UNIVERSIDADE DA BEIRA INTERIOR
Engenharia

# O Ensino da Robótica
# com Recurso à Programação Visual
## RoboSapien, um caso de estudo

**Vítor Hugo Mendes Gaudêncio Araújo**

Dissertação para obtenção do Grau de Mestre em

**Curso Mestrado em Engenharia Electromecânica**

(2º ciclo de estudos)

Orientador: Prof. Doutor António Espírito Santo

**Covilhã, 24 de Junho de 2010**

# Dedicatória

Em memória do Engenheiro Humberto Santos.

A minha esposa, filha e a mãe.

# Agradecimentos

Gostaria de agradecer aos Professores Doutores António Espírito Santo e o Abílio Manuel Pereira Silva pela orientação e disponibilidade demonstrada.

# Resumo

O trabalho que a seguir se apresenta diz respeito à concepção e desenvolvimento de uma aplicação informática, de carácter didáctico, e que visa permitir a programação de um robot através de uma interface gráfica. O robot ao qual se destina a aplicação possui a designação de Robosapien. Apesar de ser um produto comercial foi já alvo de várias modificações. Este trabalho pretende ser mais uma contribuição, no sentido em que se ambiciona disponibilizar uma ferramenta de programação intuitiva, destinada a utilizadores sem conhecimentos avançados de programação. Uma possível utilização desta ferramenta será a sala de aula, nas disciplinas de introdução à programação.

# Palavras-chave

Robosapien, Robosapien em sala de aula, Software pedagógico, Método expositivo, Aprendizagem baseada em problemas e em projecto.

# Abstract

The work presented in this document concerns the design and development of a computer application, for educational purpose, and designed to allow software development through a GUI to control the Robosapien. Despite this robot being a commercial product, it has already been subject to several modifications. This work aims to be a contribution, in the sense that proposes an intuitive programming tool, designed for users without advanced knowledge of programming. A potential use of this tool will be the classroom, in introductory programming courses.

# Keywords

Robosapien, Robosapien in the Classroom, Educational Software, lecture method, Problem Based Learning and Project.

# Índice

Capítulo 1 - Robosapien e sua contribuição no ensino ..... 19
1.1 Introdução ..... 19
1.2 Robosapien ..... 19
1.3 Mark Tilden ..... 20
1.4 Contribuição no ensino ..... 21
Capítulo 2 - Análise das soluções existentes ..... 23
2.1 Programas de estudo ..... 23
2.1.1 *Scratch* ..... 24
2.1.3 *Lego Mindstorm* ..... 26
2.1.4 *Picaxe Programming Editor* ..... 27
Capítulo 3 - Desenvolvimento da aplicação ..... 28
3.1 Escolha da ferramenta de programação ..... 28
3.2 Ambiente de desenvolvimento *Delphi* ..... 28
3.3 Princípio de funcionamento do programa ..... 29
3.3.1 Pressupostos utilizados na concepção do programa ..... 29
3.3.2 Exemplos práticos ..... 30
3.3.3 Ligações entre instruções ..... 34
3.4 Modo de utilizador ..... 43
3.5 Modo de programação ..... 44
3.5.1 Inserir instruções ..... 44
3.5.2 Ordem das instruções do Robosapien ..... 46
3.5.3 Transferência das instruções ..... 48
3.6 Descrição dos vários menus ..... 49
Capítulo 4 - Robosapien na sala de aula ..... 54
4.1 Fundamentação teórica ..... 54
4.1.1 Método expositivo ..... 54
4.1.2 Aprendizagem baseada em problemas ..... 55
4.1.3 Aprendizagem baseada em projecto ..... 55
4.2 Organização de uma aula modelo ..... 56
4.2.1 Conteúdos/Aprendizagens a adquirir pelos alunos: ..... 56
4.2.2 Estratégias/Actividades a desenvolver ..... 56
4.2.3 Diferenciação pedagógica ..... 56
4.2.4 Recursos a utilizar ..... 57
4.2.5 Avaliação ..... 57
4.2.6 Sumário da aula ..... 57
Capítulo 5 - Síntese ..... 58
Bibliografia ..... 59
Lista de Instruções ..... 62

# Lista de Figuras

Figura 1 - Ambiente gráfico do *Scratch* ..... 25
Figura 2 - Ambiente gráfico do *Lego Mindstorm* NXT ..... 26
Figura 3 - Ambiente gráfico do *Picaxe Programming Editor* ..... 27
Figura 4 - Representação das várias posições das instruções ..... 34
Figura 5- Caso 1 R1XX < R2X e RYY1 > RY2 ..... 35
Figura 6- Caso 2 R1XX < R2X e RY1 < RYY2 ..... 35
Figura 7- Caso 3 R1XX < R2X e RY1 > RYY2 ..... 36
Figura 8- Caso 4 R1X < R2X e R1Y > R2YY ..... 37
Figura 9- Caso 5 R1XX>R2X e RY1>RYY2 ..... 37
Figura 10- Caso 6 R1X > R2XX e RYY1 > RYY2 ..... 38
Figura 11 - Caso 7 R1XX < R2XX e RY1 > RYY2 ..... 39
Figura 12- Caso 8 R1X > R2XX e RYY1 > RY2 ..... 39
Figura 13- Caso 9 R1X > R2XX e RYY1 < RY2 ..... 40
Figura 14- Caso 10 R1X<R2X e RYY1<RY2 ..... 41
Figura 15- Caso 11 R1X < R2X e RYY1 < RY2 ..... 41
Figura 16 - Caso 12 R1XX < R2X e RYY1 < RY2 ..... 42
Figura 17- Estrutura principal do programa ..... 43
Figura 18- Caixa de diálogo a solicitar o tempo de duração da instrução ..... 44
Figura 19- Visualização dos comandos em texto ..... 44
Figura 20- Apresentação das instruções ..... 45
Figura 21- Menu que permite editar a instrução: cortar, copiar, apagar, colar e propriedades ..... 45
Figura 22- Menu Propriedades ..... 46
Figura 23- Função que permite interligar as instruções ..... 46
Figura 24- O programa solicita a 1ª Instrução Primeira instrução ..... 47
Figura 25- O programa solicita a 2ª instrução. ..... 47
Figura 26 - Menu de interligação de instruções ..... 48
Figura 27- Menu ficheiro Novo ..... 49
Figura 28- Menu guardar ficheiro ..... 50
Figura 29- Menu permite copiar, colar e apagar instrução. ..... 51
Figura 30 - Menu de Simulação do programa para o Robot. ..... 52
Figura 31- Menu Ajuda ..... 53

# Lista de Tabelas

Tabela 1- Estrutura da informação enviada para o Robosapien ..... 29
Tabela 2 - Exemplo de lista de instruções do Robosapien..... 30
Tabela 3 - Lista de instruções para o robot com o tempo parametrizados..... 31
Tabela 4 - Apresentação da lista de código que será enviado para o robot..... 32

# Lista de Acrónimos

| | |
|---|---|
| UBI | Universidade da Beira Interior |
| PDA | *Personal digital assistant* |
| MIT | *Massachusetts Institute of Technology* |
| WEB | *World Wide Web* |
| BASIC | *Beginner's All-purpose Symbolic Instruction Code* |

# Capítulo 1 - Robosapien e sua contribuição no ensino

## 1.1 Introdução

A existência de ferramentas pedagógicas, directamente relacionadas com o domínio científico das tecnologias, ajuda a promover o sucesso escolar. Em simultâneo, estas ferramentas auxiliam o docente na tarefa de motivar o aluno. Um dos domínios de aplicação passível de ser identificado é o da robótica. As matérias a dominar por parte do aluno para que possam desenvolver actividades no domínio da robótica são várias, das quais podemos salientar as seguintes: electrónica, programação de algoritmos, mecânica, etc. Constata-se também que a obtenção de resultados condiciona fortemente a motivação do aluno. Assim, todas as ferramentas que possam auxiliar o aluno no desenvolvimento das suas actividades contribuem para o seu sucesso escolar.

Ao longo deste documento é realizada a descrição de uma ferramenta informática, desenvolvida com o intuito de auxiliar os alunos na programação de robots. A aplicação permite ao aluno interagir com um robot comercial, designado por Robosapien, e que foi já alvo de várias intervenções com o propósito de o adaptar a vários tipos de acções pedagógicas.

O documento está organizado como a seguir se descreve:

No Capítulo 1 é feita uma breve descrição do Robosapien e de como este dispositivo pode ser integrado nas actividades de ensino-aprendizagem.

As soluções existentes no domínio de aplicação são analisadas no Capítulo 2, nomeadamente, os programas que serviram de ponto de partida para o desenvolvimento da ferramenta aqui proposta.

O desenvolvimento da aplicação é descrito no Capítulo 3. É dada especial atenção aos pressupostos tidos na concepção do programa, sendo apresentados exemplos práticos ilustrativos. O capítulo é ainda dedicado ao funcionamento do programa e à descrição das suas funcionalidades.

A utilização em sala de aula da ferramenta desenvolvida e respectivo suporte teórico dos métodos de ensino-aprendizagem são descritos no Capítulo 4.

A finalizar o documento é apresentada no Capítulo 5 uma síntese do trabalho desenvolvido identificando os pontos fortes e os futuros desenvolvimento.

## 1.2 Robosapien

O Robosapien é um robot desenhado por Mark Tilden que pode ser controlado remotamente através de uma ligação de infravermelhos. Os comandos podem ser enviados pelo telecomando que o acompanha ou por um computador ou PDA com suporte das funcionalidades necessárias ao envio de comandos pela porta de infravermelhos. Actualmente, entusiastas da electrónica desenvolvem e acrescentam novas funcionalidades ao Robosapien.

A sua tecnologia baseia-se nos princípios de coordenação motora do corpo humano (tecnologia biomórfica), que lhe confere movimentos mais próximos da realidade humana. Trata-se do primeiro robot comercial de baixo custo com a capacidade de simular o comportamento humano.

