação e Aplicação em Robótica

Por forma a possuir uma maior eficiência, quer em construção para uma maior produtividade e menor consumo de materiais, quer em melhores condições de trabalho e segurança, os robôs industriais foram surgindo com o objectivo de facilitar o trabalho dos vários processos de manufactura, para uma redução dos tempos de produção e custos. Além do mais, certos equipamentos específicos e programas são utilizados nestes processos industriais nas fábricas de produção e manufactura, para um maior ganho de produção.

Desta forma, é típico nas fábricas existirem sistemas reguladores onde toda a informação de sensores é concentrada em um controlador programável no qual, de acordo com o programa em memória, são definidos os estados dos actuadores dos vários robôs industriais. Assim sendo, é típico às funções executadas no dado controlador programável terem uma tendência de serem migradas para os tais instrumentos ou equipamentos de campo, nos robôs. Neste contexto de automação industrial, são controlados diversos actuadores tais como válvulas, actuadores eletrónicos, equipamentos robóticos e indicadores, no qual enviam sinais de informação para a central de controlo, ou seja, é a partir desta central que existe uma contínua comunicação para uma melhor organização dos sistemas e processos (BMA, 2015).

No entanto, essa central também tem um elo de ligação com os sistemas de supervisão, que são uma contribuição adicional que permitem uma partilha de dados importantes da operação diária dos processos, gerando uma maior mobilidade e confiabilidade para os dados que suportam as decisões dentro da empresa a fim de melhorar a produtividade. É nestes sistemas de supervisão que a área da simulação entra em acção dentro da área da manufactura (a simulação é aplicada também em muitos outros contextos de outras áreas de tecnologia, mais a frente mencionados), que surgem com o objectivo de assegurar a inexistência ou diminuição de erros nos processos dos robôs, para não ocorreram acidentes graves durante a produção, o que leva a uma maior qualidade dos processos.

Desta maneira, um simulador é normalmente um programa capaz de representar e reproduzir o comportamento de um determinado sistema, representando fenómenos e ocorrências que na realidade não estão a acontecer no momento, ou seja, pretende reproduzir comportamentos dos equipamentos que pretende simular, permitindo verificar os processos em questão sem existir a necessidade de se gastar materiais, recursos e tempo.

Mais especificamente, para equipamentos de automação robótica, existem certos simuladores que se focam no comportamento de acções e processos de determinados robôs industriais de manufactura, tendo em especial o facto de estes simuladores conseguirem representar objectos no qual o equipamento vai interagir, criando assim "mundos" no qual o modelo poderá funcionar. Além disso, há certos simuladores para sistemas robóticos que podem "ensinar" a partir dos erros que lhes surgem durante a fase de testes, isto é, há certos simuladores que são programados de forma a serem capazes de, após vários comandos lhes serem impostos, analisar e idealizar quais as suas limitações dentro de determinadas acções, sendo que demonstram antropomorficamente a qualidade requerida nos processos analisados (Mack, 2009).

As aplicações mais populares quanto a simuladores robóticos consistem na modelação em 3D e sua renderização do ambiente, sendo que em certos destes sistemas computacionais é também possível mostrar a partir de um motor físico as várias movimentações e acções do equipamento. Certas acções são simuladas através dos sensores do próprio equipamento. Porém, a programação está mais dificultada porque o equipamento depende da informação sensorial que é lida directamente do "mundo" real.

Desta forma, o conceito principal de simulação consiste numa imitação de uma operação ou processo do "mundo real" ou de um sistema de tempo real, ou seja, envolve características de um determinado sistema que esteja a representar (Marietta, 2000). Nos dias que correm, a simulação cresceu de tal forma que é vista como uma das técnicas de investigação operacional mais utilizadas, sendo utilizada em sistemas nas mais diversas áreas tais como redes de telecomunicações, sistemas electrónicos, manufactura, entre outras (Groover, 2007).

Como objectivos principais, a simulação tenta-se focar em pontos de estudo e análise quanto aos comportamentos de determinados sistemas, de forma a poder determinar e verificar a capacidade de cada componente do dado sistema. Desta forma, devido ao estudo aplicado é possível, através da simulação, verificar a eficiência e a produtividade do sistema analisado (Rosas, 2014). Nos sistemas onde a simulação é aplicada, são normalmente definidos por um conjunto de processos organizados e relacionados entre si, que são operados sobre a forma de relações matemáticas e lógicas, gerando os chamados modelos, que consistem em representações abstractas das propriedades e comportamento do sistema em causa. Existem diversos tipos de modelos, nomeadamente os modelos determinísticos, que não possuem qualquer tipo de elementos aleatórios, e os estocásticos, que contêm elementos aleatórios,

seguindo-se pelas probabilidades e variáveis aleatórias. Dentro destes tipos de modelos de sistemas, estes ramificam-se entre modelos estáticos (que representam um sistema num dado instante) ou dinâmicos (que representam alterações de estados ao longo do tempo). Desta forma, os modelos são utilizados para compreender e analisar o funcionamento e diferentes tipos de comportamentos dos sistemas a simular (Rosas, 2014).

Entretanto, um sistema é constituído por entidades e estados, que são subsistemas com funcionalidades independentes, que caracterizam e descrevem um dado momento relativo ao objectivo do sistema, respectivamente. Para complementar, os sistemas podem ainda ser caracterizados entre dois tipos: o discreto e o contínuo. No caso dos sistemas contínuos, as variáveis alteram-se ao longo do tempo, sendo que estão dependentes do tempo; num sistema discreto as variáveis são independentes do tempo, alterando-se apenas de forma instantânea dependendo exclusivamente de determinadas acções, isto é, são sistemas por eventos (Marietta, 2000). Todo este seguimento de caracterização dos modelos dos sistemas está melhor representada na Figura 2.4.

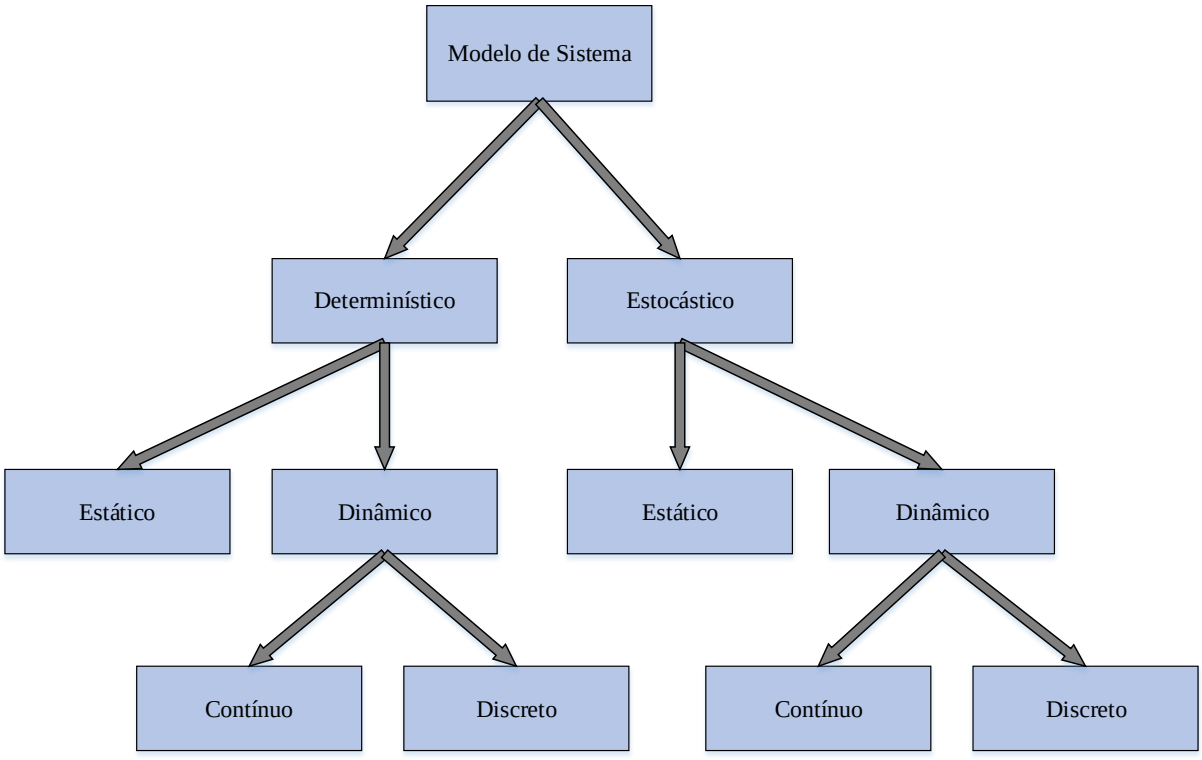

Figura 2.4: Diagrama de modelo de sistema (Rosas, 2014).

Numa simulação, investigações são inicialmente feitas para avaliar os modelos em questão, sendo recolhida informação por forma a poder ser feita uma estimativa quando às acções que se deseja atingir com o modelo. Porém, certos factores limitam o uso de simuladores visto que, por vezes, os modelos utilizados podem ser de dimensões elevadas ou bastante complexos, o que dificulta a criação de sistemas

computacionais capazes de simular tais sistemas, podendo também levar aos limites de capacidades de processamento do sistema utilizado. Entretanto, pode existir uma falta de informação quanto ao modelo do sistema a ser simulado, ou até o facto de a simulação em si ser demasiado dispendiosa para o dado modelo (Rosas, 2014).

Durante a realização de uma simulação, é seguida uma determinada arquitectura. Uma possível arquitectura que costuma ser aplicada foi imposta inicialmente por (Diaz & Behr, 2010), podendo ser visualizada na Figura 2.5. Nesta arquitectura, é possível verificar que, inicialmente, certas acções ou procedimentos são efectuados de acordo com os objectivos ao qual se quer alcançar. Devido às acções aplicadas, certos eventos são disparados, que consistem em ocorrências que mudam o estado do sistema, eventos que usam certas bibliotecas que fornecem ao simulador as bases necessárias ao seu desenvolvimento. Após toda a formulação do modelo em questão, caso se decida que se tem as informações necessárias para o funcionamento do sistema, passa-se a uma fase de relatório de resultados, ao qual se vai analisar as diferentes etapas da simulação realizada.

No entanto, actualmente, certas ferramentas têm sido desenvolvidas de forma a providenciar um auxílio quanto aos recursos necessários para a construção dos modelos e respectiva simulação, e para com os obstáculos referidos anteriormente. Assim sendo, certos simuladores foram analisados, dando especial foco aos de automação robótica, de forma a verificar certos tipos de funcionamento, lógica e processos que podem abordar e interligar com o projecto desta dissertação, ficando-se, assim, com uma breve descrição quanto às suas características e capacidades.

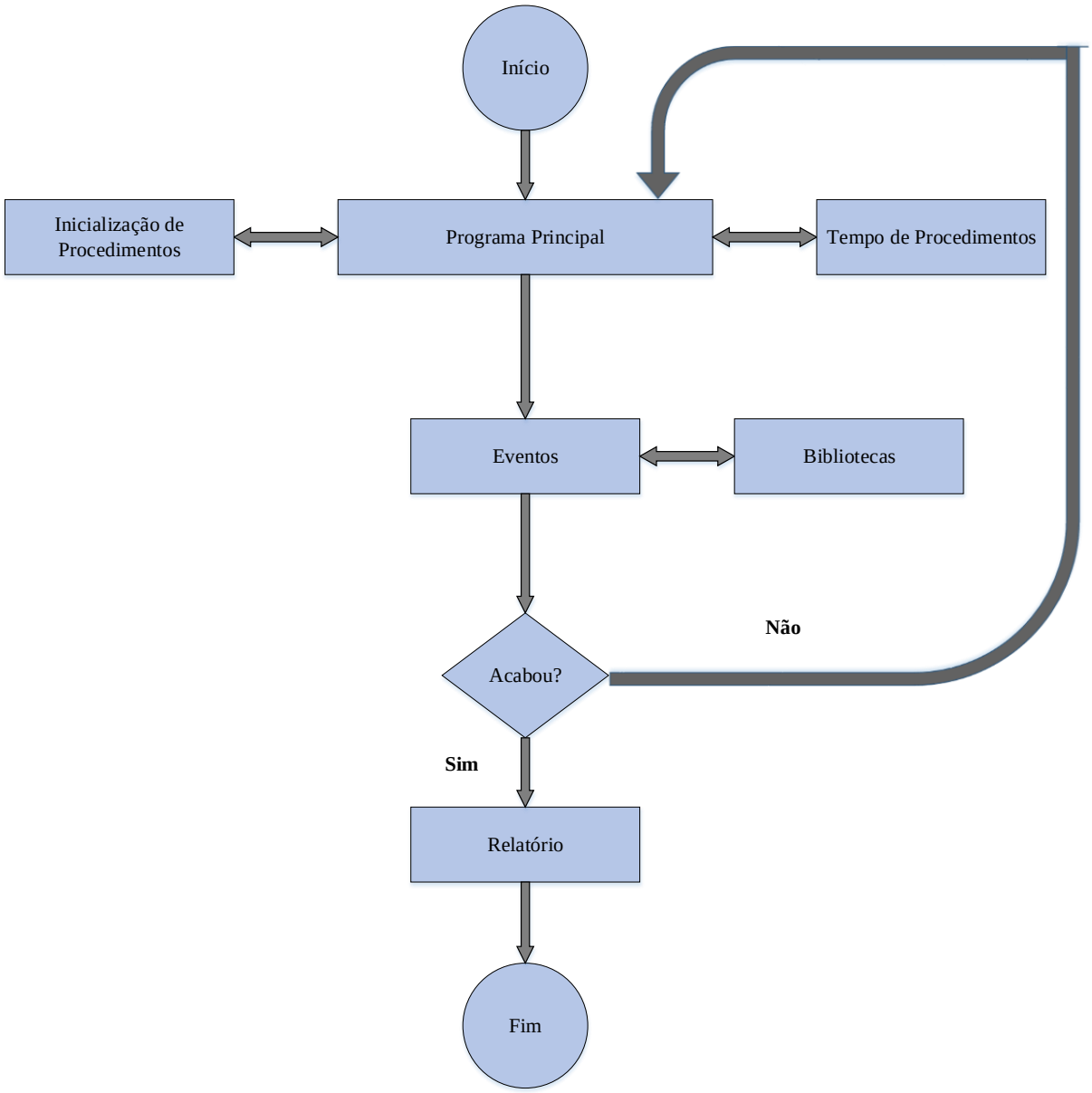

Figura 2.5: Arquitectura de funcionamento de um simulador através de procedimentos e eventos (Diaz & Behr, 2010).

2.2.1 Autonomous Robotic Manipulation (ARM)

Desenvolvido pela DARPA, empresa que tem como objectivo criar um sistema computacional capaz de controlar equipamentos industriais de maneira a conseguirem efectuar processos complexos com supervisão humana. É um simulador representativo do robô desenvolvido pela própria empresa, o robô ARM, desenhado de forma a poder-se desenvolver algoritmos da mesma forma que se poderia realizar no próprio robô, ilustrado na Figura 2.6.

Este sistema computacional tem como principal característica o facto de funcionar em paralelo com o simulador Gazebo, incorporado com o ROS (*Robotic Operating System*), que providencia serviços e ferramentas de controlo e implementação para os processos (ARM, 2015).

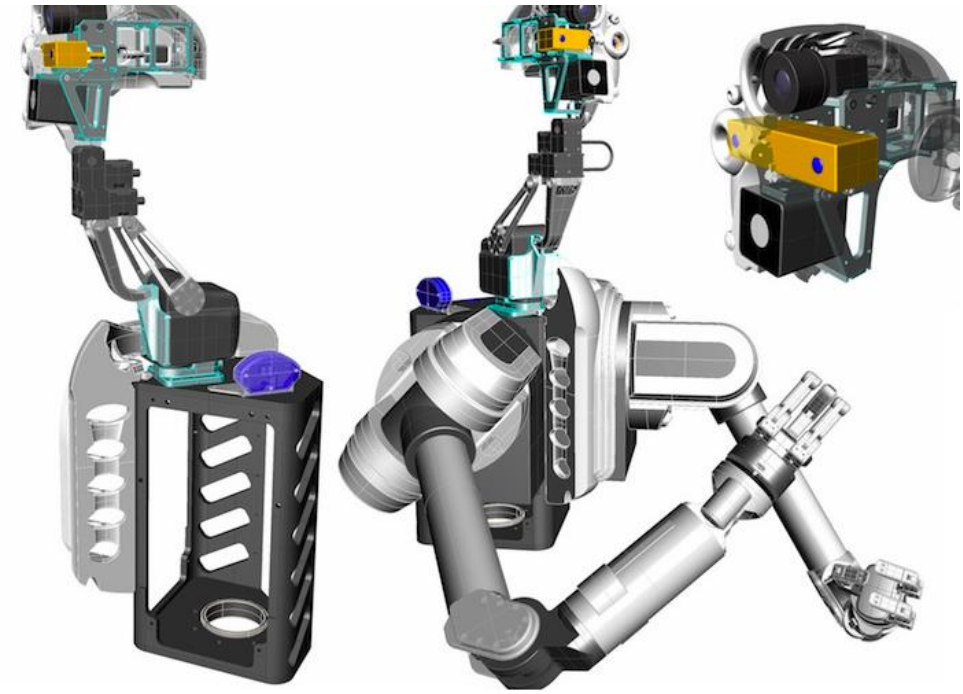

Figura 2.6: Modelo do robô criado para implementação do sistema computacional ARM (ARM, 2015).