O *RoboSapien* é sensível ao toque e ao som. Integra o sistema *Interactive Reflex* que consiste num conjunto de sensores distribuídos pelo corpo. Estas capacidades sensoriais permitem-lhe detectar obstáculos (JC, 2004).

Actualmente, o *Robosapien* revela ser uma ferramenta poderosa na educação, podendo ser utilizado para divulgar conteúdos, ser um assistente do docente ou um mero apresentador.

Um exemplo de aplicabilidade é apresentado por You, Shen, Liu, & Chen (2006). Neste trabalho, o Robosapien tem um papel importante na actividade de ensino-/aprendizagem, já que interage com os alunos em inglês. Nas acções desenvolvidas, o *Robosapien* começapor contar uma história em Inglês. Em seguida, o *Robosapien* selecciona um aluno a quem são colocadas questões relacionadas com a história. O robot é comandado pelos alunos, por intermédio de comandos de voz em Inglês. Na realidade, no decorrer do processo de aprendizagem, um indivíduo controla o robot através do computador, sem que os alunos tenham consciência da sua presença. A comunicação do robot com os alunos é realizada com recurso ao programa da Microsoft, *Speech - SDK*, "*Text to Speech*". Segundo o estudo realizado por You, Shen, Liu e Chen (2006), o docente deverá conhecer as potencialidades e as limitações do Robot para poder tomar o maior partido desta ferramenta de aprendizagem. Simultaneamente, deve ter em atenção quais as tarefas propostas para as quais os alunos demonstrem interesse e motivação. Contudo, o método torna-se ineficaz se a turma tiver mais de 30 alunos e a sala for superior a 100 metros quadrados.

O estudo descrito anteriormente ilustra os benefícios para o sucesso na educação que a interacção dos alunos com um robot pode potenciar.

### 1.3 Mark Tilden

O investigador Mark W. Tilden é conhecido pelo trabalho desenvolvido no domínio da Robótica *Beam*. A designação deriva da Biologia, Electrónica, Estética e da Mecânica (Marsh, 2010). Do seu trabalho resultaram robots que executam movimentos complexos, sendo controlados por circuitos analógicos simples. Geralmente recorrem a circuitos integrados de lógica discreta, dispensando o uso de microprocessadores.

Nascido no Reino Unido, iniciou a sua actividade académica na Universidade de Waterloo. Posteriormente, trabalhou no *Laboratório Nacional de Los Alamos*, onde desenvolveu robôs para fins militares. Colaborou também com o programa *Nasa Space*. *O Sat Bot é* uma das suas criações mais conhecidas, consiste num satélite com a capacidade de se alinhar com o campo magnético da Terra. Actualmente, Mark Tilden coopera com a *WowWee Toys* na concepção de robots biomórficos como o B.I.O. Bug e o RoboSapien.

## 1.4 Contribuição no ensino

O conceito da Robótica didáctica provém do ensino-/aprendizagem com o recurso a robots. Estes dispositivos são constituídos por partes mecânicas e eléctricas. O controlo do seu funcionamento pode estar a cargo de circuitos analógicos ou circuitos digitais como microcontroladores ou microprocessadores.

O robot passa a ser um objecto que suporta o desenvolvimento cognitivo do aluno, ou seja, segundo Seymour Papert (1986) "Um objecto para pensar".

Actualmente, os discentes possuem algumas dificuldades no que diz respeito à compreensão e ao desenvolvimento de algoritmos de programação. Com o auxílio da robótica, essas dificuldades podem ser minimizadas, aproximando os conceitos teóricos da sua utilização prática.

Em 1976, Seymour Papert desenvolveu a linguagem de programação *Logo*, tendo por base a teoria desenvolvida por Piaget, que é vocacionada para a educação. Esta linguagem de programação procura promover o desenvolvimento cognitivo do aluno, à medida que os discentes vão sendo confrontados com novas experiências.

Muitos centros de investigação, nomeadamente o MIT, estão a desenvolver equipamentos e aplicações com o intuito de serem utilizados em actividades de ensino-aprendizagem, nomeadamente, o *Scratch*, *Squeak*, *Lego Mindstorm* e *Logo*. Este tipo de aprendizagem possibilita ao discente desenvolver a sua capacidade cognitiva com o auxílio do computador.

A aplicação da robótica proporciona aos discentes a aprendizagem de conceitos e princípios. Estes objectivos são alcançados não só por intermédio da montagem e desmontagem de peças mecânicas (polias, correias, engrenagens, etc.) e eléctricas (sensores, motores e lâmpadas), mas também através da concepção de algoritmos de controlo dos movimentos do Robot. Para além de permitir desenvolver conceitos na área da Matemática, da Física e da Informática, possui ainda a particularidade de promover a interdisciplinaridade ente diferentes domínios do saber.

A programação possibilita ao robot mover-se, através da coordenação de vários movimentos, e interagir com o meio através de sensores e de motores.

Segundo esta perspectiva, a robótica didáctica proporciona ao aluno a oportunidade de melhorar o seu raciocínio lógico, num objecto que realiza as funções para o qual foi programado. Face a esta situação, obrigará o discente a reflectir e a procurar soluções que lhe permitam controlar o Robot, promovendo a aprendizagem através do erro. Perante este tipo de atitude, o aluno desenvolve competências de organização de conhecimento, de reflexão e de pesquisa.

A robótica promove o desenvolvimento de dispositivos robóticos, que poderão suportar outros domínios científicos, nomeadamente, a Física e a Matemática, promovendo o trabalho autónomo do discente.

Programar um robot consiste em interligar as partes mecânicas e eléctricas do sistema, com o objectivo de coordenar movimentos, de forma a promover a realização do processo ou da tarefa. Para isso é necessário que o algoritmo, através de comandos, accione os elementos mecânicos do sistema, tendo em atenção o estado dos sensores.

Estão disponíveis comercialmente vários de *kits* de robótica, oferecendo uma vasta variedade de componentes tais como motores, sensores, rodas e cremalheiras. Regra geral, possuem uma aplicação

informática de interface que permite ao utilizador programar o comportamento do robot. Os principais fornecedores desses *kits* de robótica são: *Symphony*, *Lego*, *Lynxmotion*, *Fishertechik*, entre outros.

Verifica-se que a robótica é uma ferramenta útil tanto para os docentes como para os alunos, pois possibilita a demonstração de conceitos teóricos, através da sua aplicação prática, facilitando a sua compreensão. Nesta perspectiva, o conhecimento é construído pelo aluno, ao invés de ser ensinado pelo professor. Neste caso, o aluno é o protagonista na sua aprendizagem.

# Capítulo 2 - Análise das soluções existentes

## 2.1 Programas de estudo

Ao iniciar o desenvolvimento deste trabalho, foi realizada uma análise das diversas aplicações informáticas existentes no mercado. Esses programas são:

- *Scratch* do *MIT;*
- *Lego Mindstorm*;
- *Picaxe Programming Editor*.

### 2.1.1 *Scratch*

Recentemente, uma série de relatórios dos Estados Unidos direccionam a sua atenção para a proliferação das novas tecnologias, concluindo que a maioria das pessoas não está devidamente preparada. Estes relatórios referem que é necessário tomar iniciativas a fim de ajudar os indivíduos económica e socialmente mais desfavorecidos. Em paralelo, reconhecem a importância que as tecnologias podem ter na reformulação do conhecimento e da criatividade.

Na última década, os Estados Unidos têm aberto muitos centros tecnológicos com o objectivo de incentivar o uso das novas tecnologias. A maior parte desses centros tecnológicos são dedicados a actividades tão simples como o processamento de texto, o envio de correio electrónico, a navegação na *internet* ou a realização de jogos educacionais.

No passado, este tipo de iniciativas estavam direccionadas para as escolas. Com o surgimento dos centros tecnológicos pretende-se proporcionar conhecimentos básicos de informática. Assim, é possível ajudar os alunos a projectar, a construir e a inventar novas tecnologias. Anteriormente, este tipo de iniciativa, ou seja, a introdução de programação nos centros tecnológicos, falhava devido às dificuldades intrínsecas, nomeadamente, à falta de pessoal especializado e à execução de actividades fora do contexto, havendo ainda o facto de a programação ser uma actividade exigente.

O *Scratch* é uma aplicação informática concebida para utilizadores com idades compreendidas entre os 10 e os 18 anos, tendo sido testado num clube criado pelo *Media MIT*. Na figura 1 é possível observar a sua interface gráfica. O desenvolvimento e a concepção do *Scratch* foram orientados de acordo com as necessidades e as limitações dos centros tecnológicos.

Nos clubes de informática os jovens visualizam os trabalhos dos outros, partilham ideias e reformulam os seus conhecimentos. O *Scratch* foi desenvolvido segundo características, que outros não possuíam e que, por sua vez, falhavam sempre que se tentava a sua introdução junto da comunidade mais jovem.

O *Scratch* é uma ferramenta acessível quando comparada com outras do mesmo género. Esta característica deve-se ao facto de integrar uma interface gráfica que permite que os programas sejam construídos como blocos de encaixe, como se fosse a montagem de um brinquedo tipo Lego.

Os comandos e os tipos de dados são representados por diferentes blocos de formas diversas, onde as peças se encaixam sempre que, sintaticamente, estejam correctas. Este tipo de abordagem anula os erros de sintaxe de uma forma fácil e intuitiva.

No *Scratch*, o programador arrasta os blocos (*drag and drop*) provenientes de uma palete, com o objectivo de criar procedimentos que regulam os comportamentos dos objectos. O multiprocessamento também pode ser utilizado no *Scratch*, onde vários blocos podem ser executados paralelamente. O *Scratch* pode ser instalado em portáteis, telemóveis, dispositivos móveis, entre outros equipamentos (Kereki, 2008). Tendo sido programado em *Squeak*, o seu código fonte é de livre acesso.