2.2.2 RoboAnalyzer

Sistema computacional desenvolvido pela Mechatronics Lab, no Departamento de Engenharia Mecânica no IIT Delhi (*Indian Institute of Technology Delhi*), em Nova Deli, Índia.

Tem como principal objectivo facilitar a aprendizagem em relação às cinemáticas inversas e directas, assim como à essência matemática na automação robótica, sendo algo que os alunos normalmente têm dificuldade de perceber.

Este simulador tem como particularidade o uso de um programa extra intitulado de Redysim (*Recursive Dynamic Simulator*) para algoritmo de cinemáticas, em MatLab, tornando-se assim, num sistema educacional capaz de recriar conceitos robóticos virtualmente usando modelos CAD de robôs industriais, indo ao encontro do controlo de movimentações cartesianas e do conjunto das articulações

do equipamento em causa, ou seja, dos seus graus de liberdade (RoboAnalyzer, 2015). Uma representação do simulador em questão está ilustrada na Figura 2.7.

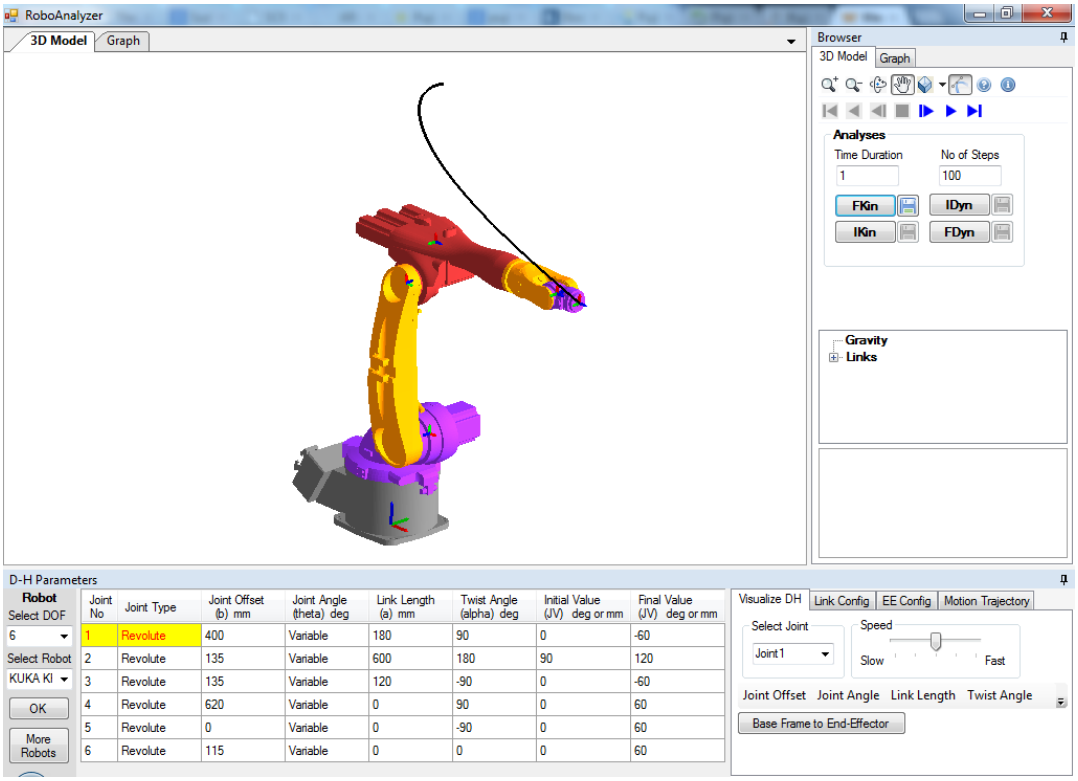

Figura 2.7: Interface de simulação utilizando o sistema computacional RoboAnalyzer (RoboAnalyzer, 2015).

2.2.3 RoboSim

Simulador pertencente à UCDavis C-STEM, que consiste numa companhia que realiza programas de preparação educacional quanto a tecnologias de automação, é um sistema computacional que desenvolve e valida equipamentos de automação robótica virtuais, de forma a poderem ser controlados sem qualquer tipo de modificação no próprio robô, sendo um simulador capaz também de realizar e validar as movimentações possíveis para cada robô (C-STEM , 2015). A linguagem para programar é o C++. Uma ilustração deste simulador em trabalho está representada na Figura 2.8.

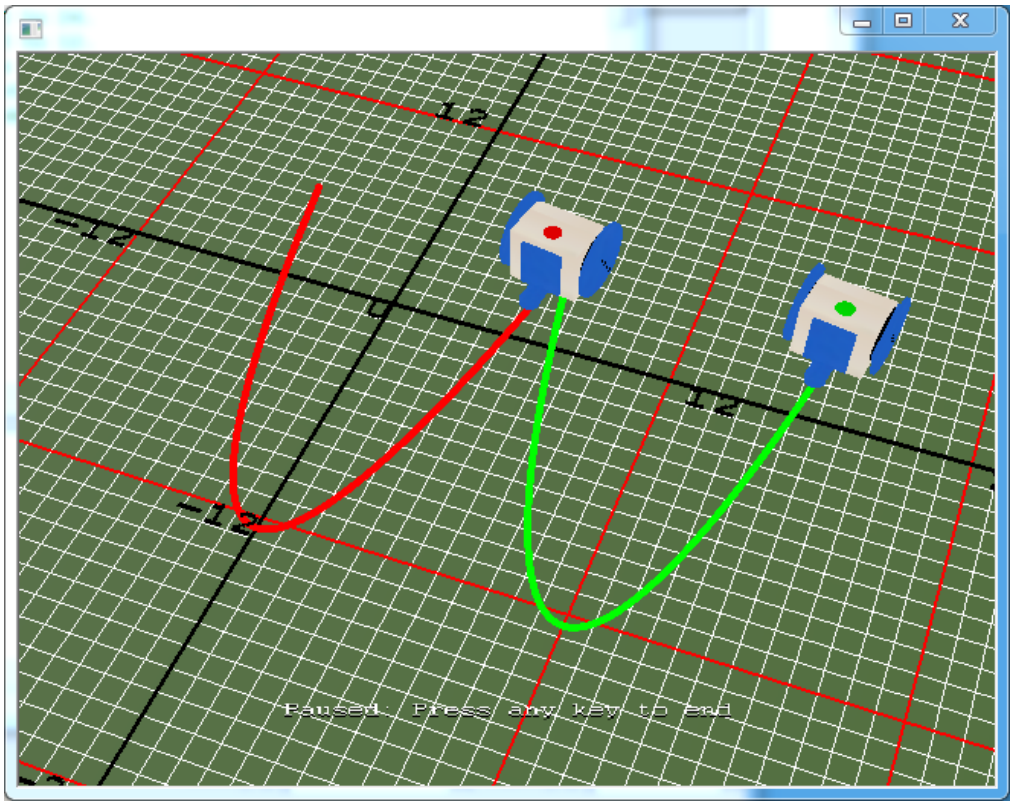

Figura 2.8: Simulação utilizando o sistema computacional RoboSim (C-STEM, 2015).

2.2.4 RobotStudio

Pertencente à ABB (*Asea Brown Boveri*), que é considerada globalmente como a líder em tecnologias de automação, este sistema computacional consiste num simulador de programação "*offline*" que permite programar equipamentos industriais com configurações idênticas às reais sem que a produção em questão seja interrompida (ABB, 2015).

Contém uma série de ferramentas disponíveis para aumentar a rentabilidade do sistema e a sua optimização quanto a redução de riscos, rapidez nos processos e aumento da produção. Utiliza linguagem de programação RAPID para controlo dos sistemas. Uma representação da interface deste simulador está ilustrada na Figura 2.9.

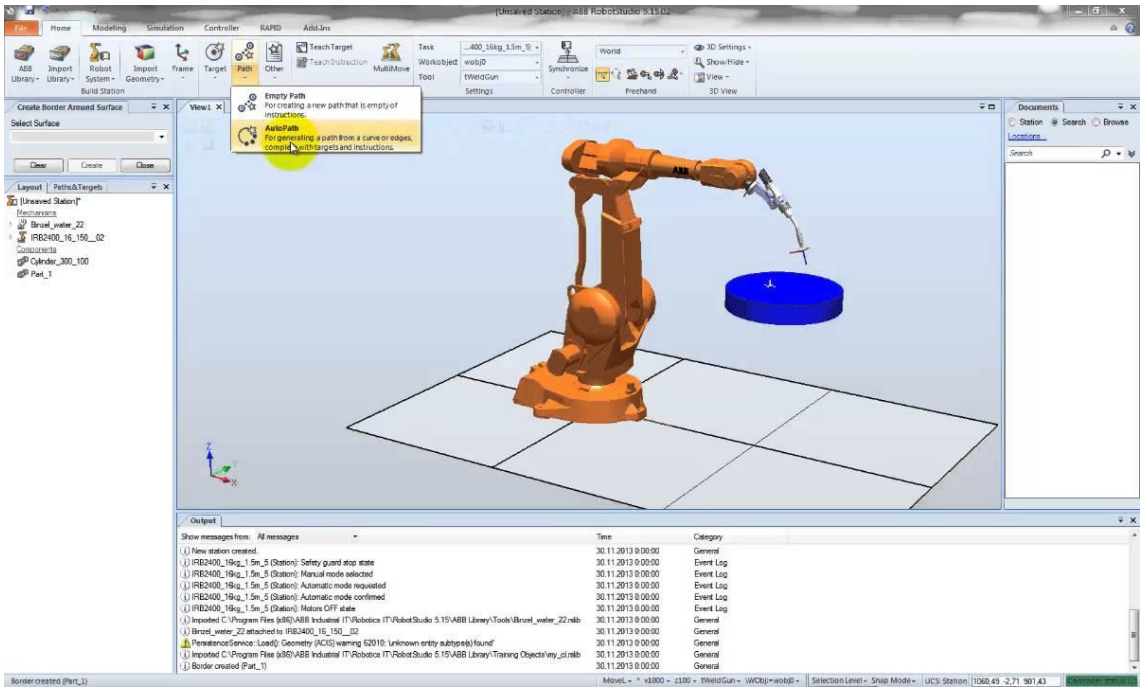

Figura 2.9: Interface de simulação utilizando o sistema computacional RobotStudio (ABB, 2015).

2.2.5 RoKiSim

Pertencente à ETS (*École de Technologie Supérieure*) em Montreal, Canadá, desenvolvido por Albert Nubiola no Laboratório de Controlo e Robótica da instituição, consiste num sistema computacional disponível a todos, servindo apenas para fins educativos (Parallemic, 2015).

É um simulador que consiste no controlo e manuseamento em 3D dos robôs tipo PUMA, com os seus seis eixos, sendo até possível importar objectos para colocar no seu ambiente gráfico para o robô interagir. No entanto, é um simulador que deixou de ser suportado em Janeiro de 2015. A sua interface de simulação está representada na Figura 2.10.

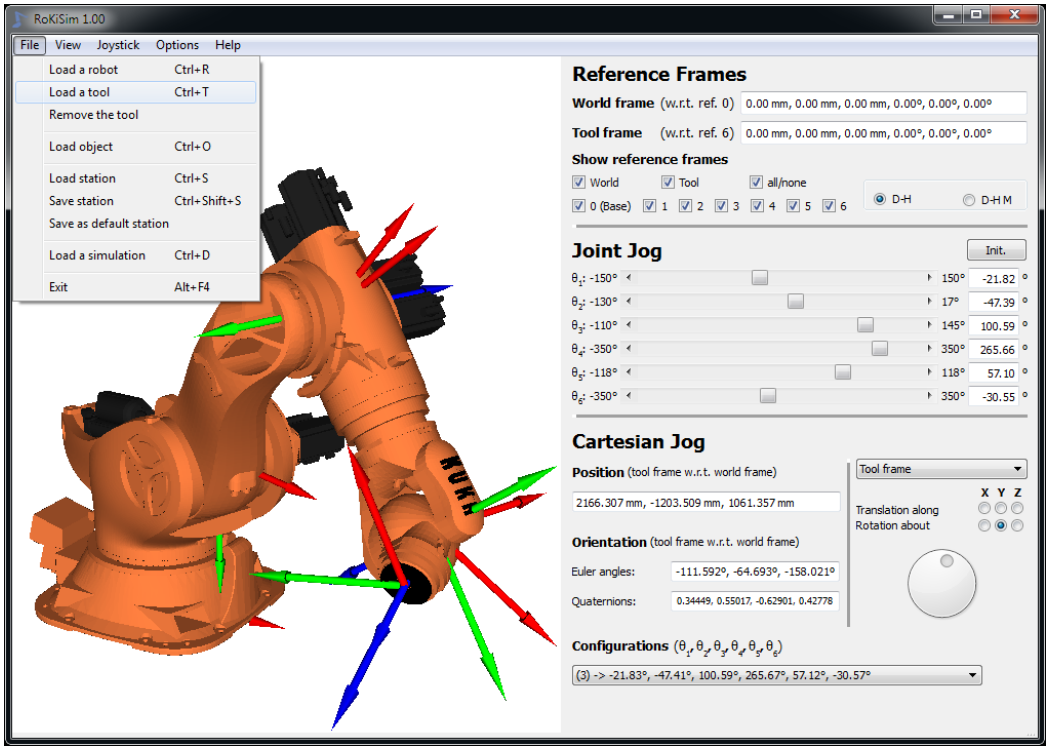

Figura 2.10: Interface de simulação utilizando o sistema computacional RoKiSim (Parallemic, 2015).

2.3 Sistemas Computacionais CAD / Motores de Renderização

Os programas de simulação emergiram e foram desenvolvidos de maneira a que sejam capazes de representar e reproduzir o comportamento de muitos tipos de sistemas. Assim sendo, na interface dos simuladores, os sistemas simulados em causa têm de ser no mínimo perceptíveis ao utilizador quanto ao que representam, por forma a poderem ser feitos os vários testes e estudos. Desta maneira, são requeridos para isso certos programas ou sistemas computacionais com o intuito de modelar e realizar um desenho técnico de robôs industriais capazes de efectuar os modelos para posteriormente serem definidos quanto a renderização e modelação e, de seguida, serem controlados pelo utilizador através dos simuladores.

Estes tais sistemas computacionais são os designados de CAD, ou seja, "*Computer Aided Design*" (ou em português DAC, Desenho Assistido por Computador), no qual fornecem uma série de ferramentas para construção de entidades planas, tais como linhas, curvas e polígonos, ou mesmo objectos tridimensionais como cubos, esferas, etc. Também é possível através destes sistemas relacionar as entidades desenhadas de forma a gerar outros objectos (Farin, Hoschek & Kim, 2002). A origem dos sistemas computacionais CAD advém da modelação de sólidos no qual consiste num leque de princípios matemáticos de modelação computacional tridimensional em sólidos, sendo que a modelação de sólidos

é distinguida em diversas áreas relacionadas com geometria e computação gráfica devido à sua ênfase na fidelidade física.

Embora seja possível também nestes sistemas computacionais recorrer a processos de renderização e animação, normalmente os sistemas CAD não têm como base esses objectivos, sendo que apenas conseguem resolver tais problemas de uma forma mais complexa do habitual e, por vezes, não sendo possível obter os resultados pretendidos. Desta forma, de maneira a chegar a estes determinados objectivos, recorre-se a um específico motor de renderização.

Para um simulador, o código de renderização, que consiste numa transformação pela qual se obtém o produto final de um processamento digital, está especificamente definido pela forma de desenhar um objecto na perfeição, sendo que o motor de renderização fornece materiais que possam ser aplicados em diversos objectos diferentes, tais como texturas, luminosidade, entre outras.

Assim sendo, a escolha do sistema computacional CAD e do motor de renderização é uma questão que se põe para o desenvolvimento de determinado robô industrial, pois o ficheiro do modelo criado terá de possuir bases de movimentação e controlo, de maneira a poder-se definir, através da sua exportação, os vários limites ao qual o robô pode ser então controlado, por forma a poder efectuar os diversos processos requeridos. Para o caso da renderização, tal como já foi referido anteriormente, visto que os sistemas computacionais CAD não se focam em certos aspectos técnicos requeridos para esta dissertação, após a modelação do desenho num sistema CAD apenas será necessário a sua renderização do modelo e a sua preparação de animação para futura simulação, visto que é necessário um foco na definição e textura dos modelos criados.

Entretanto, uma investigação e análise quanto às características e capacidades de certos sistemas computacionais CAD e motores de renderização e gráficos foram realizados, considerando vários aspectos e pontos fulcrais quanto à necessidade do funcionamento requerido para o projecto.

Desta forma, foram consideradas algumas ferramentas utilizadas para a criação de modelos, entre elas o AutoCAD pertencente à *AutoDesk*, o Creo da PTC (*Parametric Technology Corporation*), o SolidWorks da *Dassault Systèmes* e, finalmente, o Wings3D. Quanto a ferramentas para a renderização e animação de modelos foram estudados sistemas tais como o Blender da *Blender Foundation*, e algumas ferramentas da *AutoDesk* como o 3DS MAX e o Maya.

2.3.1 AutoCAD

Programa desenvolvido pela *Autodesk*, é a actual líder de *design* 3D, em sistemas computacionais de engenharia e entretenimento. Este programa consegue realizar modelação para *design* industrial e mecânico, possuindo várias ferramentas e opções ao seu dispor, não possuindo qualquer tipo de foco, no entanto, na renderização e na concepção e manuseamento de materiais (Autodesk, 2015).

Porém, o foco vai todo para o *design*, pois tem uma vasta quantidade de ferramentas e possui boas técnicas de graduação ou escala, isto é, técnicas para traçar e esboçar distâncias e dimensões de maneira proporcional. Tal como todos os produtos da *Autodesk*, possui uma versão para estudantes, tendo porém, algumas limitações; para o produto total, o programa tem de ser pago.

2.3.2 PTC Creo

Pertencente à própria PTC (*Parametric Technology Corporation Solutions*), empresa que produz soluções tecnológicas que apoiam empresas de manufactura, é um sistema computacional que contém bastantes ferramentas de modelação, ferramentas de simulação focadas em cinemáticas e factores humanos, e uma boa validação de processos (PTC, 2015).

Porém, não possui qualquer tipo de focos em renderização e na concepção e manuseamento de materiais, e é também algo complexo a forma de implementação na parte da modelação. Possui uma versão grátis para estudantes, com limitações. Para obtenção do pacote total tem-se de se fazer uma subscrição paga.

2.3.3 SolidWorks

Sistema computacional desenvolvido pela *Dassault Systèmes*, empresa que dispõe de ferramentas de *design* 3D que permitem simular, publicar e organizar a informação dos modelos criados, fornecendo soluções para equipamentos industriais e manufactura. É um programa que consegue criar superfícies e sólidos com facilidade e rapidamente, e também com controlo preciso de medidas e junção de peças para criação de modelos mais complexos (SolidWorks, 2015).

Embora não se foque tanto na renderização, apresenta várias ferramentas de modelação, sendo um pouco complexas as suas implementações, mas fornecendo, no entanto, óptimas técnicas de *design*

para equipamentos industriais e mecânicos, e certas opções de animação. Além disso, tem uma vasta biblioteca de fácil acesso de peças e objectos para serem implementados no ambiente de interface. É possível obter uma demonstração, sendo que, para se obter o pacote geral e total, vem com um certo custo.

2.3.4 Wings3D

Sistema operacional criado por dois programadores suecos, tornou-se num programa que é desenvolvido através do suporte e código feito pela comunidade.

Contém uma vasta biblioteca de ferramentas para modelação, suporte a luminosidade e tipos de materiais, e é capaz de formular texturas nos modelos. É um programa codificado em Erlang, linguagem de programação distribuído pela Ericsson (Wings3D, 2015). Contudo, não é capaz de formular animações e não tem possibilidade de renderização. É, no entanto, um programa grátis ao dispor para utilização.

2.3.5 Blender

Desenvolvido pela *Blender Foundation*, corporação pública alemã, tem como missão criar um sistema computacional grátis para criação de soluções para artistas e pequenas equipas. O programa suporta por inteiro as características essenciais quanto à criação de modelos, seja em modelação, sua renderização e animação, testes de simulação e rastreio de movimentações, concepção e manuseamento de materiais, sendo um sistema computacional que, através da comunidade, evolui pelas contribuições fornecidas (Blender, 2015).

Tem como desvantagem o facto de a interface de trabalho ser pouco intuitiva, embora customizável, o que torna os processos de trabalho algo complexos de serem efectuados, sendo que também não é um programa que se foca na parte de *design* na criação dos modelos. É, contudo, um programa grátis para se usar.

2.3.6 3DS MAX

Pertencente à *Autodesk*, já anteriormente referido, é um programa de renderização, modelação e animação que fornece soluções a artistas de jogos e filmes. Fornece bastantes ferramentas e performance para trabalhar basicamente de igual forma em comparação com o Maya, referenciado a seguir (Autodesk, 2015). A diferença entre os dois sistemas computacionais consiste na forma como se processa certas ferramentas, pois onde um dos programas é mais complexo, o outro costuma ser mais intuitivo, sendo que ambos os programas fazem relativamente o mesmo, mas de maneiras diferentes (Digital-Tutors, 2015).

Para o caso do 3DS MAX, a parte de modelação e simulação é mais facilmente implementada, enquanto a parte de animação e da concepção e manuseamento de materiais é mais complexa; ainda por mais, não possui também o tal foco no *design*. Novamente, como certos outros produtos da Autodesk, possui uma versão grátis para estudante, com limitações.

2.3.7 Maya

Novamente um sistema computacional desenvolvido pela *Autodesk*, é um sistema computacional que fornece ferramentas de renderização, modelação e animação, embora sejam ferramentas de utilização um pouco complexas. Contém um óptimo detalhe de texturização e uma vasta biblioteca ao dispor para simulação, possuindo também uma boa criação do mapeamento da concepção e manuseamento de materiais, embora bastante complexa, pois é necessário alguma programação para a sua utilização (Autodesk, 2015).

Existe uma certa dificuldade na parte de modelação e não possui o foco de *design* dos modelos, sendo um programa mais utilizado para, após importação de modelos, prepará-los para animação e simulação. Tal como quase todos os produtos da *Autodesk*, possui uma versão grátis para estudante, mas tem várias limitações.

2.4 Motores de Jogo

Nos dias que correm, devido ao facto de os sistemas computacionais de motores de jogo conseguirem realizar várias tarefas, o seu conceito (definição e descrição) é algo vago e confuso.

Tal como um motor de renderização, ou qualquer outro motor mais específico, seja de animação, modelação ou simulação, o que distingue um motor de jogo é a sua arquitectura orientada por dados, neste caso em comparação com um programa que seja um jogo e não um motor (Sherrod, 2006). Em mais detalhe, enquanto um jogo tem as suas regras ou lógicas de processos pré-programadas pelo código já efectuado, sendo que apenas consegue aplicar processos em objectos exclusivos, sendo impossível a reutilização desse programa para criação de um novo jogo, um motor de jogo permite efectuar mudanças quanto aos processos nos objectos em causa, permitindo adicionalmente efectuar apenas certas mudanças especificas no código para que os mesmos processos de um jogo sejam realizados em diferentes objectos e em diferentes ambientes.

Assim sendo, o termo motor de jogo está direccionado a um sistema computacional que é extensível e que possa ser utilizado como base de diferentes jogos sem grandes alterações, tratando-se de um programa que suporta uma vasta biblioteca de ferramentas ao dispor, com toda a configuração de controlo de informação, independentemente de ser um jogo 2D ou 3D (Anderson, Engel, Comninos & McLoughlin, 2008). Os jogos criados num motor são, desta maneira, aplicações multimédia em que os dados de entrada chegam ao motor de jogo em variadas formas, desde modelos, imagens representadas em mapas de bits, animações e até ficheiros de vídeo e áudio (Anderson et al., 2008).

Para complementar, visto que os jogos, hoje em dia, são construídos de forma modular, então o motor de jogo refere-se ao conjunto de módulos de simulação de código, sendo estes módulos responsáveis pelas variáveis de entrada, de saída, e pela dinâmica genérica do ambiente do mundo do jogo criado (Gregory, 2009), algo que é possível observar pelo diagrama da Figura 2.11.

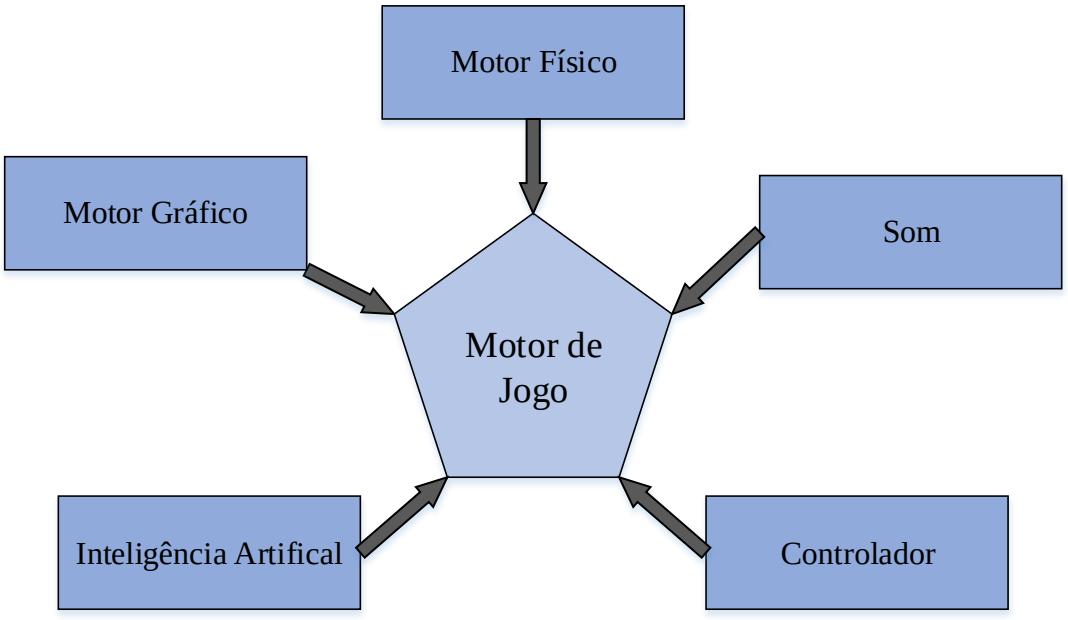

Figura 2.11: Diagrama de estrutura de motores de jogo (Nilson & Söderberg, 2007).

- **Motor Gráfico:** Esta componente é a responsável por efectuar os modelos e suas texturas, assim como o mundo do jogo, por forma a representar algo o mais real possível, efectuando todo o processamento matemático de maneira a que o jogo consiga ser visualizado. Desta forma, é uma parte importante em qualquer jogo pois reflecte a imagem mostrada, sendo uma componente em constante avanço tecnológico pois existe cada vez mais um aumento na complexidade de polígonos utilizados para cada modelo gerado;

- **Motor Físico:** Elemento com um factor cada vez mais importante nos jogos, consiste na parte em que os modelos representados num jogo comportam-se de uma forma realista, através das suas colisões e tipos de movimentos. Esta componente contribui para que o jogo contenha uma experiência mais imersiva, sendo que todos os cálculos de física efectuados consigam fornecer ao jogo uma capacidade tal de executar acções bastante semelhantes às da realidade;

- **Som:** Elemento também importante para qualquer jogo, visto que consiste em reproduzir qualquer ficheiro de áudio que foram impostos, de forma sincronizada de acordo com o que está a decorrer no jogo em determinados momentos. Isto implica num elemento que pode ser interactivo, tornando-o complexo, já que os ficheiros têm de estar prontos na memória para quando os eventos de disparo de reprodução activarem, estes ficheiros reproduzirem automaticamente sem atrasos;

- **Inteligência Artificial:** Componente desenhada por forma a indicar como é que os modelos de um determinado jogo reagem e respondem consoante os comandos implementados, assim como aos modelos não controlados pelo utilizador, no qual têm de tomar decisões por si mesmos de acordo com as situações que lhes vão surgindo;

- **Controlador:** Consiste numa componente mais de interacção para com o utilizador, sendo que a forma como este interage com o jogo possa ser feita de várias formas, através de vários dispositivos. Exemplos de tais dispositivos costumam ser o rato e teclado, ou mesmo até um comando. No entanto, têm vindo a serem desenvolvidas tecnologias capazes de permitir ao utilizador uma interacção através de sensores de movimento ou de pressão.

Assim sendo, um motor de jogo consiste na junção de vários motores, conciliando todo o tipo de dados fornecido por cada um, integrando-se num só, de maneira a ser possível criar jogos bastante

complexos quer em termos de grafismo, banda sonora, físicas e vertentes de controlos e inteligência. Desta forma, chega-se a um consenso de que estes sistemas computacionais contêm várias componentes que estão interligadas e dependentes umas das outras (Busby, Parrish & Wilson, 2009).

Entretanto, uma investigação e análise quanto às características e capacidades de certos motores de jogo foram realizadas, considerando vários aspectos e pontos fulcrais quanto à necessidade do funcionamento requerido para o projecto em questão, mais particularmente na parte de animação. Foram considerados alguns motores de jogo tais como a Microsoft XNA Game Studio da *Microsoft*, o MonoGame, e o Unity Engine da *Unity Technologies*.

2.4.1 Microsoft XNA Game Studio

Criado pela *Microsoft*, líder global de sistemas computacionais, serviços e soluções que ajudam imensas empresas e pessoas a efectuar os seus projectos em todo o seu potencial, o XNA (*XNA's Not Acronymed*) consiste num programa de desenvolvimento de ambientes gráficos que permite a criação de jogos para todas as plataformas da própria Microsoft, podendo-se definir, em paralelo com o Visual Studio, num motor de jogo.

É capaz de realizar, desta forma, ambientes gráficos e suporta como linguagem de programação o C# e o C++ / CLI. Tem como problema o facto de ter sido descontinuado em Abril de 2014, o que faz com que deixe de ter suporte e de obter actualizações (Wiki: Microsoft XNA, 2015).

2.4.2 MonoGame

Desenvolvido pelo programador brasileiro José António Leal de Farias, um membro activo da comunidade da Microsoft XNA que começou por desenvolver um projecto à parte antigamente intitulado de XNA Touch, tinha como objectivo a exportação de simples jogos 2D do XNA para dispositivos móveis (MonoGame, 2015). Tornou-se num projecto aberto à comunidade, evoluindo com as contribuições fornecidas, mudando para o nome de MonoGame, sendo um programa que suporta qualquer sistema operativo, ao contrário do XNA.

Em junção com um ambiente integrado para desenvolvimento de programas, tal como o Visual Studio, por exemplo, torna-se também num motor de jogo que é capaz de realizar vários tipos de ambientes gráficos e criação de jogos. Suporta linguagens de programação C# e qualquer linguagem .NET na *Microsoft*. É, desta forma, visto como uma evolução do próprio XNA, sendo que, como é um

sistema ainda recente e em desenvolvimento, possui alguns problemas que serão resolvidos num futuro próximo.

2.4.3 Unity Engine

Sistema computacional criado pela própria *Unity Technologies*, actual líder global da indústria de jogos, ao qual tem aumentado a sua promoção de jogos complementares e integrados, consiste num motor de jogo que permite a criação rápida de vários modelos e projecções com pouca codificação requerida, com uma vasta biblioteca de ferramentas ao dispor.

Aceita vários tipos de ficheiros para importação e, da codificação necessária, aceita o C#, o Javascript e o Boo (que é um dialecto baseado em Python), sendo que qualquer jogo criado pode ser exportado para basicamente todas as plataformas possíveis. Dependendo da plataforma para qual o jogo é exportado, utiliza como sistemas de renderização o Direct3D ou OpenGL, por exemplo (Creighton, 2010). Contém uma versão grátis para a comunidade, embora com várias limitações; para acesso a todo o leque da biblioteca de ferramentas, subscrições pagas estão ao dispor.

2.5 Síntese

Neste capítulo foi fornecido um enquadramento quanto aos principais tópicos da dissertação, fazendo um resumo das principais características de cada temática, e interligando cada tópico com o problema desta dissertação. É possível verificar também as diferenças de cada sistema computacional, de forma a poder ser feita uma escolha dependente da abordagem a tomar, e das vantagens que existem em cada um, para poder ser desenvolvida uma solução capaz de responder aos requisitos pretendidos.