A programação é uma actividade intelectual complexa. Tem-se observado, ao longo dos anos, que os alunos têm muitas dificuldades na compreensão da lógica associada à actividade de programação. Para os alunos é difícil lidar com a lógica da programação e familiarizarem-se com os seus algoritmos. O *Scratch* pretende introduzir os alunos na programação, além dos motivar e melhorar as suas experiências.

O ensino da programação implica que os alunos adquiram novos conhecimentos e que se adoptem estratégias de índole prática.

As principais dificuldades dos discentes estão relacionadas com as actividades de resolução de problemas. Para isso, devem propor soluções e exprimi-las através de algoritmos de programação. Existem diferentes tipos de abordagens. Umas utilizam ferramentas de simulação, outras utilizam jogos, programação em pares, outras fichas, mas ainda não existe nenhum resultado que confirme a técnica mais indicada na aprendizagem da programação. O *Scratch* é uma linguagem de programação que pode servir inicialmente para proporcionar experiências, além de permitir um elo de ligação entre outras linguagens.

Histórias interactivas podem ser facilmente criadas através da programação utilizando o *Scratch*. O utilizador pode incluir animações, jogos e música. À medida que as crianças desenvolvem os programas, é possível observar o desenvolvimento das suas capacidades relacionadas com a área da Matemática e da Informática, além de permitir aprofundar os conhecimentos sobre o processo de concepção. Alguns dos algoritmos utilizados podem servir de comparação para outras linguagens. Com a utilização do *Scratch* é possível aos docentes identificar os alunos com mais dificuldades de programação e intervir mais rapidamente, sugerindo aos alunos a resolução de exercícios de crescente nível de dificuldade para, assim, colmatarem as suas dificuldades. Segundo Kereki (2008), não existe melhoria na aplicação do *Scratch* como linguagem de programação base, por isso, deve-se avaliar cuidadosamente a sua aplicação, devido à ausência de benefícios tangíveis (Maloney, Burd, & Kafai, 2004).

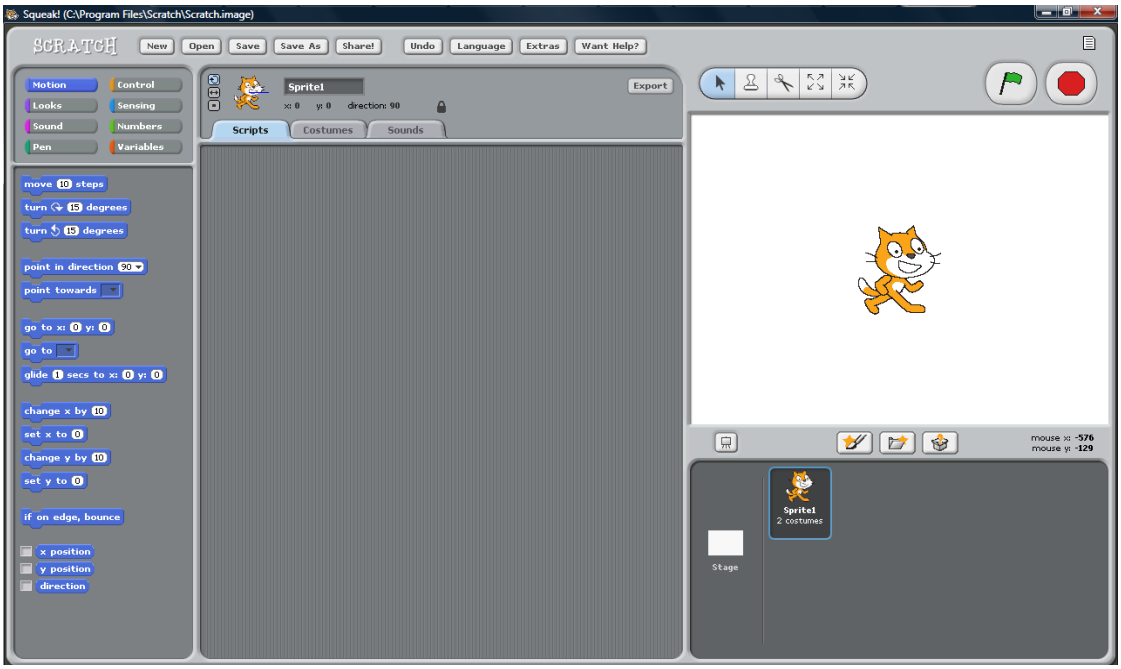


**Figura 1 - Ambiente gráfico do *Scratch***

### 2.1.3 *Lego Mindstorm*

O *Lego Mindstorms* é uma linha do brinquedo da Lego, vocacionada para a educação tecnológica, resultado de uma parceria entre o *Media Lab* do *Massachusetts Institute of Technology* (MIT) e o *Lego Group*. É constituído por um conjunto de peças mecânicas (rodas, correias, polias, engrenagens), acrescido de peças eléctricas como sensores de toque, de intensidade luminosa e de temperatura, que são controlados por um processador programável: o módulo RCX (*Robotic Command Explorer*).

O conjunto permite criar robots simples, passíveis de executar funções básicas pré-programadas.

O ambiente de programação pode ser observado na figura 2. Nele é possível desenvolver uma aplicação informática sem que seja necessário escrever código. O *Robolab* baseia-se na instrumentação e aquisição de dados. A aplicação informática permite estimular as capacidades cognitivas e criatividade do utilizador, além de, simultaneamente, despertar a sua curiosidade. O *Kit Mindstrom* pode ser aplicado a uma variedade de situações, no decorrer das quais é permitido ao utilizador introduzir conhecimentos das diversas áreas, nomeadamente, da Matemática, da Física, ou da Engenharia de Instrumentação e aquisição de dados.

Segundo os investigadores, não existem resultados suficientes para quantificar a aplicação desta plataforma no ensino-aprendizagem (Nagchaudhuri, Singh, Kaur, & George, 2002). Esta plataforma foi utilizada no estudo "*Children Cognitive Abilities in Construction and Programing Robots*". Ao longo deste estudo, pretendeu-se avaliar as capacidades de raciocínio, construção visual e lógica de crianças na construção e programação de robots, tendo-se concluído que estas crianças têm melhores aptidões para a construção e programação do que as restantes (Caci & D' Amico, 2002).

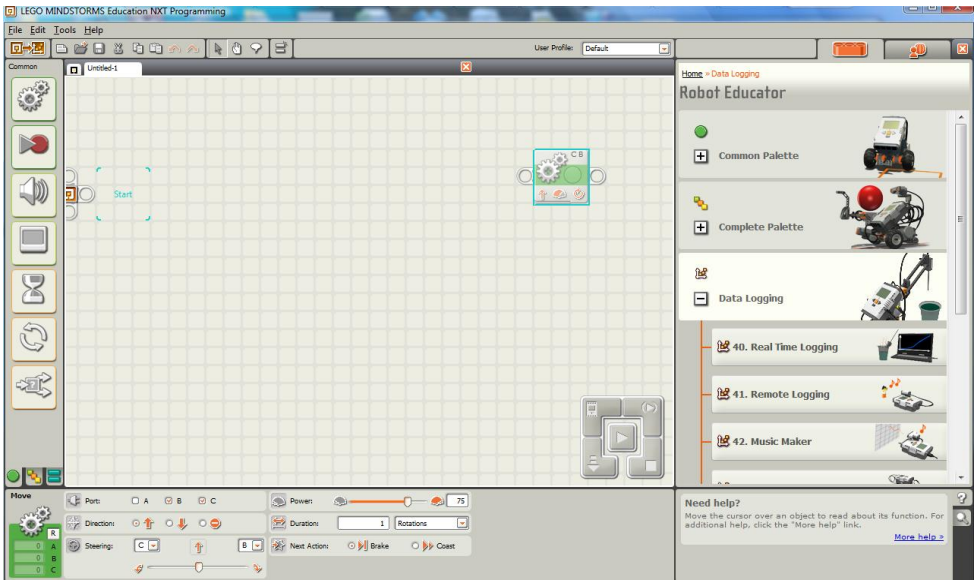


**Figura 2 - Ambiente gráfico do *Lego Mindstorm* NXT**

### 2.1.4 *Picaxe Programming Editor*

A *Picaxe* é uma marca de microcontroladores, com 11 dispositivos diferentes, de 8 a 40 pinos. A sua aplicação envolve as áreas da educação, dos *hobbies* da electrónica, das técnicas comerciais, incluindo a prototipagem rápida e o desenvolvimento. O editor também inclui uma série de assistentes, que ajudam na programação, e desenvolvimento do projecto.

A programação inclui um editor linha-a-linha na tela de simulação do programa BASIC (ver figura 3). Este modo de execução permite aos utilizadores observar o programa em funcionamento, bem como ajuda a identificar eventuais erros de programação.

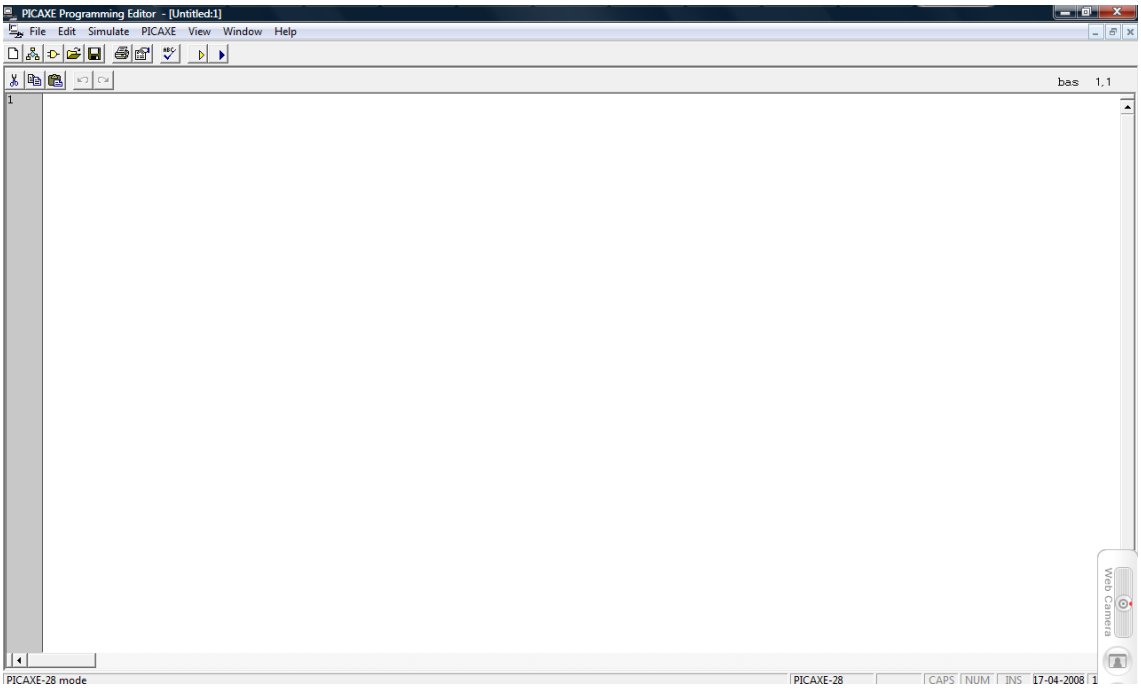


**Figura 3 - Ambiente gráfico do *Picaxe Programming Editor***

# Capítulo 3 - Desenvolvimento da aplicação

## 3.1 Escolha da ferramenta de programação

A escolha do *Delphi* como ferramenta de programação deve-se ao facto desta ser uma ferramenta de desenvolvimento da última geração do *Object Pascal*.

O *Delphi* engloba praticamente todos os aspectos do desenvolvimento, como aplicações em tempo real, sistemas de base de dados, cliente servidor, *ActiveX*. Além disso, proporciona uma grande facilidade de programação, sendo simples e rápida. Em simultâneo, suporta a programação orientada a objectos.

Os seus principais concorrentes são *Visual Basic*, *Visual Fox Pro*, *Java* e *Power Builder*.

## 3.2 Ambiente de desenvolvimento *Delphi*

O Delphi é um compilador projecto pela *Borland Software Corporation*, que se destina ao desenvolvimento de softwares, nomeadamente Base Dados, aplicações orientadas a objectos para o ambiente de trabalho, aplicações multi-camada cliente/servidor e aplicações para *internet*. O *Delphi* inicialmente foi concebido para funcionar na plataforma do sistema operativo *Windows*. Além disso, este compilador foi utilizado no desenvolvimento de aplicações para os sistemas operativos *Linux* e *Mac Os*. Existe um compilador similar ao *Delphi*, o *Lazarus* que funciona tanto com o sistema operativo *Windows* como nos sistemas operativos *Linux*, tendo a vantagem de ser um software livre.

A linguagem de programação foi utilizada no desenvolvimento de um número alargado de aplicações das quais se destacam: sistemas de simulação de controlo de aviões e helicópteros; sistemas de gestão de combustíveis; *Skype*; e base de dados.

Actualmente o Delphi produz aplicações de 32bits.

## 3.3 Princípio de funcionamento do programa

### 3.3.1 Pressupostos utilizados na concepção do programa

No programa foi necessário estipular o número de instruções máximo de 10000 instruções, devido à utilização das variáveis do tipo *ficha*.

Na concepção do programa considerou-se que a informação a enviar para o robot era a seguinte: instrução, duração da instrução; a próxima posição da instrução, dependendo do estado do sensor. Esta informação está reunida na tabela 1.

Tabela 1- Estrutura da informação enviada para o Robosapien

| Instrução | Duração da Instrução | Posição da Próxima Instrução Sensor *OFF* | Posição da Próxima Instrução Sensor *ON* |
|---|---|---|---|

Caso a instrução detecte que o sensor está *OFF*, o programa considera que próxima instrução é na posição "Próxima Instrução – Sensor *OFF"*. Se a instrução detectar que o sensor está *ON*, o programa considera que próxima instrução é na posição "Próxima Instrução – Sensor *ON"*.

Todas as instruções do robot são numeradas (ver anexos – tabela 5 ) e associadas a ficheiros, na qual o programa converte os números das instruções para binário. A instrução do robot e a sua duração temporal ocupa um espaço de 6-bits. A próxima instrução ("Sensor *ON"* e "*OFF"*) ocupa 14-bits cada uma, significando que cada instrução necessita de 40-bits para ser codificada.

O tempo máximo para cada instrução é de 60 segundos. Sempre que a duração da instrução ultrapassar os 60 segundos, o programa divide a duração da instrução por múltiplos de 60 segundos. A instrução é repetida o número de vezes necessário até que seja concluída.

No arranque do programa, houve a necessidade de criar um registo fictício para que a primeira instrução do programa do robot fosse comparada relativamente à sua posição. Caso o conceito fosse implementado, o programa, ao inserir a primeira instrução, iria verificar que ainda não existiam elementos registados. O *Delphi* devolveria o erro "*List Index out of bounds*", ou seja, nenhum registo teria sido criado.

A situação anterior era observada com a função que verificava se existia alguma instrução na posição da primeira instrução.

3.3.2 Exemplos práticos

*Exemplos 1*

Imaginemos que temos uma instrução A com uma duração de 150 segundos. Então, o programa envia a instrução para o robot da seguinte forma:

Instrução A, 60 segundos
Instrução A, 60 segundos
Instrução A, 30 segundos

*Exemplo 2:*

Supondo que pretendemos que o Robosapien execute as tarefas conforme a tabela 2.

Tabela 2 - Exemplo de lista de instruções do Robosapien

| N.º da posição | Instrução | Duração Instrução (s) | Posição da próxima Instrução Sensor *OFF* | Posição da próxima Instrução Sensor *ON* |
|---|---|---|---|---|
| 1 | 16 – *Start* | 0 | 3 | 0 |
| 2 | 13 - Virar para direita | 70 | 4 | 0 |
| 3 | 11 – Andar para a frente | 45 | 2 | 0 |
| 4 | 18 - Pancada com mão direita | 67 | 5 | 0 |
| 5 | 32 - Sensor pé direito frente | 0 | 3 | 6 |
| 6 | 15 – *Stop* | 0 | 0 | 0 |

A tabela 3 mostra a lista de instruções com os tempos parametrizados.

Tabela 3 - Lista de instruções para o robot com o tempo parametrizados

| Pos. Actual | Instrução | Duração Instrução (s) | Próxima Instrução Sensor *OFF* | Próxima Instrução Sensor *ON* |
|---|---|---|---|---|
| 1 | 16 – *Start* | 0 | 4 | 0 |
| 2 | 13 - Virar para direita | 60 | 3 | 0 |
| 3 | 13 - Virar para direita | 10 | 4 | 0 |
| 4 | 11 – Andar para a frente | 45 | 2 | 0 |
| 5 | 18 - Pancada com mão direita | 60 | 6 | 0 |
| 6 | 18 - Pancada com mão direita | 7 | 7 | 0 |
| 7 | 32 - Sensor pé direito frente | 0 | 4 | 6 |
| 8 | 15 – *Stop* | 0 | 0 | 0 |

O código binário a ser transferido para o robot é listado na tabela 4.

Tabela 4 - Apresentação da lista de código que será enviado para o robot

| Pos. Actual | Instrução | Duração Instrução (s) | Posição da Próxima Instrução Sensor *OFF* | Posição da Próxima Instrução Sensor *ON* |
|---|---|---|---|---|
| 1 | 16 – *Start* | 0 | 4 | 0 |
| | **Código em binário** | | | |
| | 010000 000000 00000000000100 00000000000000 | | | |
| 2 | 13 - Virar para direita | 60 | 3 | 0 |
| | **Código em binário** | | | |
| | 001101 111100 00000000000011 00000000000000 | | | |
| 3 | 13 - Virar para direita | 10 | 4 | 0 |
| | **Código em binário** | | | |
| | 001101 001010 00000000000100 00000000000000 | | | |
| 4 | 11 – Andar para a frente | 45 | 2 | 0 |
| | **Código em binário** | | | |
| | 001011 101101 00000000000010 00000000000000 | | | |
| 5 | 18 - Pancada com mão direita | 60 | 6 | 0 |
| | Código em binário | | | |
| | 010010 111100 00000000000110 00000000000000 | | | |
| 6 | 18 - Pancada com mão direita | 7 | 7 | 0 |
| | **Código em binário** | | | |
| | 010010 000111 0000000000111 000000000000000 | | | |
| 7 | 32 - Sensor pé direito frente | 0 | 4 | 6 |
| | **Código em binário** | | | |
| | 100000 000000 0000000000100 000000000000110 | | | |
| 8 | 15 – *Stop* | 0 | 0 | 0 |
| | **Código em binário** | | | |
| | 001111 000000 00000000000000 00000000000000 | | | |

O código binário que codifica as instruções a serem transferidas para o robot é listado a seguir:

```
010000 000000 00000000000100 00000000000000
001101 111100 00000000000011 00000000000000
001101 001010 00000000000100 00000000000000
001011 101101 00000000000010 00000000000000
010010 111100 00000000000110 00000000000000
010010 000111 00000000001110 00000000000000
100000 000000 00000000001000 00000000000110
001111 000000 00000000000000 00000000000000
```

O ficheiro a enviar para o Robosapien terá um tamanho de 40 bytes.

### 3.3.3 Ligações entre instruções

Para interligar os blocos de instruções entre eles, houve a necessidade de criar um algoritmo que abrangesse todos os casos possíveis. Suponhamos que os blocos de instruções a interligar são os rectângulos, na qual o caso 0 é a primeira instrução e os restantes casos poderão ser a posição da instrução dois. A instrução 1 tem as fronteiras no eixo dos xx, nos pontos R1X e R1XX e no eixo dos yy no ponto (R1Y a R1YY) enquanto a instrução dois, tem as fronteiras no eixo dos xx, nos pontos R2X e R2XX, e no eixo dos yy, tem as fronteiras nos pontos R2Y e R2YY. Na figura 4 são representadas as diversas posições da instrução 1 e 2.