Tendo em conta o que foi referido anteriormente, apresenta-se as Tabelas 2.1 e 2.2 comparativas com várias características (a primeira focando os sistemas computacionais CAD e motores de renderização, e a segunda os motores de jogo analisados), onde é representado o que cada sistema computacional permite para cada característica de trabalho requerido a partir de níveis de funcionalidade, onde um “certo” significa muito boa potencialidade para determinada característica, e um “errado” o facto de não possuir qualquer tipo de focos nesse mesmo tipo de funcionalidade.

Visto o enquadramento representado quanto às principais características de cada elemento, relacionando-as com o problema deste projecto (já que o projecto a ser desenvolvido tem como finalidade a criação de um programa com fins educativos de vantagens para a área de robótica), o recurso

a estes sistemas computacionais escolhidos permite a utilização de modelos detalhados existentes nas suas bibliotecas, evitando a criação de componentes complexos.

Tabela 2.1: Tabela comparativa de características dos sistemas CAD e motores de renderização.

	Wings 3D	PTC Creo	AutoCAD	Maya	3DS MAX	Blender	SolidWorks
Design	●○○	✓	✓	●○○	●○○	●○○	✓
Modelação	✓	●●○	●○○	●●○	✓	✓	●●○
Concepção de Materiais	●●○	✗	✗	●●○	●●○	✓	✗
Renderização	✗	✗	✗	✓	✓	✓	✗
Animação	✗	●○○	●○○	✓	●●○	●●○	●○○
Simulação	●●○	●●○	●●○	✓	✓	✓	●●○
Licença	Gratuita	Gratuita (limitações) Subscrição	Gratuita (limitações) Subscrição	Gratuita (limitações) Subscrição	Gratuita (limitações) Subscrição	Gratuita	Gratuita (limitações) Subscrição

Tabela 2.2: Tabela comparativa de características de motores de jogo.

	Microsoft XNA	MonoGame	Unity Engine
Sistema Operativo	Windows	Windows, Linux, iOS	Windows, iOS
Linguagens Programação	C#, C++/CLI	C#, C++/CLI	C#, JavaScript, Boo
Multi-Plataformas	Plataformas Windows	✓	✓
Suporte Gráfico	2D e 3D	2D e 3D	2D e 3D
Extensões	✗	✓	✓
Importação Modelos	Apenas ficheiros .x e.fbx	Apenas ficheiros .x e.fbx	✓
Licença	Gratuita	Gratuita	Gratuita (limitações) Subscrição

3

Desenvolvimento da Ferramenta Proposta

Neste capítulo são apresentadas as principais características do simulador proposto, através de uma contextualização dos aspectos mais importantes, que servirá de modelo de base à implementação da solução. Como já foi anteriormente referido, esta ferramenta proposta é focada para um âmbito educativo, sendo então a seguir descritas as diversas funcionalidades do modelo, e representados os diferentes requisitos pretendidos.

Desta forma, é relevante identificar os aspectos teóricos do modelo e as especificações mais técnicas quanto ao simulador e métodos de comunicação, para que se consiga criar uma solução que consiga resolver os problemas identificados.

Através da implementação de este simulador, pretende-se gerar um sistema capaz de simplificar o processo de conhecimento de funcionamento e testes ao robô industrial escolhido, a do braço robótico das versões ROB3 / ROB3i (caracterizado e descrito no subcapítulo 3.3.1), de maneira a serem efectuados o menor número de testes no determinado robô para uma menor probabilidade de falhas e acidentes de trabalho.

3.1 Identificação e Descrição do Problema

Para uma melhor percepção das funcionalidades de robôs industriais num âmbito mais educativo, surge então a criação de uma ferramenta que permita modelar e simular diversas situações recorrendo a modelos desses robôs.

Assim sendo, após a escolha do robô industrial, o braço robótico das versões ROB3 / ROB3i, procede-se à criação de uma ferramenta de simulação. Este simulador será capaz de, através de uma interface acessível, permitir ao utilizador testar e verificar as diferentes acções de movimentação possíveis para o modelo do robô industrial simulado em questão, sem a necessidade de testar no

robô real. Desta forma, não são requeridas ferramentas específicas bastante complexas, sendo suficiente a utilização de uma ferramenta de fácil compreensão e utilização.

Entretanto, devido ao modelo escolhido para simular, é pretendido a realização de uma ilustração o mais próxima possível do real, ou seja, do ambiente a que se está a imitar, pelas várias características do modelo que está a ser aplicado. Esta ilustração terá de ser o mais precisa possível para que, de todos os processos e eventos aplicados, se obtenham dados em que se possa confiar para serem feitas previsões sobre o comportamento do dado sistema real. Desta forma, através dos dados adquiridos, caso sejam favoráveis, é então possível posteriormente realizar os testes nos robôs pretendidos.

Além disto, a realização de uma ferramenta de simulação também irá trazer vantagens quanto ao facto de existir uma diferença entre a natureza dos processos que se pretende simular, isto é, ao simular as diferentes etapas dos vários processos e acções, cada uma destas fases decorrem em diferentes momentos e possuem as suas próprias características, sendo possível retirar dados para analisar individualmente cada fase de funcionamento do sistema. Desta maneira, a criação de um simulador gera uma acessibilidade quanto à aplicação de determinadas acções no modelo a ser analisado, para posteriormente ser possível realizar de forma mais segura os diversos testes no robô industrial.

De forma a representar melhor os problemas identificados dentro da abordagem mencionada anteriormente, fica então uma pequena ilustração mostrada pela Figura 3.1.

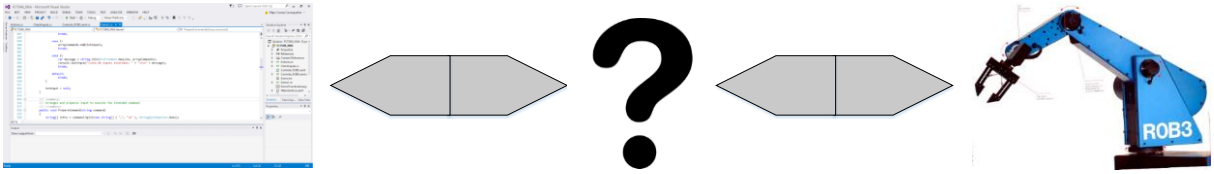

Figura 3.1: Identificação do problema.

3.2 Especificações do Sistema

É pelos requisitos que é possível observar as diferentes especificações escolhidas, sendo também possível analisar os diferentes géneros de funcionalidades do sistema a ser desenvolvido.

Assim sendo, os requisitos de um sistema são uma descrição das funções e suas restrições, que permitem uma compreensão do problema do sistema que se pretende solucionar. É através destes

requisitos que se pode guiar tanto o processo de desenvolvimento de um sistema computacional como também o processo de aquisição (Sommerville, Melnikoff, Arakaki, & de Andrade Barbosa, 2003).

3.2.1 Especificações Funcionais

Os requisitos funcionais são os que definem a forma como um dado sistema deverá reagir quanto às necessidades propostas, pelos comportamentos em situações específicas. Além disso, descrevem as diversas funcionalidades e tipos de serviços que o sistema consegue fornecer e, dependem assim, dos sistemas computacionais utilizados na realização da solução durante a sua implementação.

Com isto, são a seguir representados os requisitos funcionais do projecto desta dissertação através da Tabela 3.1.

Tabela 3.1: Tabela de requisitos funcionais do sistema.

Requisitos Funcionais	Descrição dos Requisitos
RF 01	Visualizar a posição e movimentos do robô simulado de forma similar à do congênere real.
RF 02	Interacção com o robô simulado através do seu protocolo de comunicação, de forma similar ao robô real.
RF 03	Controlar cada eixo / junta do robô utilizando coordenadas de actuadores, de forma similar ao equipamento real.
RF 04	Controlar os comandos robô utilizando cinemática directa e inversa.
RF 05	Permitir testar o funcionamento de programas que utilizam o robô em operações de manufactura.
RF 06	Permitir a aprendizagem na utilização e programação de robôs.

3.2.2 Especificações Técnicas

Os requisitos técnicos ou não funcionais são os que não especificam directamente as funções ou restrições do sistema, visto que simbolizam as características mínimas de qualidade de uma qualquer aplicação ou ferramenta (Sommerville, et al., 2003). Portanto, estão por isso relacionados apenas pelos processos de desenvolvimento e qualidade disponibilizados pelo dado sistema, sendo requisitos que surgem de acordo com a necessidade dos utilizadores. Desta forma, não revelam qualquer função específica a ser realizada na implementação, ao contrário dos requisitos funcionais. No entanto, em caso

de falha do não atendimento a um requisito não funcional, pode levar a que todo o sistema seja incapaz de realizar qualquer função ou processo.

Generalizando, enquanto os requisitos funcionais representam os diversos aspectos e funções a que o sistema deve aplicar, os requisitos técnicos determinam como o sistema deve ser realizado. São então a seguir mencionados os diversos requisitos não funcionais seguidos durante a implementação da solução:

- **Operacionalidade:** A ferramenta irá funcionar em ambiente Windows. No entanto, com as formas de comunicação impostas no sistema será possível controlar o modelo simulado em qualquer aplicação adicional;

- **Desempenho / Resiliência:** O modelo simulado respeitará os limites de movimentação e acções pelas animações de manipulação dos eixos no modo de controlo manual, por forma a ser o mais semelhante e representativo do modelo real. No caso de algum tipo de controlo específico deixar de funcionar ou de responder (seja por falha ou avaria), o sistema deverá ser capaz de manter um desempenho adequado, no sentido de conseguir continuar a realizar as suas operações através dos restantes controlos;

- **Acessibilidade / Eficiência:** O simulador deverá conter uma interface intuitiva, de modo que o utilizador consiga perceber o que consegue aplicar e como realizar as diferentes acções de controlo. Desta forma, o sistema deverá ter um comportamento estável e previsível ao utilizador;

- **Similaridade:** O sistema deverá ser capaz de realizar simulações de um ambiente e ferramenta real, neste caso, de um robô industrial. Assim sendo, o modelo utilizado no simulador deverá ter em atenção as diversas características do sistema real para que este possa ser simulado.

3.3 Arquitectura da Ferramenta de Simulação

Considerando o problema identificado anteriormente para esta dissertação, é então idealizada uma solução proposta, que terá de satisfazer os requisitos funcionais formulados previamente, o que implica

a construção de um sistema capaz de executar as várias tarefas de acordo com os respectivos requisitos pretendidos.

Desta forma, o sistema destina-se a simular um robô industrial, mais especificamente as versões do braço robótico ROB3 / ROB3i (caracterizado no subcapítulo 3.3.1), no qual o utilizador possui várias formas de conseguir controlar o robô em questão dentro da aplicação criada. É possível realizar um controlo manual a partir dos controlos fornecidos pelo próprio simulador, ou então, através de uma de duas comunicações possíveis entre o simulador e uma aplicação exterior: através de serviços WEB REST ou recorrendo a portas virtuais (ambas mais aprofundadas no subcapítulo 3.3.2).

Entretanto, para o caso de o utilizador querer testar a sua aplicação no robô real, o utilizador terá de executar a comunicação através do cabo RS 232 fornecido ao aluno pelo professor (referenciado no subcapítulo 3.3.2), que faz o elo de ligação para com o robô, enviando os comandos requisitados directamente para o equipamento. Toda esta arquitectura é ilustrada na Figura 3.2, mostrando melhor como é seguida a linha de trabalho do sistema.

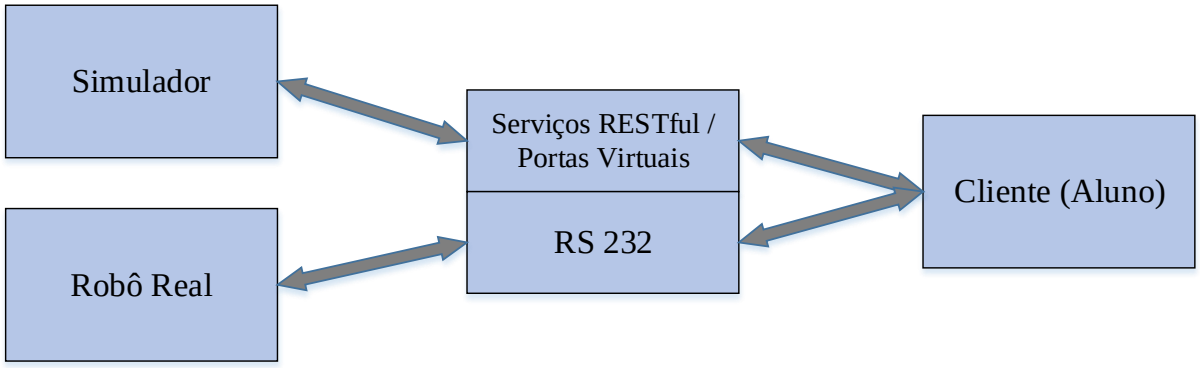

Figura 3.2: Diagrama de arquitectura do sistema.

3.3.1 Arquitectura do Robô e Simulador

Para o projecto desta dissertação foi escolhido o braço robótico das versões **ROB3 / ROB3i** (ambos ilustrados na Figura 3.3) para ser utilizado como modelo no simulador. Consiste num robô mais pequeno em escala e representativo dos equipamentos que são normalmente utilizados em linhas de montagens nos sistemas de manufactura em automação robótica, com o intuito de conseguir realizar

funções onde o desempenho humano não é suficiente, seja em velocidade, precisão, repetibilidade ou durabilidade.

Figura 3.3: a) Representação do robô ROB3 (IPS, 2015). **b)** Representação do robô ROB3i (IPCB, 2015).

Assim sendo, é descrito como um robô articulado, pois este estilo é caracterizado por ser uma representação de um braço humano, contendo 6 graus de liberdade, onde todas as juntas são de revolução. Os seus manipuladores de geometria são antropomórficos, e é sabido que a resolução do equipamento é de 0.625 graus por passo. Os limites de movimentação para cada eixo são representados na Tabela 3.2, onde são indicados os limites para cada peça em ângulos, e velocidades máximas de cada eixo em ângulos por segundo. Todas estas especificações podem ser consultadas no respectivo manual em "*TR5 Robot Manual*" (GmbH, 2002).

Tabela 3.2: Tabela de limites dos eixos do robô (GmbH, 2002).

	Limites	Velocidades
Eixo 1 (Base)	160º	46º/s
Eixo 2 (Ombro)	100º	40º/s
Eixo 3 (Cotovelo)	100º	100º/s
Eixo 4 (Pulso)	200º	174º/s
Eixo 5 (Ferramenta)	200º	176º/s
Eixo 6 (Garra)	60mm	-----

Para finalizar a descrição do robô, é importante referir que, tal como qualquer outro equipamento industrial, o ROB3 / ROB3i está preparado para receber comandos de uma forma específica, sendo que, caso o comando requisitado venha num formato incorrecto ou que implique uma acção fora dos limites impostos pelo robô, pode resultar no equipamento não funcionar, ou de não funcionar correctamente. Isto pode gerar possíveis acidentes, não só para o robô em questão mas, em caso de ser um equipamento de grande escala de uma linha de montagem, causar graves acidentes e trazer prejuízo quanto a custos de equipamentos e atrasos na produção da empresa onde surgiu o acidente.

Desta maneira, de forma a seguir o protocolo de comunicação, para o robô real é suposto enviar-se comandos de acção escritos em apenas formatos de numeração binária ou hexadecimal, sendo algumas das possíveis formas de requisitar comandos as seguintes representadas na Tabela 3.3.

Tabela 3.3: Tabela de comandos para o robô (Klauser, 1990).