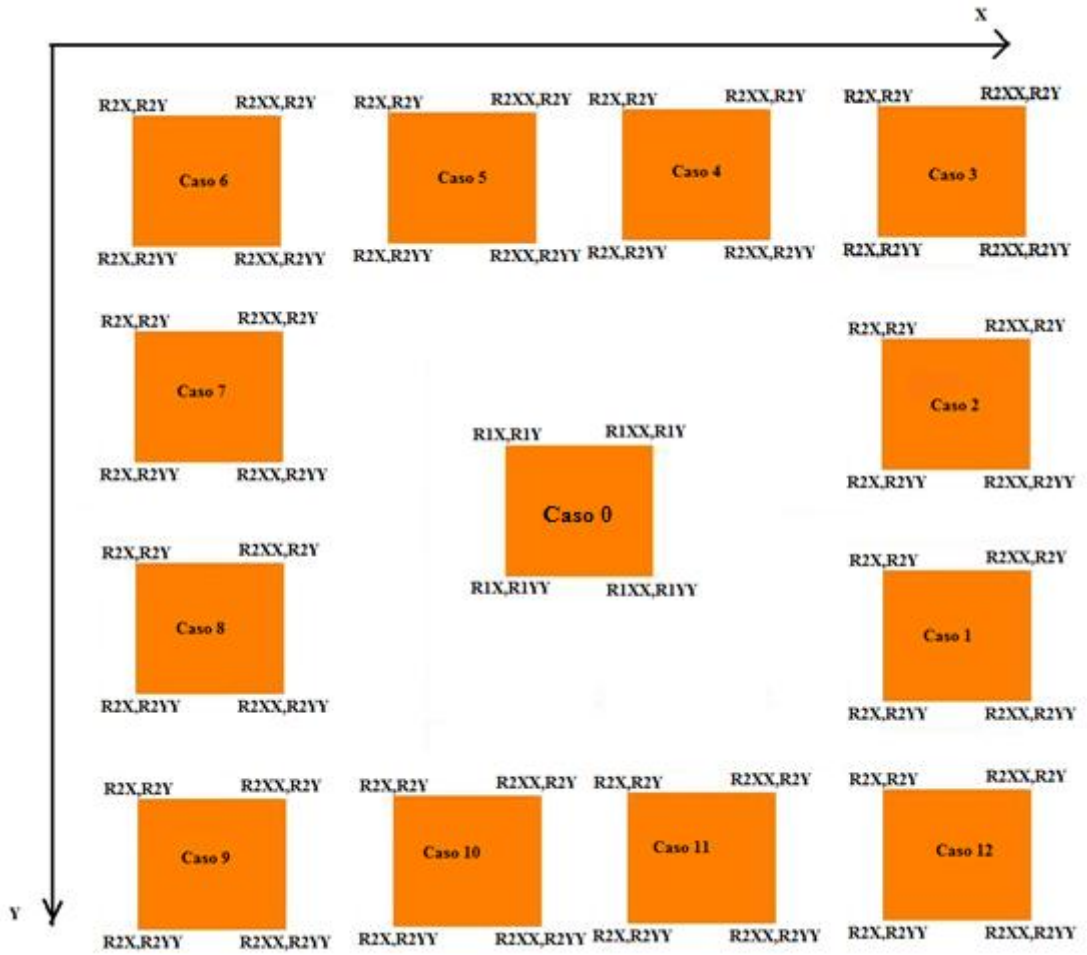


Figura 4 - Representação das várias posições das instruções

Caso 1 – R1XX < R2X e RYY1 > RY2

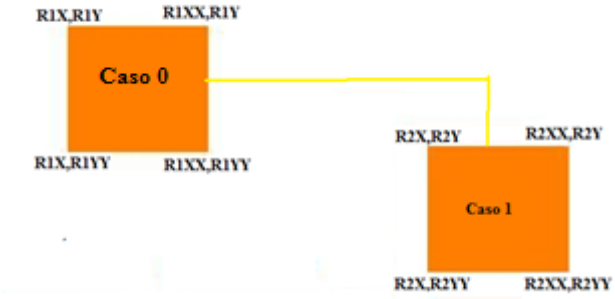


Figura 5- Caso 1 R1XX < R2X e RYY1 > RY2

O programa começa a desenhar a linha no ponto de origem $(R1XX, \frac{R1Y+R1YY}{2})$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R1Y+R1YY}{2})$, dessa posição desenha outra linha até $(\frac{R2X+R2XX}{2}, R2Y)$.

Caso 2 - R1XX < R2X e RY1 < RYY2

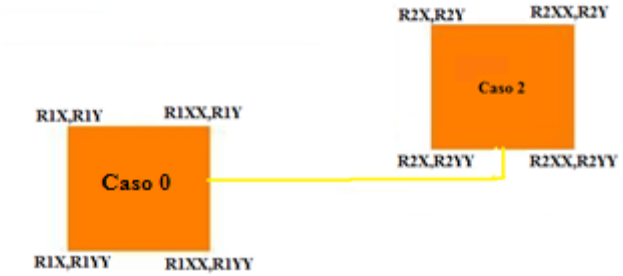


Figura 6- Caso 2 R1XX < R2X e RY1 < RYY2

O programa começa a desenhar a linha no ponto de origem $(R1XX, \frac{R1Y+R1YY}{2})$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R1Y+R1YY}{2})$, dessa posição desenha outra linha até $(\frac{R2X+R2XX}{2}, R2Y)$.

Caso 3 - R1XX < R2X e RY1 > RYY2

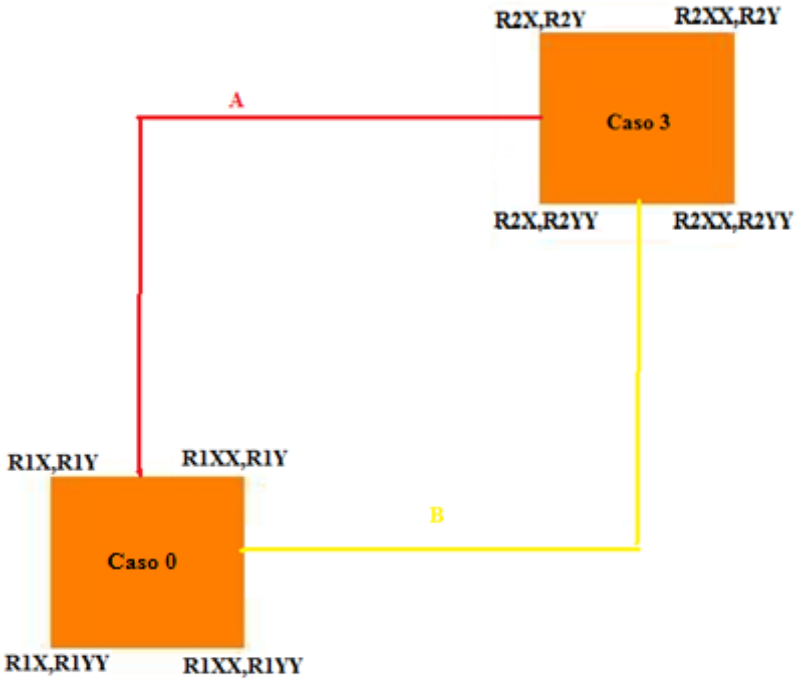


Figura 7- Caso 3 R1XX < R2X e RY1 > RYY2

Neste caso existem dois caminhos possíveis, o programa calcula a distância entre:

A: O programa começa a desenhar a linha no ponto de origem $(\frac{R1X+R1XX}{2}, R1Y)$ até à posição $(\frac{R1X+R2XX}{2}, \frac{R2Y+R2YY}{2})$, dessa posição desenha outra linha até $(R2X, \frac{R2Y+R2YY}{2})$.

B: O programa começa a desenhar a linha no ponto de origem $(R1XX, \frac{R1Y+R1YY}{2})$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R1Y+R1YY}{2})$, dessa posição desenha outra linha até $(\frac{R2X+R2XX}{2}, R2YY)$.

A alínea seleccionada é aquela em que a distância é a mais pequena.

Caso 4 - R1X < R2X e R1Y > R2YY

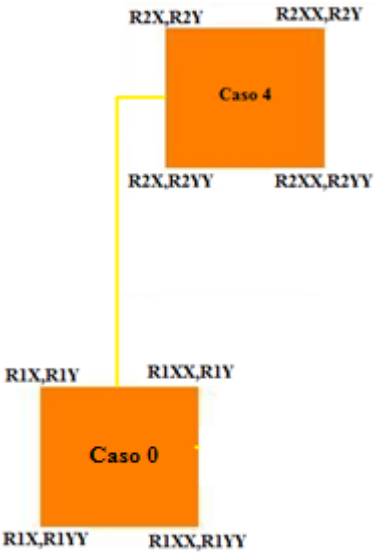


Figura 8- Caso 4 R1X < R2X e R1Y > R2YY

O programa começa a desenhar a linha no ponto de origem $(\frac{R1X+R1XX}{2}, R1Y)$ até à posição $(\frac{R1X+R2XX}{2}, \frac{R2Y+R2YY}{2})$,dessa posição desenha outra linha até (R2X,$\frac{R2Y+R2YY}{2}$).

Caso 5- R1XX>R2X e RY1>RYY2

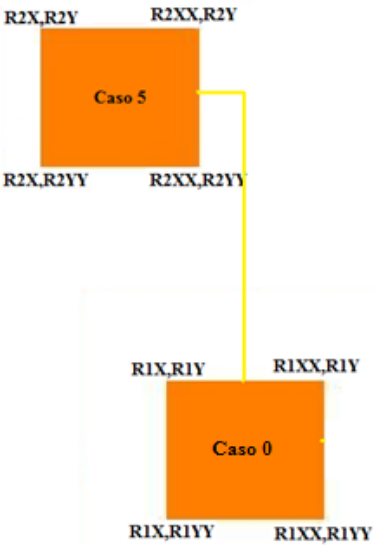


Figura 9- Caso 5 R1XX>R2X e RY1>RYY2

O programa começa a desenhar a linha no ponto de origem $(\frac{R1X+R1XX}{2}, R1Y)$ até à posição $(\frac{R1X+R2XX}{2}, \frac{R2Y+R2YY}{2})$, dessa posição desenha outra linha até (R2XX,$\frac{R2Y+R2YY}{2}$).

Caso 6- R1X>R2XX e RYY1>RYY2

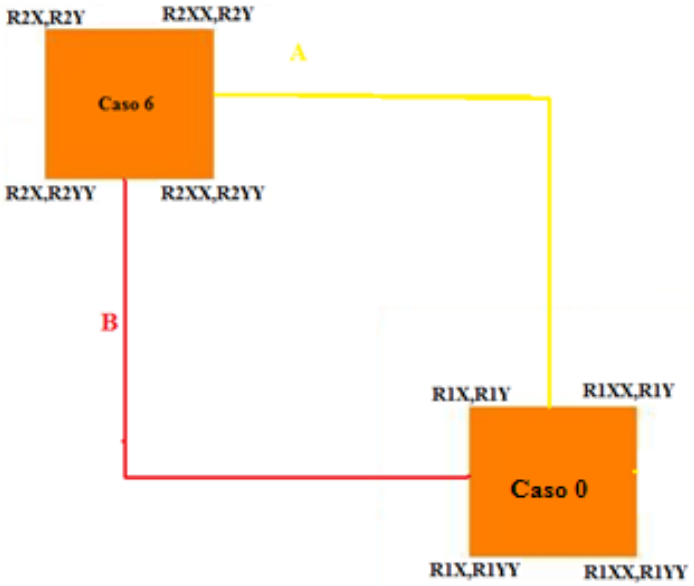


**Figura 10- Caso 6 R1X > R2XX e RYY1 > RYY2**

A: O programa começa a desenhar a linha no ponto de origem $(\frac{R1X+R1XX}{2}, R1Y)$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R2Y+R2YY}{2})$,dessa posição desenha outra linha até (R2XX, $\frac{R2Y+R2YY}{2}$).

B: O programa começa a desenhar a linha no ponto de origem $(R1X, \frac{R1Y+R1YY}{2})$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R1Y+R1YY}{2})$, dessa posição desenha outra linha até $(\frac{R2X+R2XX}{2}, R2YY)$.

A alínea seleccionada é aquela em que a distância é a mais pequena.

Caso 7 - R1XX < R2XX e RY1 > RYY2


Figura 11 - Caso 7 R1XX < R2XX e RY1 > RYY2

O programa começa a desenhar a linha no ponto de origem $(R1X, \frac{R1Y+R1YY}{2})$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R1Y+R1YY}{2})$, dessa posição desenha outra linha até $(\frac{R2X+R2XX}{2}, R2YY)$.

Caso 8 - R1X > R2XX e RYY1 > RY2

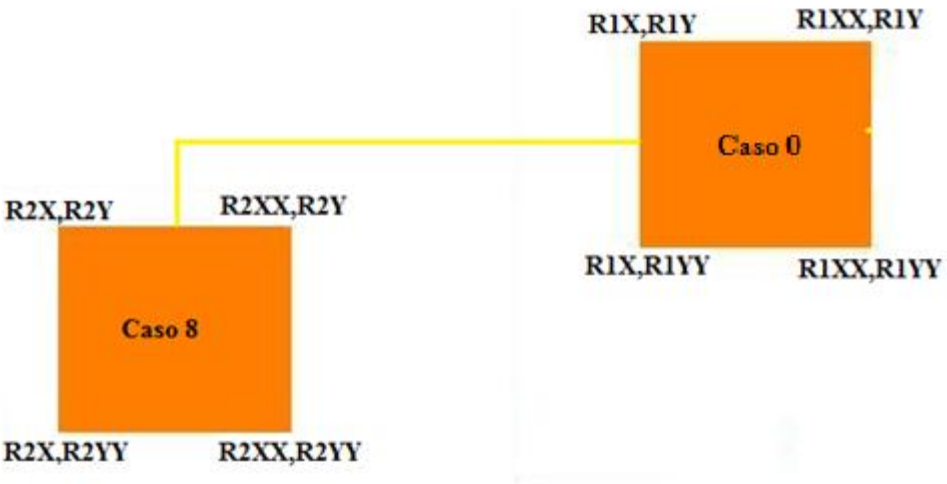


Figura 12- Caso 8 R1X > R2XX e RYY1 > RY2

O programa começa a desenhar a linha no ponto de origem $(R1X, \frac{R1Y+R1YY}{2})$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R1Y+R1YY}{2})$, dessa posição desenha outra linha até $(\frac{R2X+R2XX}{2}, R2YY)$.

Caso 9 - R1X > R2XX e RYY1 < RY2

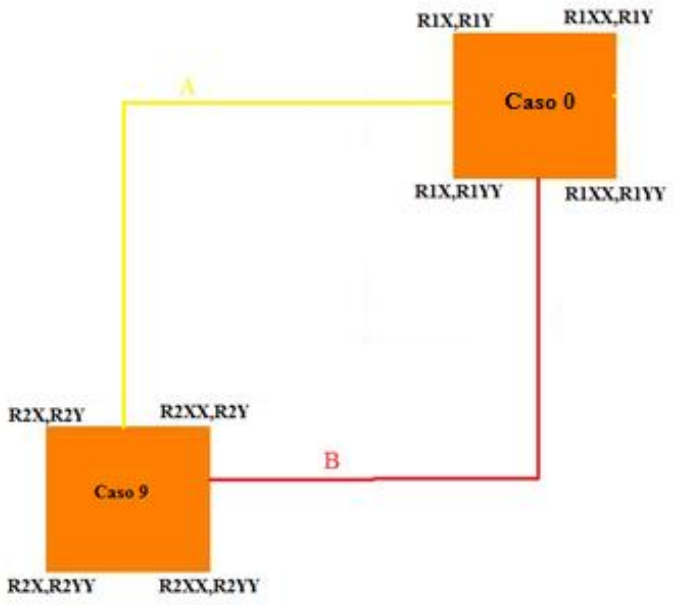


**Figura 13- Caso 9 R1X > R2XX e RYY1 < RY2**

A: O programa começa a desenhar a linha no ponto de origem $(R1X, \frac{R1Y+R1YY}{2})$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R1Y+R1YY}{2})$, dessa posição desenha outra linha até $(\frac{R2X+R2XX}{2}, R2Y)$.

B: O programa começa a desenhar a linha no ponto de origem $(\frac{R1X+R1XX}{2}, R1YY)$ até à posição $(\frac{R1X+R1XX}{2}, \frac{R2Y+R2YY}{2})$, dessa posição desenha outra linha até $(R2XX, \frac{R2Y+R2YY}{2})$.

A alínea seleccionada é aquela em que a distância é a mais pequena.

Caso 10 - R1X < R2X e RYY1 < RY2

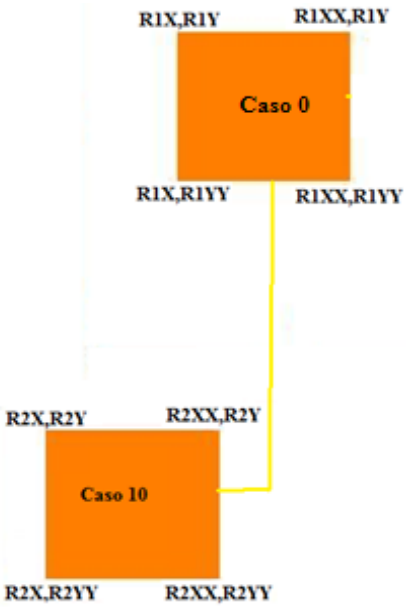


Figura 14- Caso 10 R1X<R2X e RYY1<RY2

O programa começa a desenhar a linha no ponto de origem $(\frac{R1X+R1XX}{2}, R1YY)$ até à posição $(\frac{R1X+R1XX}{2}, \frac{R2Y+R2YY}{2})$, dessa posição desenha outra linha até (R2XX, $\frac{R2Y+R2YY}{2}$).

Caso 11- R1X<R2X e RYY1<RY2

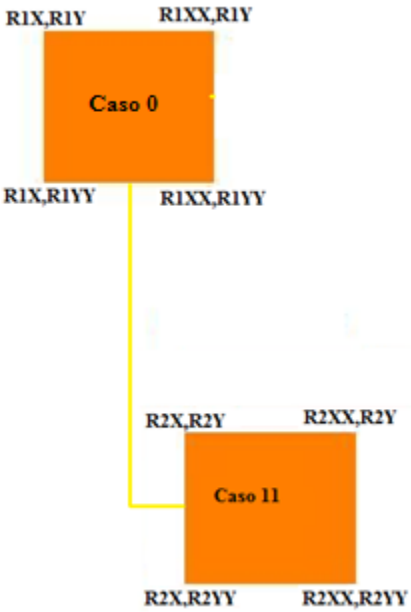


Figura 15- Caso 11 R1X < R2X e RYY1 < RY2

O programa começa a desenhar a linha no ponto de origem $(\frac{R1X+R1XX}{2}, R1YY)$ até à posição $(\frac{R1X+R1XX}{2}, \frac{R2Y+R2YY}{2})$,dessa posição desenha outra linha até (R2X, $\frac{R2Y+R2YY}{2}$).