Comandos	Comand Byte, data, ETX	Exemplos (R = 0 ou R = 1, a = 010B = 2D ou a = 111B = 7D, ETX = 03H = 3D)	Mensagens de Retorno para Exemplos
Movimentação de um Eixo	0 0 0 0 R a, posição em byte, ETX	0 0 0 0 1 0 1 0, 96H, 03H	R = 0: Não retoma nada. R = 1: 0 0 0 0 1 0 1 0, 03H
Movimentação de Todos os Eixos	0 0 0 0 R 1 1 1, posições em byte, ETX	0 0 0 0 1 1 1 1, 96H, 96H, 96H, 96H, 96H, 96H, 03H	R = 0: Não retoma nada. R = 1: 0 0 0 0 1 1 1 1, 03H
Movimentação de um Eixo com Velocidade	0 1 1 1 R a, posição em byte, velocidade em byte, ETX	0 1 1 1 1 0 1 0, 96H, 0AH, 03H	R = 0: Não retoma nada. R = 1: 0 1 1 1 1 0 1 0, 03H
Movimentação de Todos os Eixos com Velocidade	0 1 1 1 R 1 1 1, posições em byte, velocidades em byte, ETX	0 1 1 1 1 1 1 1, 96H, 96H, 96H, 96H, 96H, 96H, 0AH, 0AH, 0AH, 0AH, 0AH, 0AH, 03H	R = 0: Não retoma nada. R = 1: 0 1 1 1 1 0 1 0, 03H
Leitura de um Eixo	0 1 0 0 R a, ETX	0 1 0 0 0 0 1 0, 03H	0 1 0 0 0 0 1 0, 96H, 03H
Leitura de Todos os Eixos	0 1 0 0 0 1 1 1, ETX	0 1 0 0 0 1 1 1, 03H	0 1 0 0 0 1 1 1, 96H, 96H, 96H, 96H, 96H, 96H, 03H

Desta forma, o simulador implementado, ao realizar uma representação bastante semelhante do robô real, irá ter a possibilidade de receber os comandos pretendidos de acordo com o que é imposto pelo protocolo de comunicação do robô real. Em caso de erro de implementação de comando no simulador, será apresentada uma mensagem de erro.

Para uma melhor ilustração da arquitectura do simulador com o robô modelado, são representados na Figura 3.4 os pontos onde esta ferramenta se foca, representando os seus diferentes aspectos.

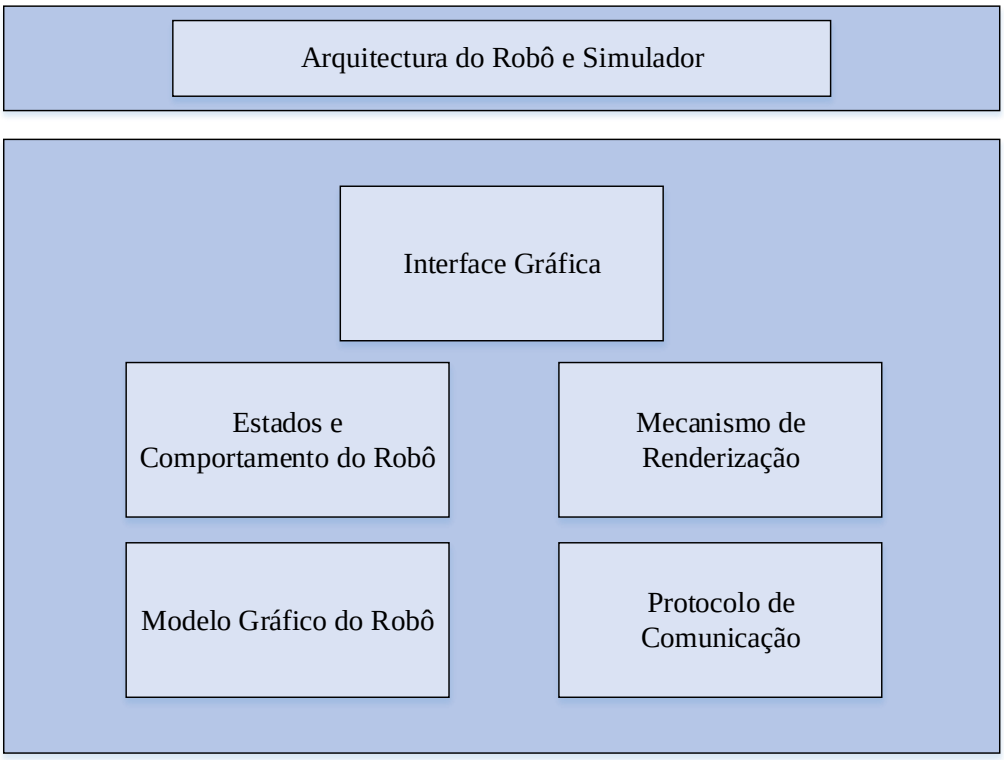

Figura 3.4: Arquitectura do robô e simulador.

Os pontos do modelo do robô industrial e seu protocolo de comunicação já foram caracterizados, onde também foram mencionados os diferentes tipos de limites e comportamentos esperados para o robô industrial, no qual a ferramenta de simulação deverá representar de forma semelhante. Quanto aos aspectos de renderização e interface gráfica, tal como já foi referido neste documento, o modelo terá de ser renderizado dentro do "mundo" gerado pelo motor de jogo, de forma a ser possível efectuar as animações pretendidas. Adicionalmente, o simulador deverá conter uma interface para que o utilizador possua ao seu dispor os possíveis controlos dos vários eixos do robô, assim como as informações necessárias básicas em relação a limites de movimentação do mesmo.

3.3.2 Arquitectura Cliente / Servidor

Após ter conhecimento do modelo a ser simulado, dos seus possíveis limites de movimentação e tipos de comandos que consegue receber, falta perceber melhor sobre os métodos de comunicação possíveis entre uma aplicação exterior e o simulador criado, para um possível controlo do modelo simulado. Caso seja necessário, um controlo manual feito através do próprio simulador está também disponível ao utilizador como opção de testes de implementação.

Assim sendo, é necessário existir uma comunicação com o simulador capaz de representar, efectivamente, a ligação real com a porta série do robô industrial intitulada de *COM Port / RS-232*. Uma aplicação exterior num computador conectado ao equipamento através desta ligação é capaz de receber informação através da porta série, que será processada de acordo com a lógica guardada na base de dados, sendo informação capaz de realizar e gerar, por exemplo, gráficos, relatórios ou tabelas (Electronic Industries Association, 1969).

Desta forma, visto ser impossível modificar a forma como o equipamento recebe os comandos, ou ainda menos modificar o robô em si, de forma a resolver o problema de estar sempre conectado através do cabo série, chegou-se a duas formas de permitir uma transmissão de dados entre uma aplicação e o simulador. Começando então pelos serviços WEB (*Wide World Web*) REST (*Representational State Transfer*), este tipo de comunicação consiste num estilo de arquitectura desenhado para sistemas distribuídos, existindo uma relação de cliente / servidor (neste caso aluno / simulador), contendo uma interface uniforme entre componentes (Fielding, 2000). Contém várias vantagens em relação a outras arquitecturas, isto porque oferece benefícios de interoperabilidade, evolução independente e eficiente, e um desempenho geral melhorada. Além disso, os serviços REST não estão apenas relacionados por HTTP (*HyperText Transfer Protocol*), embora sejam normalmente associados a tal (MSDN, 2015).

O facto de possuir a interface uniforme permite, ao existir uma interface generalizada entre componentes, fornecer ao sistema em que está a ser implementado uma forma mais simples de interacções e comunicações entre várias aplicações, podendo identificar recursos e manipulá-los através de representações, nomeadamente JSON (*JavaScript Object Notation*) ou XML (*eXtensible Markup Language*), que representam informação de objectos e seus atributos (MSDN, 2015). Quando a arquitectura implementada consegue seguir todas estas componentes ao qual REST é caracterizado, é normalmente referido aos serviços como **"**RESTful**".**

Assim sendo, esta arquitectura é uma das possíveis a ser utilizadas para comunicação entre uma aplicação exterior e o simulador criado, sendo que são introduzidos e enviados, pela aplicação do utilizador, os diversos comandos de controlo do robô, para depois serem recebidos e tratados pelo servidor, neste caso o simulador, existindo hipótese de resposta de certos dados por parte do simulador para com o utilizador. Estas mensagens transmitidas pelos serviços REST usam explicitamente métodos HTTP, tais como GET, POST, PUT e DELETE, para criar, adicionar, modificar e apagar, respectivamente. Alguns destes métodos de comunicação são caracterizados a seguir na Tabela 3.4, onde são apenas ilustrados os mais comuns.

Tabela 3.4: Tabela de métodos de comunicação de serviços REST (MSDN, 2008).

Métodos	Descrição	Segurança
GET	Requisita uma representação de um recurso.	Sim
PUT	Cria ou modifica um recurso com a representação requerida.	Não
DELETE	Apaga um específico recurso.	Não
POST	Implementa informação para ser processada por um recurso identificado.	Não
HEAD	Semelhante ao GET, apenas retorna os cabeçalhos e não o corpo da representação de um recurso.	Sim
OPTIONS	Retorna os métodos suportados pelo recurso identificado.	Sim

Por fim, o último tipo de comunicação possível entre o utilizador e o simulador consiste nas portas virtuais, que são uma representação mais fiel de comunicação em termos de funcionamento em relação à porta série *RS-232*. Têm como vantagem o facto de permitir um número de ligações ou conexões virtuais ilimitados, assim como distância ilimitada entre as conexões virtuais (CodeProject, 2008).

Ainda por mais, este método é uma aproximação de comunicações virtuais muito mais simplificadas para com por exemplo, os serviços REST acima descritos, conseguindo para o projecto desta dissertação também realizar as várias comunicações entre um utilizador e uma aplicação extra para com o simulador implementado. No entanto, para fins mais aprofundados à área de serviços WEB, as portas virtuais em comparação aos serviços REST já não conseguem acompanhar as várias funcionalidades requisitadas, servindo apenas como substituto ao cabo série normalmente utilizado com os robôs industriais. Visto que os serviços REST serem um pouco mais complexos que as portas virtuais devido ao seu estilo de comunicação entre cliente e servidor, está então de seguida representado uma ilustração desta arquitectura de comunicação na Figura 3.5.

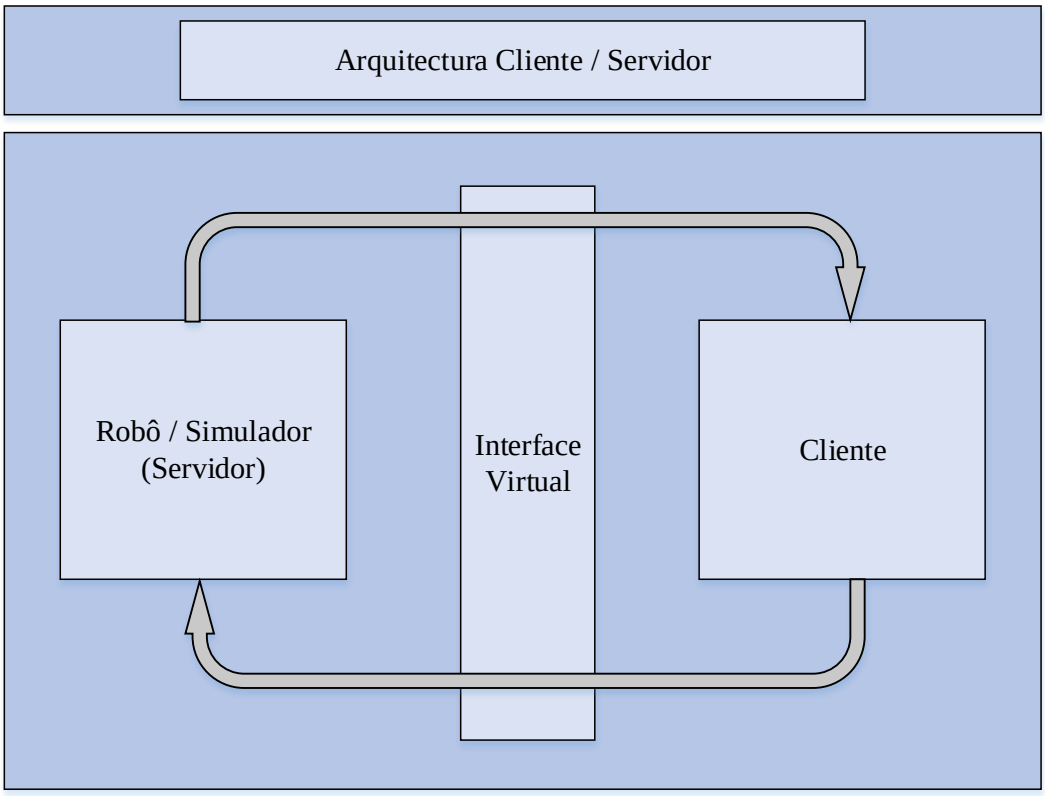

Figura 3.5: Arquitectura cliente / servidor.

Desta maneira, a comunicação entre cliente e servidor é feita sempre entre uma interface virtual onde vão ser definidos os meios de comunicação entre duas posições remotas, definições estas acima descritas. O servidor consiste na arquitectura referenciada no subcapítulo 3.3.1, constituído pelo robô e simulador. Quando ao cliente, este consiste numa aplicação desenvolvida pelos utilizadores do robô para modelarem os programas de controlo do robô, assim como procederem à respectiva simulação, sendo que, após verificados os comandos enviados ao simulador, em caso de bom funcionamento do modelo em questão na ferramenta, podem ser então colocados em modo de produção no robô real.

3.4 Implementação do Simulador

Para esta secção da dissertação são analisados quais os tipos de sistemas computacionais escolhidos para a criação do sistema, assim como os vários métodos e passos aplicados para a implementação da solução, para consequente validação.

Desta maneira, para a realização deste projecto, é verificado neste capítulo que todos os conhecimentos investigados e obtidos que foram mencionados nos anteriores capítulos são postos em

prática, com o fim de garantir os vários requisitos fundamentais pretendidos e vários tipos de funcionamento.

3.4.1 Cenário de Utilização do Simulador

A forma como o utilizador será capaz de explorar o sistema ou solução proposta é algo importante a mencionar, pois é neste cenário que o utilizador terá o conhecimento de como o sistema é suposto comportar.

Assim sendo, tal como se pôde verificar na Figura 3.2 e como já foi mencionado anteriormente, o utilizador (nomeadamente o aluno ou professor), terá duas formas de executar os comandos pretendidos: ou através de serviços WEB REST / portas virtuais (já detalhados no anterior subcapítulo), onde os comandos requisitados serão recebidos pela solução proposta, o simulador (onde esta dissertação se foca); ou através do cabo de ligação RS 232 fornecido ao aluno pelo professor (também já acima descrito), no qual vai fornecer o elo de ligação para com o robô real, implementando os comandos pretendidos imediatamente na máquina em questão.

Desta forma, tendo em conta este cenário explorado que vai existir entre o simulador / robô real e o utilizador, para se conseguir chegar à solução proposta, é necessário garantir um número de funcionalidades em diferentes etapas da implementação. Entretanto, partes destas etapas funcionam de forma independente das restantes, contendo funções que são apenas processadas localmente; no entanto, quando integrados formam o sistema global descrito e pretendido.

São então apresentadas de seguida as diferentes etapas da implementação, sendo possível dividi-las em três partes:

- **Modelação**: onde foi desenhado e construído num sistema CAD o modelo do robô industrial em questão;

- **Renderização**: consistindo na transformação pela qual se obtém o produto final de um processamento digital, é onde foi implementado o "esqueleto" do modelo para que as animações fossem criadas e consequente texturas;

- **Programação**: secção onde toda a lógica em como o simulador funciona se encontra.

Cada parte será explicada nos subcapítulos 3.4.3 e 3.4.4 em mais detalhe, aprofundando certos aspectos técnicos e métodos de implementação.

3.4.2 Factores de Escolha e Opções Tomadas

De toda a investigação realizada e descrita anteriormente dos vários sistemas computacionais, teve-se de realizar certas escolhas quanto aos sistemas computacionais a serem utilizados para cada fase do projecto, sendo que se teve de analisar os vários pontos fortes de cada sistema computacional, e compará-los entre si para se observar onde se encontravam mais vantagens para a sua utilização.

Desta forma, visto que para o projecto desta dissertação teve de se inicializar a partir da criação dos modelos de equipamentos industriais, passa-se à escolha de um sistema computacional CAD, ao qual foi escolhido o SolidWorks.

Tal como foi possível observar na Tabela 2.1, e como foi também referido, o SolidWorks é um programa com uma vasta biblioteca de ferramentas de *design* e modelação, fornecendo adicionalmente certas opções de animação. Ainda por mais, um sistema CAD como este, tem o intuito de desenhar especificamente equipamentos para funções a nível industrial, mecânico, de arquitectura, e até para a área de engenharia espacial, ou seja, é um programa que foca-se essencialmente em técnicas de escala, dimensionamentos e precisão, visto que os equipamentos criados têm de ser precisos para posteriormente serem utilizados para produção, construção e, claro está, testes de simulação (SolidWorks, 2015).

A escolha do sistema CAD não foi realizada com facilidade, visto que sistemas tais como o PTC Creo e o AutoCAD são também bastante eficientes nos seus processos, porém, em relação ao AutoCAD, este perde um pouco na parte da modelação e animação, apenas por ser mais complexa a forma e estrutura dos processos, embora consiga fazer o mesmo. Quanto ao PTC Creo, este é um programa muito semelhante ao SolidWorks, sendo que a escolha aqui feita ter sido devido ao facto de, durante a investigação efectuada de sistemas CAD, o PTC Creo ter sido dos últimos, e já tinha sido efectuado um avanço do projecto, tornando-se numa escolha um pouco pessoal pois já tinha sido obtido alguma experiência no SolidWorks. Em relação ao Wings3D, ainda consiste num sistema computacional CAD algo primitivo pois não oferece tantas ferramentas como os outros, focando-se apenas na modelação e na concepção e manuseamento de materiais.

Após realizados os vários modelos dos robôs industriais requeridos para o projecto, é necessário então a renderização e animação destes, para que no simulador consigam efectuar as várias acções

pretendidas. Entretanto, passou-se à escolha de um motor de renderização capaz de efectuar estes processos, sendo que a escolha realizada foi o Blender.

Novamente, voltando à Tabela 2.1 do capítulo 2, o Blender é um sistema computacional com óptimas técnicas essenciais de concepção e manuseamento de materiais, renderização e animação, sendo que fornece adicionalmente bastantes opções de modelação e simulação. Um motor de renderização como o Blender foca-se mais em recriar um "mundo" de raiz, e de seguida animá-lo o mais simples possível com os vários modelos, ou seja, é um programa com um objectivo de modelação e animação virado para um lado mais criativo, com um variado leque de formas e texturas, conseguindo até gerar animações e interacções entre vários objectos do tal "mundo" criado. Assim sendo, um motor de renderização contém o foco total na dada renderização dos objectos, com texturas muito bem detalhadas, podendo recriar acções tais como o cair de folhas das árvores, do cabelo ao vento, e até da água a movimentar-se entre as rochas num rio, por exemplo, sendo o objectivo principal o de se obter algo que seja o mais interessante possível em termos visuais.

Para a escolha deste sistema computacional, o Blender foi comparado com o Maya e o 3DS MAX, sendo que, embora perca um pouco nas ferramentas de animação, apenas pela maneira de se realizar os processos de uma forma mais complexa, na questão da modelação e na de concepção e manuseamento de materiais, o Blender contém alguma acessibilidade em relação ao Maya e 3DS MAX, e mais ferramentas ao dispor (Digital-Tutors, 2015). Ainda por mais, a licença além de completamente gratuita, é um sistema que vai evoluindo através da contribuição feita pela comunidade, enquanto a licença nos outros dois sistemas é gratuita mas com limitações, porque para obtenção do pacote total de ferramentas ao dispor, existe uma subscrição paga.

Finalmente, após os modelos serem criados e posteriormente renderizados, passa-se à fase final de controlo das várias acções requeridas, sendo necessário um motor de jogo para tal, ao qual foi escolhido o MonoGame.

Com base na investigação feita, como pode ser observado na Tabela 2.2 no capítulo 2, o MonoGame é um sistema computacional que vai evoluindo através da contribuição feita pela comunidade, ao contrário da Microsoft XNA que foi completamente descontinuado, sendo possível a sua utilização em qualquer sistema operativo, e fornecendo exportação de jogos (quer 2D quer 3D) para qualquer plataforma, o que mostra uma capacidade de acessibilidade enorme. Comparativamente com o Unity Engine, perde em relação à importação de modelos CAD, visto que apenas suporta ficheiros .x e .fbx (tal como o XNA), e o Unity importa vários tipos de ficheiros. No entanto, o MonoGame é completamente gratuito, enquanto o Unity, para se obter o pacote total de ferramentas ao dispor, é necessário uma subscrição paga.

Após esta análise, o problema foi abordado de forma a seguir os passos de forma ordenada, sendo que, primeiramente, começou-se pela tal criação do modelo da máquina industrial, seguido da sua renderização e, por fim, do seu controlo para o simulador estar operacional.

3.4.3 Modelação e Renderização

Face ao que já foi descrito previamente, para a realização do simulador desejado iniciou-se pela modelação do modelo do equipamento industrial em SolidWorks. Nesta etapa inicial, efectuou-se uma modelação individual de cada peça do braço robótico, tendo-se juntado todas estas peças no fim para chegar ao modelo esperado, a do braço robótico ROB3 / ROB3i.

Todas as peças do modelo estão o mais próximo possível da realidade, visto que as dimensões utilizadas estão de acordo com as do modelo real, tendo-se seguido também a maioria do estilo de *design* que o braço robótico apresenta, nomeadamente a da versão do ROB3, dimensões essas ilustradas na Figura 3.6 a seguir representada.

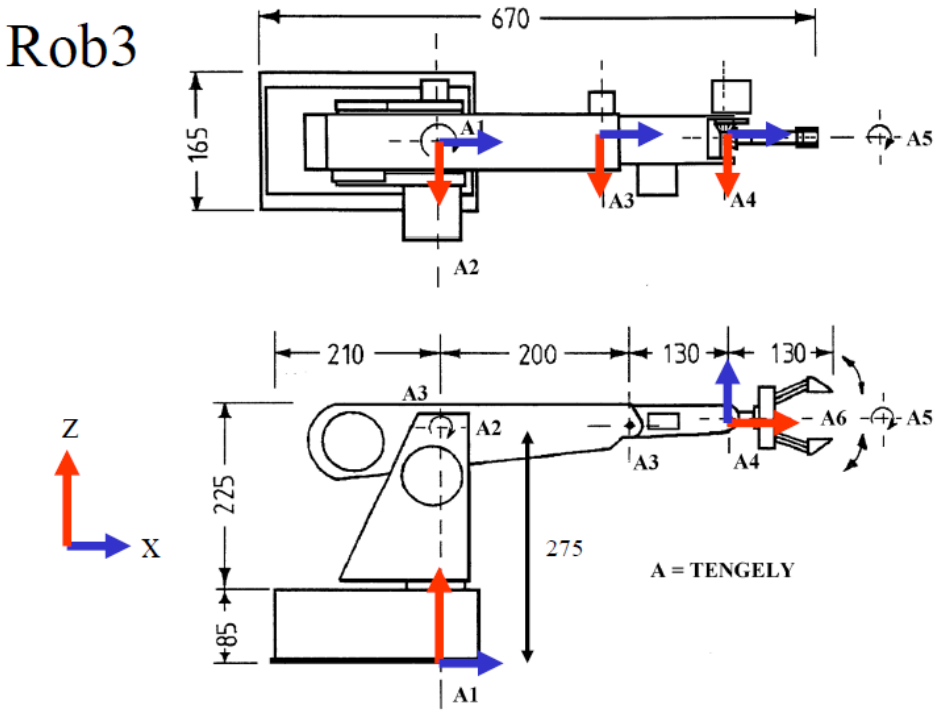

Figura 3.6: Representação das dimensões para a versão do braço robótico ROB3 (Barata, 2012) (GmbH, 2002).

Desta maneira, o robô modelado é bastante semelhante ao modelo real, tal como está representado na Figura 3.7. Após a junção de todas as peças para a criação do modelo em si por inteiro, foram

implementados tipos de movimentação e respectivos limites para os diferentes eixos, para que o modelo fosse futuramente controlado a efectuar as animações de forma correcta. No entanto, chegou-se à conclusão que o motor de jogo MonoGame não consegue aceitar os tipos de ficheiro que o sistema computacional SolidWorks salva. Isto gerou um problema, pois além de o ficheiro ter de ser convertido para um dos possíveis ao motor de jogo receber, implicou a que, durante a conversão, as animações previamente feitas no SolidWorks fossem perdidas, além de o modelo em si ter de ser renderizado, algo que também foi notado de que tinha de ser realizado para que as animações funcionassem correctamente no MonoGame.

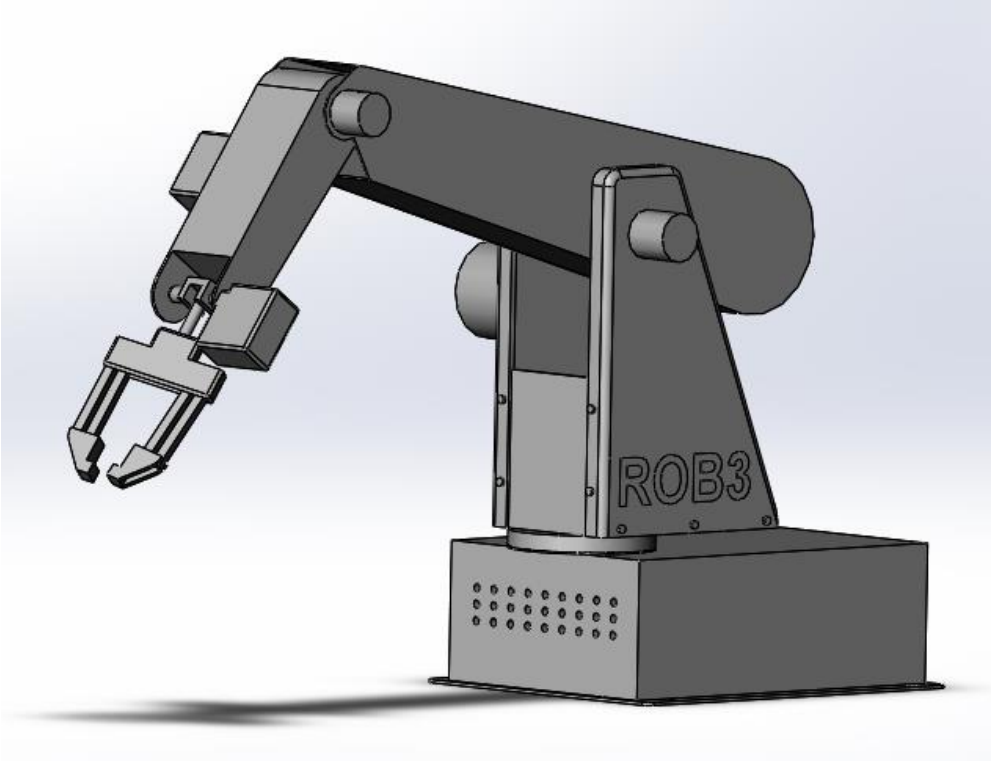

Figura 3.7: Fase de modelação. Representação do modelo do robô em SolidWorks.

Com isto, teve de se trocar de sistema computacional, sendo que o escolhido foi o Blender, isto porque oferece ferramentas capazes de realizar ambas as tarefas necessárias ao momento: as animações e a renderização do modelo. Adicionalmente, o Blender também oferece a possibilidade de realizar uma conversão de .stl (tipo de ficheiro gravado pelo Solidworks) para .fbx (tipo de ficheiro de possível recepção para o MonoGame). Assim sendo, passou-se para uma nova fase, onde foi idealizado um esqueleto do modelo do braço robótico, um osso por peça, gerando animações o mais próximos possível do robô real, que fossem depois perceptíveis para o motor de jogo.

Além disso, foi inicializada uma tarefa intitulada de "*Skinning*" por "*UV Unwrapping*" (tarefa representada na Figura 3.8), no qual vai delimitar os vértices pretendidos de uma determinada peça para o osso idealizado de essa mesma peça, isto para prevenir que o modelo tenha defeitos ou anomalias durante a movimentação do mesmo, isto é, que sofra deformações nas peças respectivas. Esta tarefa merece especial atenção, pois embora tenha o intuito de prevenir tais deformações de acontecerem no modelo durante o seu controlo, caso a escolha de vértices seja mal idealizada para o osso em questão, ou o facto de algum vértice importante escapar durante a sua etapa de escolha, isto pode originar também os tais defeitos durante a fase de movimentações. Visto que o modelo é bastante semelhante ao modelo real, dificultou ainda mais esta tarefa: quanto mais próximo do real um modelo desenhado num sistema CAD é, mais polígonos são gerados, o que leva a que, quantos mais polígonos existem num modelo, mais vértices existem.

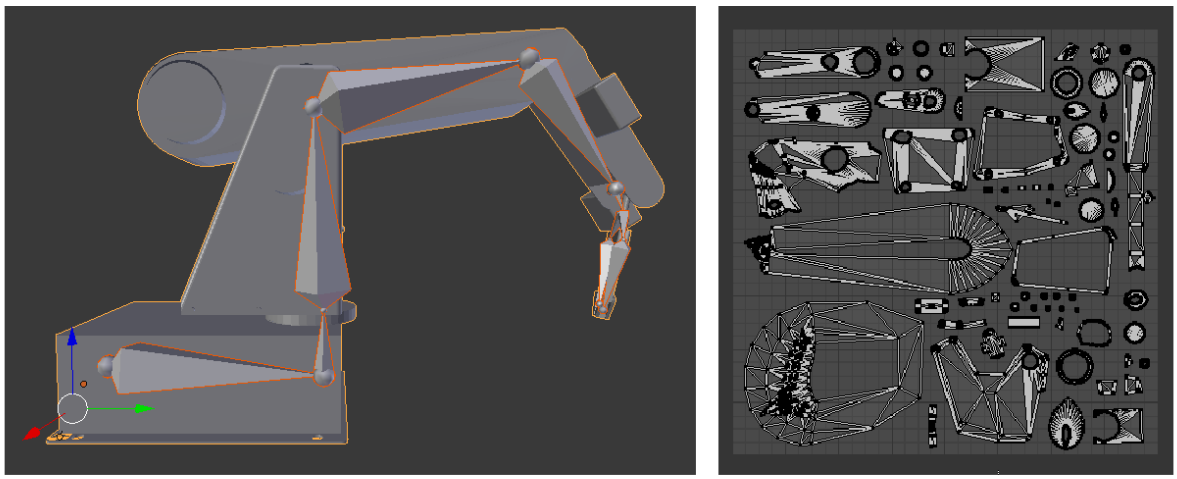

Figura 3.8: Fase de renderização. **a)** Representação do modelo do robô com esqueleto em Blender. **b)** Representação das peças do modelo delimitadas para animações (*"Skinning"*).

Adicionalmente, após alguns testes terem sido feitos no robô simulado durante a fase seguinte de programação (detalhado no subcapítulo 3.4.4), notou-se que teria de se recolocar o robô modelado num ponto de origem diferente ao que ele contém inicialmente quando é importado no Blender vindo do SolidWorks, visto que certas animações em determinados eixos do robô não conseguiam obedecer às orientações das movimentações correctamente. Isto acontece também devido ao facto de, após um modelo ser exportado pelo Blender, a orientação dos eixos serem diferentes para com os do MonoGame, o que também dificultou nas animações.

Esta fase de renderização é algo que foi sempre repetida várias vezes durante a implementação, isto devido a certas deformações por vezes aparecerem no simulador no motor de jogo, tendo que ser revisitada a tarefa de criação de ossos e escolha de vértices múltiplas vezes. Entretanto, após a fase de

modelação e posterior renderização do modelo pretendido, o ficheiro gravado pelo Blender do modelo é exportado para o motor de jogo, onde a lógica de todo o simulador (controlo, comunicações, câmara) se encontra.

3.4.4 Programação

Através da exportação do modelo renderizado a partir do sistema Blender, inicia-se a fase de programação no motor de jogo MonoGame, dentro do ambiente integrado de desenvolvimento de programas Visual Studio. Foi idealizado que fosse implementado dentro do simulador algo que fosse simples de entender e utilizar, sendo que era interessante oferecer ao aluno um controlo manual para o robô, tudo no mesmo ecrã ao mesmo tempo, para além das funcionalidades pretendidas no simulador. Com este objectivo em mente, foram realizados vários testes de possíveis interfaces e formas de apresentar a informação ao utilizador de forma a não preencher muito o ecrã, mas também de maneira a que tudo o que seja importante esteja disponível ao utilizador.

Desta forma, esta idealização implicou que fosse desenvolvido algo que pudesse realizar uma integração de uma aplicação Windows dentro da aplicação construída a partir do motor de jogo MonoGame. Inicialmente o objectivo seria efectuar o inverso – uma integração do "mundo" gerado no MonoGame dentro de uma aplicação interface Windows. No entanto, chegou-se à conclusão de que, embora a integração fosse possível, o modelo do robô não iria ser renderizado na janela de visualização, isto porque o modelo não tinha sido criado dentro do MonoGame, pois originou de uma importação de outro sistema computacional (Blender); o motor de jogo não conseguia formular todas as definições base do modelo para um tipo de aplicação diferente, resultando numa impossível representação do mesmo. Assim sendo, esta aplicação Windows contém uma interface estilo WPF (*Windows Presentation Foundation*), programada pela linguagem XAML (*eXtensible Application Markup Language*), pois é uma linguagem utilizada para desenvolvimento de interfaces de utilizador no qual consegue efectuar as inicializações estruturais de vários valores e objectos, nomeadamente para as tais formas de interfaces.