Caso 12- R1XX < R2X e RYY1 < RY2

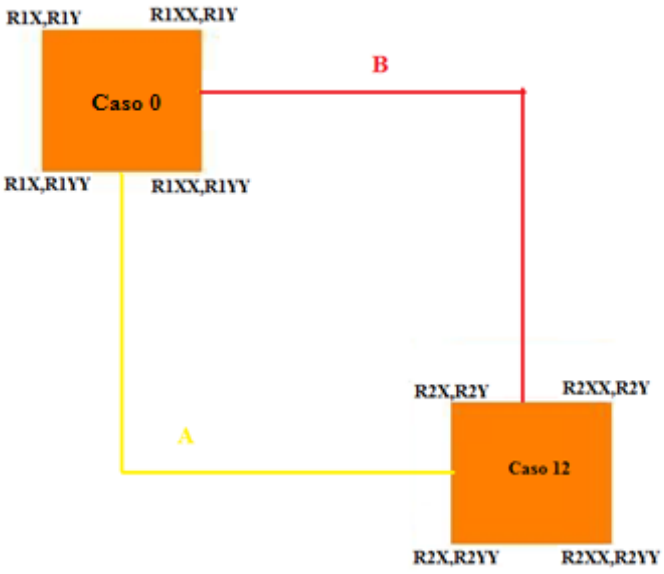


Figura 16 - Caso 12 R1XX < R2X e RYY1 < RY2

A: O programa começa a desenhar a linha no ponto de origem $(\frac{R1X+R1XX}{2}, R1YY)$ até à posição $(\frac{R1X+R1XX}{2}, \frac{R2Y+R2YY}{2})$,dessa posição desenha outra linha até (R2X,$\frac{R2Y+R2YY}{2}$).

B: O programa começa a desenhar a linha no ponto de origem $(R1XX, \frac{R1Y+R1YY}{2})$ até à posição $(\frac{R2X+R2XX}{2}, \frac{R1Y+R1YY}{2})$, dessa posição desenha outra linha até $(\frac{R2X+R2XX}{2}, R2Y)$.

A alínea seleccionada é aquela em que a distância é a mais pequena.

## 3.4 Modo de utilizador

A aplicação desenvolvida destina-se a controlar o Robosapien através do computador. Do lado esquerdo do interface, o utilizador tem acesso às instruções disponíveis. Para as utilizar no programa que está a desenvolver, o utilizador deve inserir na zona de programação, situada do lado direito do interface, as instruções necessárias.

O interface gráfico de programação é ilustrado na figura 17. Quando um dos botões é pressionado, logo após o clique no painel de programação do robot para seleccionar o bloco de instrução, o mesmo é inserido na zona de programação. Nesse momento, surge uma caixa de diálogo, a solicitar o tempo de duração da instrução. A instrução é guardada automaticamente na memória do programa, utilizando um formato que suporta a seguinte informação:

- número da instrução;
- posição do bloco de instrução no painel de programação;
- posição da instrução para o robot;
- ordem da instrução.

Todos os blocos de instruções são ordenados pela ordem de chegada. Durante a programação das instruções do robot, é possível executar tarefas de edição das instruções: modificar a posição na zona de programação e a ordem na lista de instruções, copiar, substituir e apagar as respectivas propriedades da instrução. Após a selecção das instruções, é necessário definir a ordem das instruções para o robot. Para isso, é necessário seleccionar ou pressionar o comando para interligar os blocos das várias instruções. Após a programação da aplicação, é possível visualizar as instruções em código binário. Para isso, basta fazer um clique no botão *código*. A transferência das instruções do robot é feita através do botão PC-Robosapien.

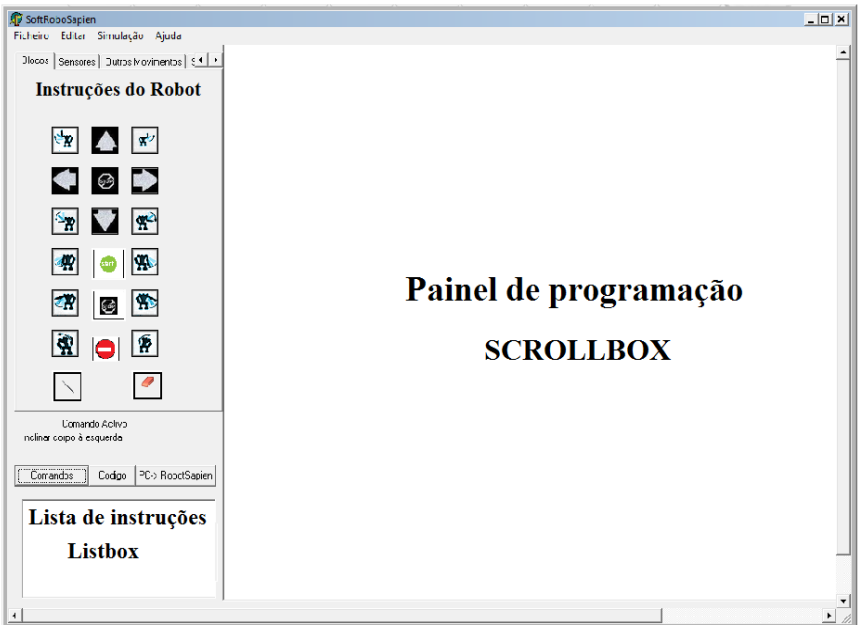


Figura 17- Estrutura principal do programa

## 3.5 Modo de programação

### 3.5.1 Inserir instruções

No arranque do programa do Robosapien a variável *reiniciar* possui com o valor lógico falso. Quando um dos comandos do robot é seleccionado, um dos procedimentos é verificar o estado da variável reiniciar. Caso esta esteja no estado lógico falso, o programa reinicializa todas as variáveis, esvazia a lista de instruções e modifica o estado da variável reiniciar para verdadeiro.

Posteriormente, ao clicar com a tecla do lado esquerdo do rato no painel de programação, o programa vai verificar se existe naquela posição alguma instrução. Caso não exista, surge uma caixa de texto, a solicitar a duração da instrução (ver figura 18).

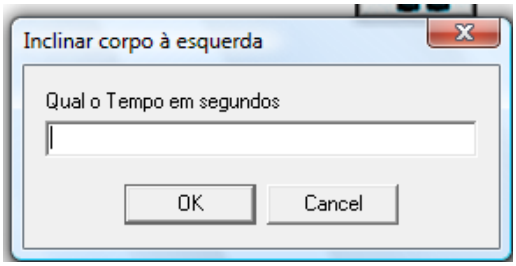


**Figura 18- Caixa de diálogo a solicitar o tempo de duração da instrução**

Apenas são aceites valores de tempo múltiplos de 1 segundo. Esta caixa de diálogo não aparece caso a instrução seja uma condição, como por exemplo, as instruções relativas ao estado dos sensores do Robosapien. Posteriormente, o programa guarda o número da instrução, a sua duração e a sua posição. Além disso, o bloco de comando recebe os atributos de movimento (*Drag and Drop*).

Se pretendermos ver as instruções actualizadas deve-se pressionar do lado esquerdo do comando (ver figura 19 e figura 20 na zona limitada a vermelho).


**Figura 19- Visualização dos comandos em texto**

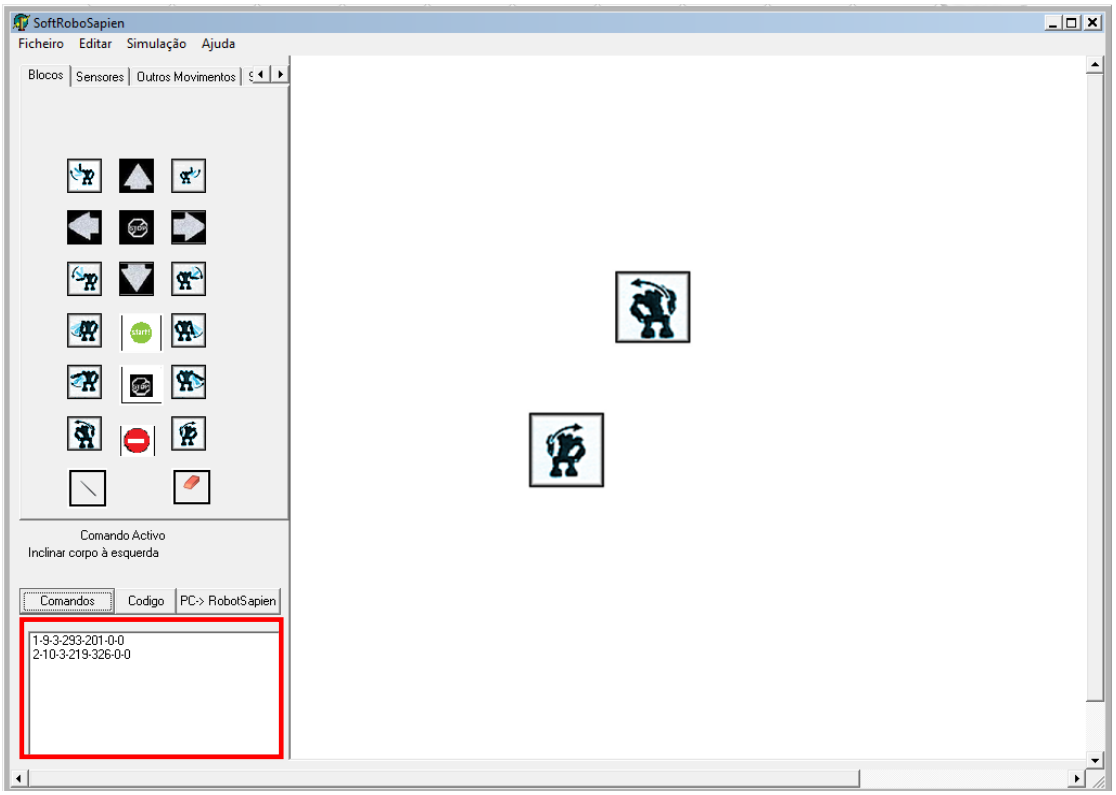


Figura 20- Apresentação das instruções

Todas as instruções podem ser cortadas, apagados ou copiadas. Para além disso, é possível editar as suas propriedades, conforme se ilustra na figura 21.