Foi então realizada uma interface que contém o controlo manual dos vários eixos do robô através de botões, e caso alguma velocidade pretendida seja imposta, o utilizador apenas terá de selecionar a "*checkbox*" do eixo pretendido e escrever o valor de velocidade requisitada na caixa de texto. Além disso, é possível verificar nessa interface os vários limites de cada eixo do robô, e uns botões extra de controlo e informação que permitem ao utilizador realizar uma calibração do robô simulado para a sua posição de origem e também verificar os vários comandos já requisitados até ao momento, por exemplo. Para este último extra, foi introduzida em baixo do modelo simulado uma caixa de texto género consola de informações, onde tais comandos requisitados aparecem caso o utilizador pretenda, sendo que é onde

são também apresentados as várias mensagens que o robô possa enviar, tal como o robô real efectua através do seu protocolo de comunicação. Toda esta interface está representada na Figura 3.9, juntamente com o modelo simulado.

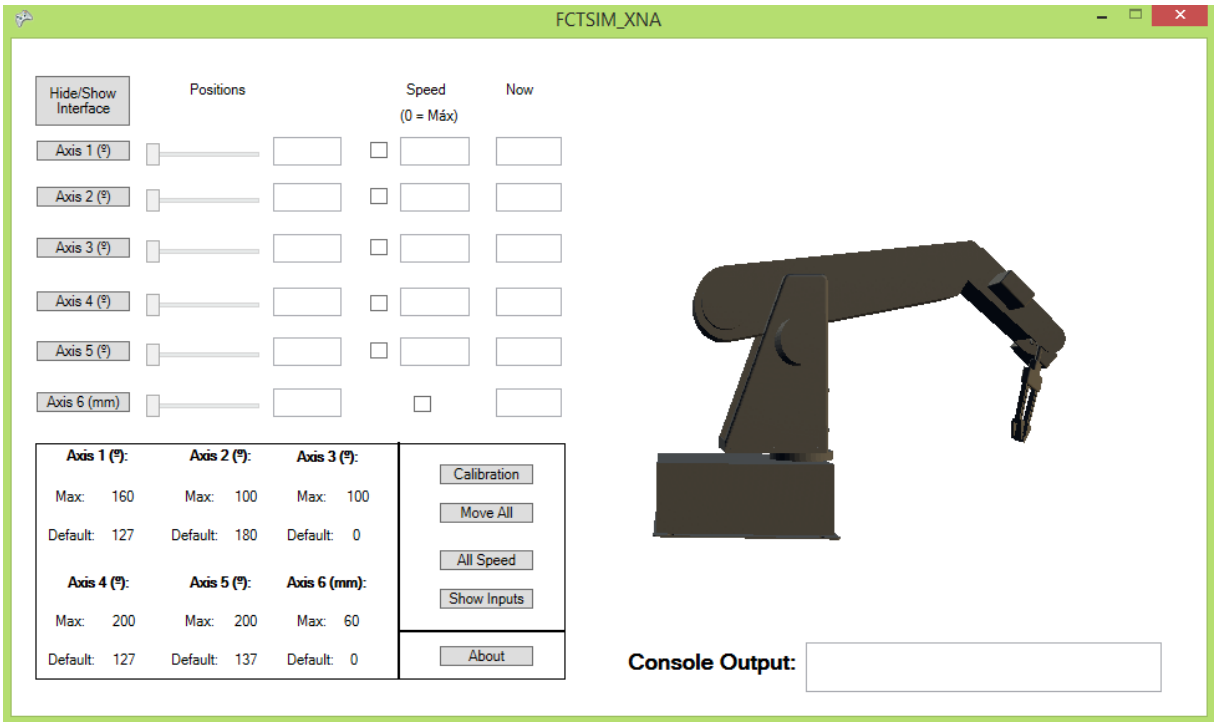

Figura 3.9: Interface do simulador implementado com controlo manual activo e informações do robô.

Caso o utilizador seja uma pessoa mais experiente quanto ao controlo do modelo do braço robótico, tem a opção de esconder esta interface implementada, deixando apenas a ser apresentado o modelo simulado, assim como a consola de informações, para uma melhor imersão de controlo e visualização do robô simulado. Para este estilo de interface mais "imersivo" está a Figura 3.10 representada para melhor ilustrar esta opção de trabalho.

Este simulador oferece ao utilizador a possibilidade de verificar melhor vários pontos de vista do modelo simulado através de um controlo da câmara manuseado pelo rato e teclado:

- **Roda do Rato:** efectua um *zoom in* ou *zoom out*, caso seja rodado para cima e para baixo, respectivamente;

- **Teclas Q e E:** enquanto premidas movimentam a câmara para a esquerda e direita, respectivamente;

- **Botão Direito do Rato:** ao ser premido retorna a câmara do simulador para a sua posição predefinida;

- **Botão Direito e Esquerdo do Rato:** pressionando ambos os botões do rato, serve de atalho para que a aplicação do simulador seja encerrada.

Figura 3.10: Interface do simulador sem controlo manual (oculto).

Finalmente, quanto aos tipos de comunicação entre o simulador e o utilizador, e consequentes animações do robô industrial, o simulador ao ser iniciado abre uma porta virtual para uma possível comunicação externa, intitulada de "*ComA*", assim como é efectuado um método de ligação por serviços REST. Desta maneira, após ter sido lido algum tipo de informação enviada por uma aplicação externa em qualquer uma das formas de comunicação, o simulador irá verificar o tipo de comando enviado, organizando e verificando as informações recebidas, de forma a poder realizar tarefas no robô, isto caso o comando enviado respeite o protocolo de comunicação do robô, não contendo qualquer lacuna na forma de requisitar o comando.

Caso os comandos enviados estejam a respeitar o protocolo de comunicação, ou algum comando seja requisitado pelo controlo manual disponível, passa-se à fase de movimentações do robô, no qual é indicado ao modelo qual o eixo a movimentar, de acordo com a posição e possível velocidade pretendidas pelo comando enviado.

4

Validação

Neste capítulo da dissertação, é onde serão mostrados os vários testes efectuados para avaliar a ferramenta de simulação, com o intuito de melhorar e perceber as potenciais características a desenvolver, para que o comportamento corresponda ao resultado esperado.

Assim sendo, os testes efectuados serão separados em duas frentes, em que uma é focada na movimentação e orientação que o modelo simulado reage de acordo com os comandos requisitados, e a outra consiste no meio de ligação entre cliente e servidor, no qual se verifica se as transmissões de dados estão a ser realizadas correctamente.

4.1 Validação da Arquitectura de Comunicação

Começando pela fase de validação da arquitectura de comunicação (visto que será um ponto de entrada entre a ferramenta de simulação e uma aplicação externa, caso não se pretenda o controlo manual anterior referido), vai ser testado uma verificação dos comandos enviados para o simulador, de forma a verificar se respeitam o protocolo de comunicação do robô e verificar como o simulador reage consoante os diferentes comandos que podem surgir.

Adicionalmente, será testado também as comunicações entre servidor e cliente, neste caso entre simulador e utilizador, respectivamente. Para esta parte, serão testados as duas vias de comunicação fornecidas, tanto com portas virtuais como com serviços REST, verificando como a ferramenta de simulação organiza os comandos recebidos para posterior realização dos mesmos no robô.

4.1.1 Verificação do Protocolo de Comunicação do Robô

Para a validação das movimentações do modelo do robô, é necessário primeiramente que os comandos recebidos pelo simulador estejam dentro dos parâmetros do protocolo de comunicação do robô real, de forma a existir o bom funcionamento do modelo (algo que foi referido no subcapítulo 3.3.1).

Desta forma, caso exista uma recepção de um comando que não respeite o protocolo de comunicação, o simulador deverá mostrar uma mensagem ao utilizador na sua consola de informações (pertencente à interface integrada para auxiliar o utilizador, referenciado anteriormente no subcapítulo 3.4.4) de que existe um erro, em que o comando pretendido contém os valores de entrada incorrectos. No caso de o comando enviado respeitar o protocolo de comunicação, mas conter valores de coordenadas fora dos limites dos eixos do robô, foi então idealizado que o simulador mostrasse uma nova mensagem na sua consola a informar o utilizador de que os valores inseridos estão fora dos limites possíveis do robô, fornecendo também ao utilizador a informação de quais os valores máximos e mínimos para o eixo que foi requisitado (isto acontece nos limites de posição como também nos de velocidade, sendo que os de posição encontram-se representados na própria interface do simulador). Um exemplo deste tipo de erro está ilustrado na Figura 4.1 a seguir representada, onde se introduziu uma velocidade acima do possível para um dos eixos, levando ao erro mencionado na consola de informações do simulador.

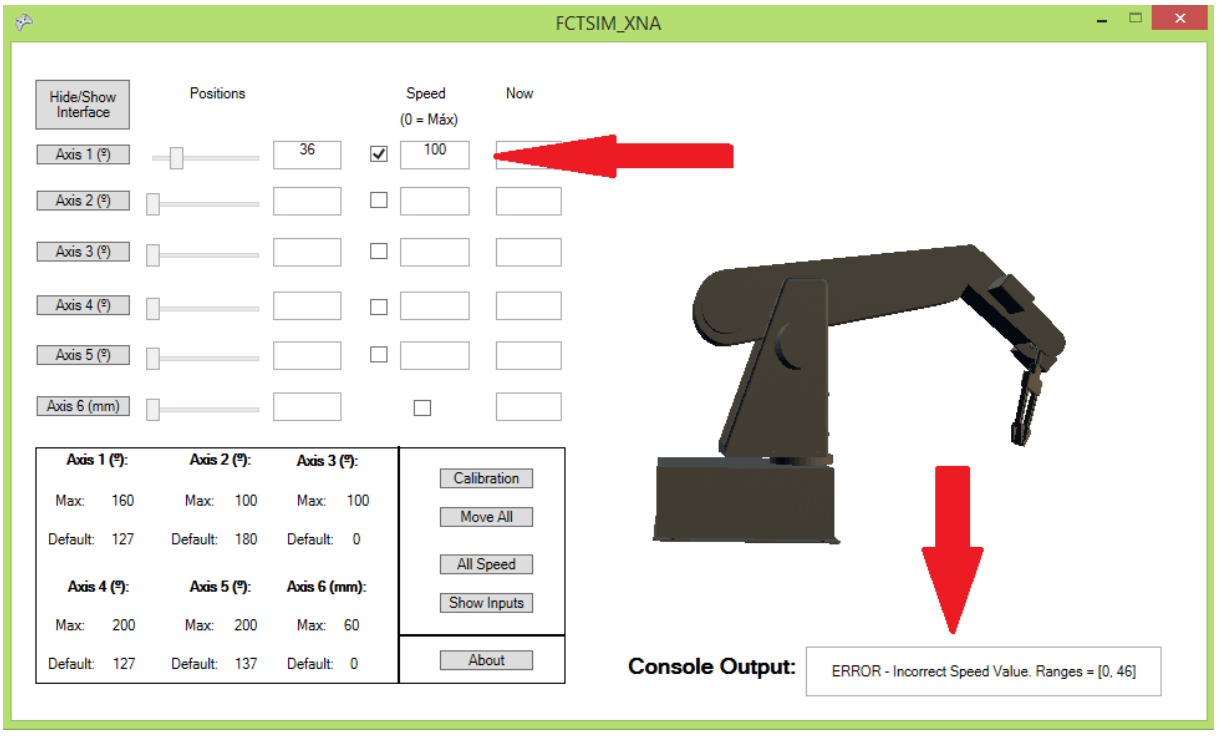

Figura 4.1: Exemplo de apresentação do erro dos limites de velocidade.

Para o caso do erro ocorrido ser subjacente ao facto do comando introduzido no simulador não respeitar o protocolo de comunicação do robô, aparece então o erro mencionado anteriormente, que está representado na Figura 4.2.

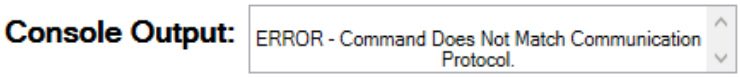

Figura 4.2: Representação de erro devido a protocolo de comunicação do robô.

Assim sendo, caso não haja quaisquer erros nos comandos enviados, se respeitarem o protocolo de comunicação do robô modelado, e contenha valores correctos dentro dos limites de posição ou velocidade, passa-se então à fase de animação e movimentação do robô simulado, testes esses referidos no subcapítulo 4.2.1.

4.1.2 Teste de Comunicação entre Cliente e Servidor

Embora exista um tipo de controlo manual fornecido ao utilizador dentro da interface do simulador, um dos objectivos principais desta ferramenta será a de possibilitar a comunicação de uma aplicação externa desenvolvida pelo próprio utilizador, no qual é capaz de receber comandos de dada aplicação e organizar as diversas informações que recebeu, de forma a realizar o pretendido.

Assim sendo, começando pelo estilo de comunicações entre portas virtuais, é possibilitado ao utilizador este tipo de comunicação pois, ao ser iniciado o simulador, é aberta uma porta virtual, "*ComA*", no qual ficará à espera de qualquer tipo de leitura feita por esse meio de comunicação. Este método de comunicação tem a desvantagem de obrigar o utilizador de abrir no seu programa uma segunda porta virtal, "*ComB"* por exemplo, de forma a poder existir um meio de envio e recepção de dados. Caso este meio de comunicação não consiga ser aplicado no início do funcionamento da ferramenta de simulação, esta deverá afirmar tal condição na sua consola de informações, tal como é ilustrado na Figura 4.3.

Figura 4.3: Representação da mensagem de impossível comunicação via portas virtuais.

No entanto, a porta virtual do simulador manter-se-á aberta, para o caso de surgir alguma outra aplicação com uma porta virtual aberta. Em caso de possível comunicação, os comandos enviados pelo utilizador serão tratados e organizados para futura implementação no robô modelado, algo que está mencionado no subcapítulo 4.2; se o simulador tiver de enviar algo de volta ao utilizador, efectuará um envio de dados por este meio de comunicação também.

Passando agora ao estilo de comunicação via serviços WEB REST, aqui este tipo de comunicação passa-se de uma forma mais simples para o utilizador, pois é sempre fornecido ao aluno pelo professor um ficheiro no qual já tem implementado a abertura de uma comunicação REST. Neste tipo de comunicação, o simulador que se representa como servidor ao ser iniciado, vai sempre abrir um "caminho" HTTP de forma a permitir ao utilizador, com o tal ficheiro fornecido, de estabelecer uma ligação para com o simulador. Desta forma, visto que estes tais "caminhos" em ambos os lados serem os mesmos, dependente da parte final do "caminho" HTTP, "*/SerialSend*" ou "*/SerialClose*" por exemplo, será realizada uma tarefa diferente, no qual o servidor responderá ao cliente com algo após cada comando requisitado, ou em caso de necessidade de resposta para certos comandos. Os testes feitos por serviços REST foram realizados pelo "*browser*" em modo "*offline*" de forma a representar um potencial cliente, resultando num modo de representação de mensagens de retorno da forma ilustrada nas Figuras 4.4 e 4.5.

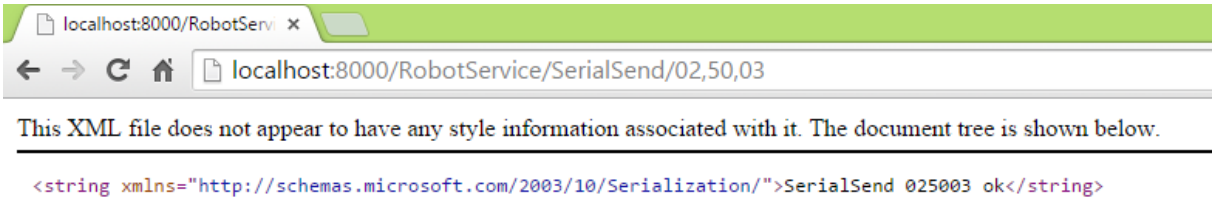

Figura 4.4: Representação da mensagem de retorno do servidor após comando de controlo do robô.

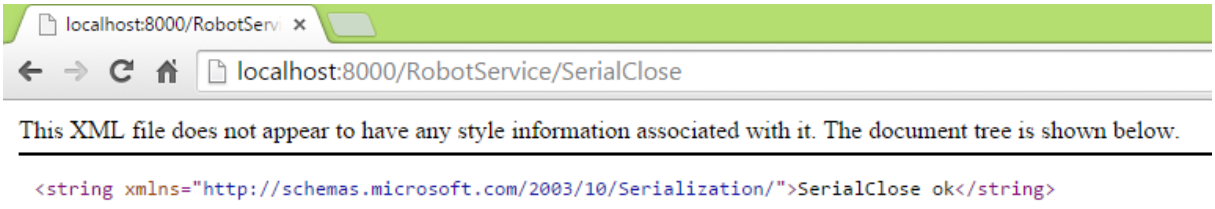

Figura 4.5: Representação da mensagem de retorno do servidor em caso de pedido de fecho de comunicação para com o servidor.

Caso seja efectuada uma tentativa de comunicação com uma aplicação externa que não tenha uma abertura de comunicação via serviços REST, a mesma coisa se passará como com as portas virtuais, em que a ferramenta de simulação manter-se-á sempre aberta como servidor, para o caso de uma aplicação com este estilo de comunicação surgir para enviar certos comandos ao robô simulado.

Finalmente, se uma possível e boa comunicação seja aberta entre o simulador e um cliente, o simulador ficará à espera de receber comandos do cliente, organizando e tratando dos dados recebidos para um posterior controlo do modelo, enviando depois para o cliente as diversas informações em como o controlo foi efectuado, dependente do comando recebido anteriormente.

4.2 Validação de Funcionalidades do Robô Simulado

Passando para uma fase de validação das funcionalidades do robô simulado, vai-se referir nesta secção nos testes de manipulação dos eixos, verificando se a orientação do robô simulado é a correcta para os diversos comandos requisitados.

4.2.1 Teste de Manipulação de Eixos

Entretanto, caso os comandos enviados para o simulador respeitem o protocolo de comunicação do robô, e contenham as coordenadas de posições e velocidades correctas, é realizada uma animação para o eixo pretendido. Para a questão das velocidades, considerando que cada eixo tem uma determinada velocidade normal, caso seja pretendida uma maior velocidade para um determinado eixo, é apenas realizada uma multiplicação do valor pretendido em relação ao valor normal de movimentação dos eixos do modelo no simulador, existindo então, uma movimentação mais rápida do eixo pretendido consoante o valor de velocidade implementado.

Para os diferentes eixos de rotação, novamente, em caso de um envio de comandos correcto de acordo com o protocolo de comunicação, em que as posições estão dentro dos limites esperados, é então realizada uma animação de rotação para o eixo desejado. Em caso de o comando enviado ao simulador indicar movimentação de todos os eixos, a animação é feita tal como o robô real, em que todos os eixos rodam ao mesmo tempo até às posições desejadas, resultando em algo semelhante ao representado na Figura 4.6.

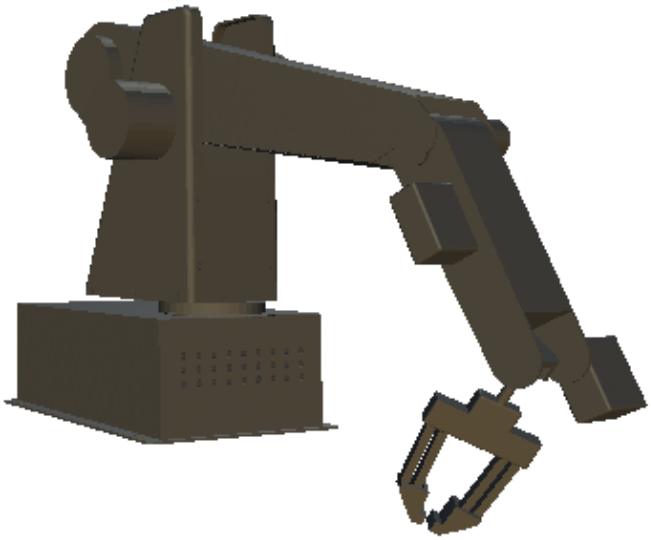

Figura 4.6: Representação do robô após comandos de movimentação de vários eixos.

Dependente do eixo, existe um tipo de movimentação diferente, por exemplo, a base do robô deverá rodar sobre si mesma, enquanto os outros eixos rodam consoante a origem dos seus pontos de rotação em questão.

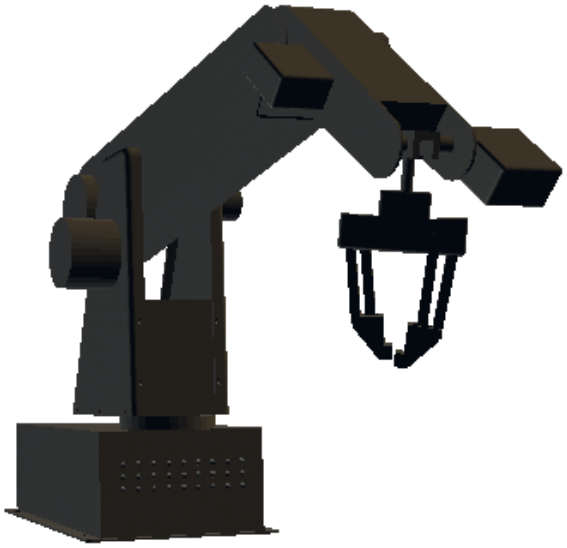

Figura 4.7: Exemplo de movimentação para fecho da garra.

A excepção vai para a garra do modelo, que apenas abre e fecha, contendo um estilo de movimentação também diferente, tal como a base. Ambas as animações da garra, de fecho e abertura, estão representadas nas Figuras 4.7 e 4.8, respectivamente.

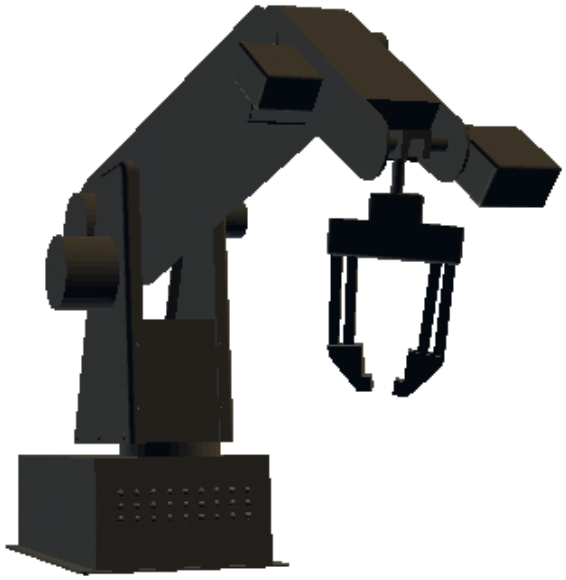

Figura 4.8: Exemplo de movimentação para abertura da garra.

4.3 Resultados da Validação

Após vários testes na ferramenta de simulação, chega-se à verificação de que a ferramenta de simulação desenvolvida vai resultar numa vantagem para os alunos quanto aos conhecimentos das funcionalidades do robô específico a ser simulado, assim como ajudar a conhecer quais os parâmetros a ter em consideração aquando do controlo do mesmo, para uma redução de acidentes durante a implementação no robô real. Em suma, a ferramenta presta-se como um bom auxiliar na aprendizagem de robótica.

Tendo sidos verificados e satisfeitos os vários requisitos estabelecidos como necessários num simulador de robô industrial, comprova-se que o simulador consegue substituir o robô real nas fases mais cruciais em que a segurança e integridade do equipamento estão mais em causa. Desta forma, conclui-se que a simulação é uma abordagem adequada para ajudar no ensino de robótica, pelo que a hipótese sugerida como resposta à questão de pesquisa é válida e satisfatória.

5

Conclusões

Neste último capítulo é realizado um sumário desta dissertação, no qual são referidos quais os aspectos mais importantes que ocorreram neste projecto durante a sua fase de investigação e implementação, seguido de pontos conclusivos quanto aos resultados obtidos, assim como uma referência quanto a um potencial futuro trabalho que poderá ser adicionado ao sistema construído.

Assim sendo, é realizada nesta secção final um resumo de todo o trabalho realizado no projecto, com o intuito de formular as possíveis conclusões a retirar para o projecto desta dissertação.

5.1 Síntese do Trabalho Efectuado

Conforme foi observado ao longo do projecto desta dissertação, a solução apresentada funciona com base no processo de criação de um simulador de um robô industrial, de forma a eliminar os obstáculos que costumam aparecer para um utilizador quanto ao conhecimento do funcionamento desse mesmo robô.

Assim sendo, a ferramenta de simulação proposta foi alcançada através de uma aplicação integrada sobre o motor de jogo MonoGame, fornecendo ao utilizador formas de controlar o robô simulado para facilitar os testes de implementação pretendidos no seu congênere real. Este simulador contém um modelo do robô industrial efectuado no sistema computacional SolidWorks, de forma a ser o mais semelhante possível do seu congênere real em termos de *design* e dimensões, uma interface simples e intuitiva, assim como várias formas de controlo do robô industrial, seja de forma manual ou através de serviços WEB. Em relação ao modelo do robô simulado, é importante referir que também passou pelo motor de renderização Blender de forma a serem efectuadas as diferentes animações e texturas para que fosse renderizado dentro do "mundo" do motor de jogo utilizado, tal como foi referido anteriormente neste documento.

Além disso, no contexto educacional em que o simulador foi idealizado, esta solução será uma grande vantagem para os alunos, pois permite-lhes uma forma mais acessível de perceber o funcionamento do robô, onde pode ser verificado os seus tipos de movimentação, limites, e comunicação através do próprio protocolo de comunicação da máquina. Desta maneira, é possível através deste simulador obter uma representação bastante similar do modelo em relação ao seu congênere real, de forma a obter os conhecimentos necessários aquando a implementação de testes no robô simulado, para uma possível implementação dos mesmos testes no robô real posteriormente.

A necessidade de um tipo de simulador como este é justificada pela quantidade de riscos que podem ser evitados aquando da implementação de testes sobre o robô real, o que vai proporcionar um melhor controlo sobre os equipamentos. No âmbito educacional em que o simulador é inserido, será uma mais-valia em relação aos poucos robôs funcionais existentes para testes de implementação para o elevado número de alunos existentes na disciplina específica. Desta forma, o objectivo de proporcionar uma ferramenta capaz de simular um robô industrial foi conseguida.

5.2 Trabalho Futuro

Vários pormenores podem ser adicionados a este projecto, dependente da ideia de funcionamento da ferramenta desejada. Desta maneira, uma possibilidade seria a de adicionar, utilizando a já implementada comunicação por serviços REST, um controlo do robô via "*browser*" em modo "*online*", de maneira a ser possível indicar ao utilizador com problemas ou dúvidas sobre qual o funcionamento correcto a tomar no robô de uma forma remota.

Entretanto, uma possível ferramenta que englobasse vários robôs industriais, em que pudesse ser feita uma escolha prévia de qual o sistema que se queria simular e testar, também seria interessante; no âmbito educacional no qual o projecto se encontra, uma possível junção de vários simuladores dos robôs industriais disponíveis numa só ferramenta.

Por último, seria também vantajoso oferecer ao utilizador uma abordagem em que se poderia escolher certas ligações de controlo entre vários robôs, por exemplo, em que um robô pegaria num cubo e oferecesse a um segundo robô, ou então o de delinear certos "caminhos" de movimentação seguidos para realizar, previamente definidos pelo utilizador, algo que não é possível efectuar dentro da mecânica de ferramenta de simulação idealizada e criada, podendo oferecer mais vantagens para o utilizador quanto a funcionalidades dos robôs simulados.

6

Bibliografia

ABB. (10 de Agosto de 2015). Obtido de "Robotics Produtcs' Overview": http://www.abb.com/product/pt/9AAC910011.aspx?country=PT

Anderson, E. F., Engel, S., Comninos, P., & McLoughlin, L. (2008). Em *The Case for Research in Game Engine Architecture.* Paper presented at the Proceeding of the 2008 Conference on Future Play: Research, Play, Share.

ARM. (10 de Agosto de 2015). Obtido de "Overview": http://thearmrobot.com/

Autodesk. (10 de Agosto de 2015). Obtido de "Products' Overview": http://www.autodesk.com/produtcs/

Barata, J. (2012). FCT-UNL. *Aula de Cinemática: Cadeira de Robótica*.

Barata, J. (2012). FCT-UNL. *Aula de Introdução: Cadeira de Robótica*.

Blender. (10 de Agosto de 2015). Obtido de "Features": http://www.blender.org/

BMA. (8 de Setembro de 2015). Obtido de "Process Automation": http://www.bma-automation.com/Prozessautomatisierung.2102.0.html?id=2102&L=1

Busby, J., Parrish, Z., & Wilson, J. (2009). "Volume I: Introduction to Level Design with Unreal Engine 3". Em *Mastering Unreal Technology (Vol. 1).* Pearson Education.

CodeProject. (22 de Julho de 2015). Obtido de "Testing Serial Application with Virtual Ports: Testing Serial Application": http://www.codeproject.com/Articles/27705/Testing-Serial-Application-with-Virtual-Ports

Creighton, R. H. (2010). "Example: A Seat-of-Your-Pant Manual for Building Fun, Groovy Little Games Quickly". Em *Unity 3D Game Development.* Packt Publishing Ltd.

C-STEM. (10 de Agosto de 2015). Obtido de "Program Support": http://c-stem.ucdavis.edu/teachers-administrators/

Diaz, R., & Behr, J. G. (2010). "Modeling and Simulation Fundamentals". Em *Discrete-Event Simulation* (p. 57).

Digital-Tutors. (25 de Janeiro de 2015). Obtido de 3ds Max vs. Maya: Is One Better than the Other?: http://blog.digitaltutors.com/3ds-max-vs-maya-is-one-better-than-the-other/

Electronic Industries Association. (1969). Em *Interface Between Data Terminal Equipment and data Communication Equipment Emplying Serial Binary Data Interchange.* Washinton: Electronic Industries Association, Engineering Dept. (OCLC).

Escola Superior Tecnologia: IPCB. (8 de Setembro de 2015). Obtido de "The Robots": http://www.est.ipcb.pt/laboratorios/robotica/The_Robots.html

Escola Superior Tecnologia: IPS. (8 de Setembro de 2015). Obtido de "Robótica": http://ltodi.est.ips.pt/manuell/robotica/robotica0405.html

Farin, G., Hoschek, J., & Kim, M. (2002). Em *Handbook of Computer Aided Geometric Design.* Elsevier.

Fielding, R. (2000). Em *Architectural Styles and the Design of Network-based Sofware Architectures: PhD Dissertation, Chapter 5.* University of California.

GmbH, P. P. (30 de Outubro de 2002). Em *TR5 Robot Manual: Technical Specifications - Version 2.4e.*

Gregory, J. (2009). Em *Game Engine Architecture.* CRC Press.

Kay, & Jennifer. (2005). Em *Introduction to Homogeneous Transformations & Robot Kinematics.* Rowan University Computer Science Department.

Klauser, S. (1990). *Patente Nº VS24KMOE.BES.*

Mack, P. (2009). Understading Simulation-Based Learning. *SGH-Life Support Training Centre*.

Marietta, J. B. (1999). Em *Discrete Event Simulation* (pp. 7-13). Initially published in the Proceedings of the 1999 Winter Simulation Conference (ed. Nembhard, D. T. Sturrock, G. W. Evans).

MonoGame. (10 de Agosto de 2015). Obtido de "About": http://www.monogame.net/.

MSDN. (13 de Setembro de 2015). Obtido de "A Guide to Designing and Building RESTful Web Services with WCF 3.5: REST Defined": https://msdn.microsoft.com/en-us/library/dd203052.aspx

Nilson, B., & Söderberg, M. (2007). Em *Game Engine Architecture.*

Parallemic. (10 de Agosto de 2015). Obtido de "RoKiSim 1.7: Robot Kinematics Simulator": http://www.parallemic.org/RoKiSim.html

PTC. (10 de Agosto de 2015). Obtido de "Produtcs": http://www.ptc.com/

RoboAnalyzer. (10 de Agosto de 2015). Obtido de "About": http://www.roboanalyzer.com/

Rosas, J. (2014). FCT-UNL. *Aula de Simulação em SDM: Cadeira de Sistema Distribuídos de Manufactura*.

Schafersman, S. (Janeiro, 1994). "Scientific Thinking and the Scientific Method". Em *An Introduction to Science.* Online Whitepaper.

Sherrod, A. (2006). Em *Ultimate 3D Game Engine Design & Architecture.* Charles River Media, Inc.

SolidWorks. (10 de Agosto de 2015). Obtido de "3D Design Produtcs' Capabilities": http://www.solidworks.com/sw/products/

Sommerville, I., Melnikoff, S. S., Arakaki, R., & Andrade Barbosa, E. (2003). Em *Engenharia de Software (Vol. 6).*

Wiki: Microsoft XNA. (10 de Agosto de 2015). Obtido de "Overview: XNA Game Studio 4.0": http://en.wikipedia.org/wiki/Microsoft_XNA

Wings3D. (10 de Agosto de 2015). Obtido de "Features": http://www.wings3d.com/