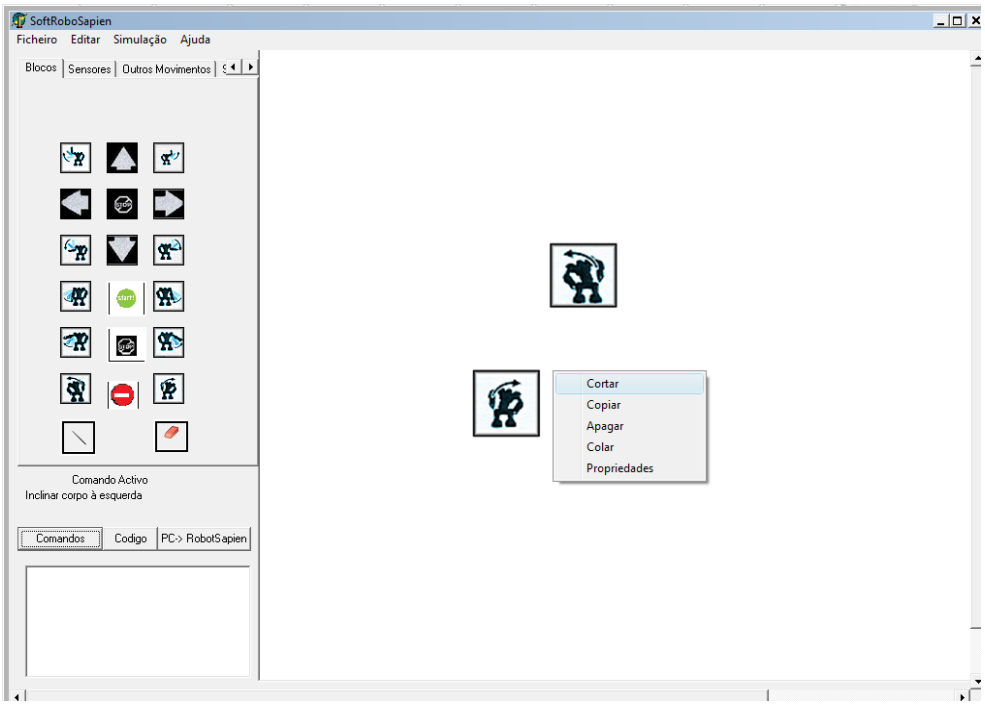


Figura 21- Menu que permite editar a instrução: cortar, copiar, apagar, colar e propriedades

As propriedades das instruções podem ser visualizadas na figura 22.


Figura 22- Menu Propriedades

Neste menu, o programador pode alterar o tipo de instrução e a sua duração temporal.

### 3.5.2 Ordem das instruções do Robosapien

A próxima etapa do programa consiste em definir a sequência das instruções. Este processo é realizado pelo clique com o rato no comando (ver figura 23).

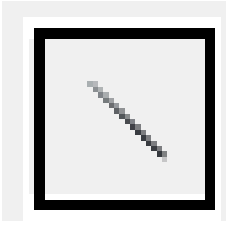

Figura 23- Função que permite interligar as instruções

O programa solicita a primeira instrução e posteriormente a segunda conforme demonstrado na figura 24 e na figura 25.

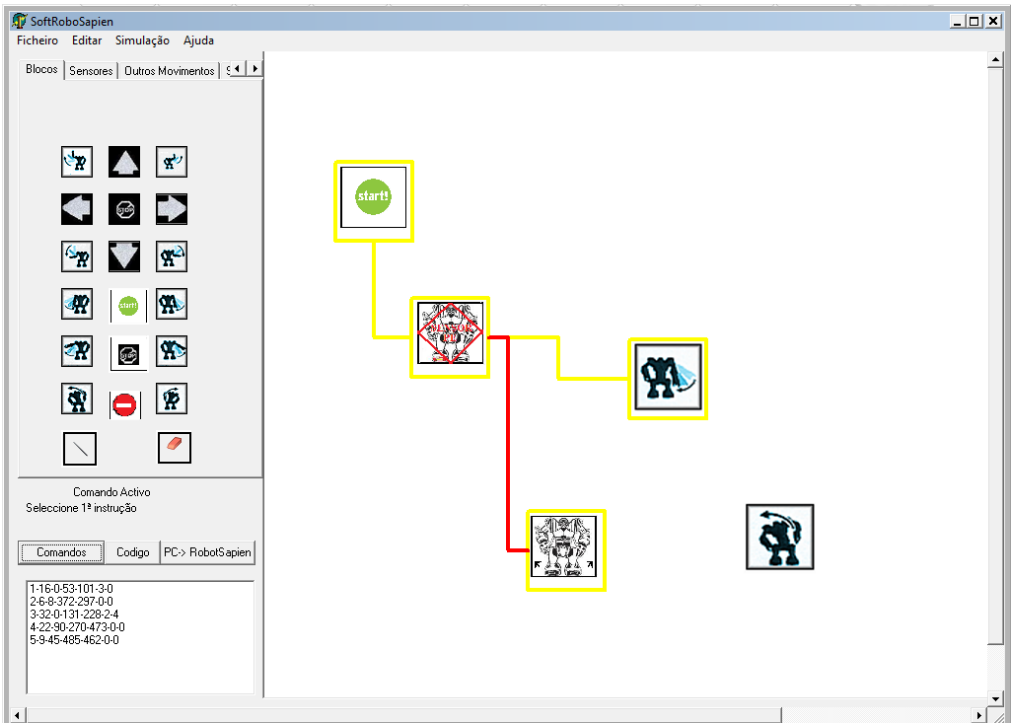


Figura 24- O programa solicita a 1ª Instrução Primeira instrução

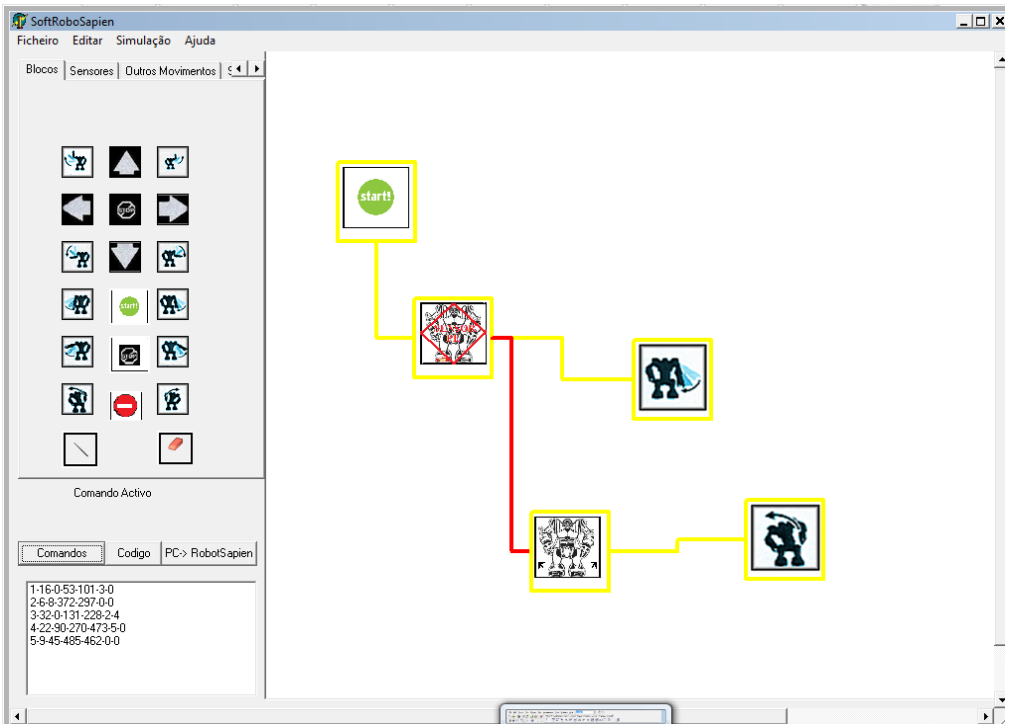


Figura 25- O programa solicita a 2ª instrução.

Caso a primeira instrução diga respeito ao estado de um sensor (condição), no ecrã surge o menu mostrado na figura 26.

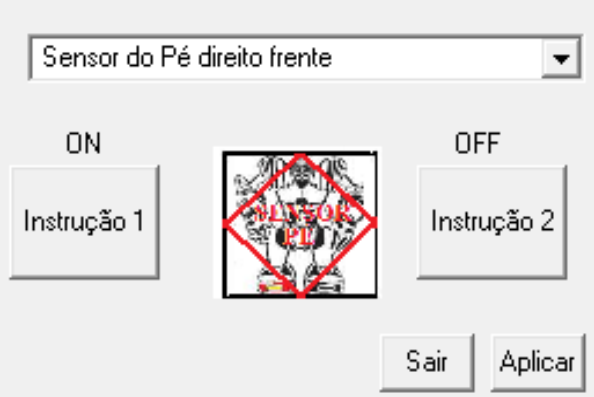


Figura 26 - Menu de interligação de instruções

O programador tem que seleccionar a instrução a executar caso o sensor esteja no estado *ON* e a instrução a executar caso o sensor esteja no estado *OFF*. Neste menu, o programador poderá alterar o tipo do sensor.

### 3.5.3 Transferência das instruções

A próxima etapa consiste na visualização das instruções em binário e a sua transferência para o robot.

## 3.6 Descrição dos vários menus

Quando se inicia a execução da aplicação SoftRobsapien, automaticamente é disponibilizada uma nova folha de programação, ver figura 27. Em alternativa, o utilizador também pode abrir uma nova folha de programação indo ao menu *ficheiro* e seleccionando o comando *Novo*.

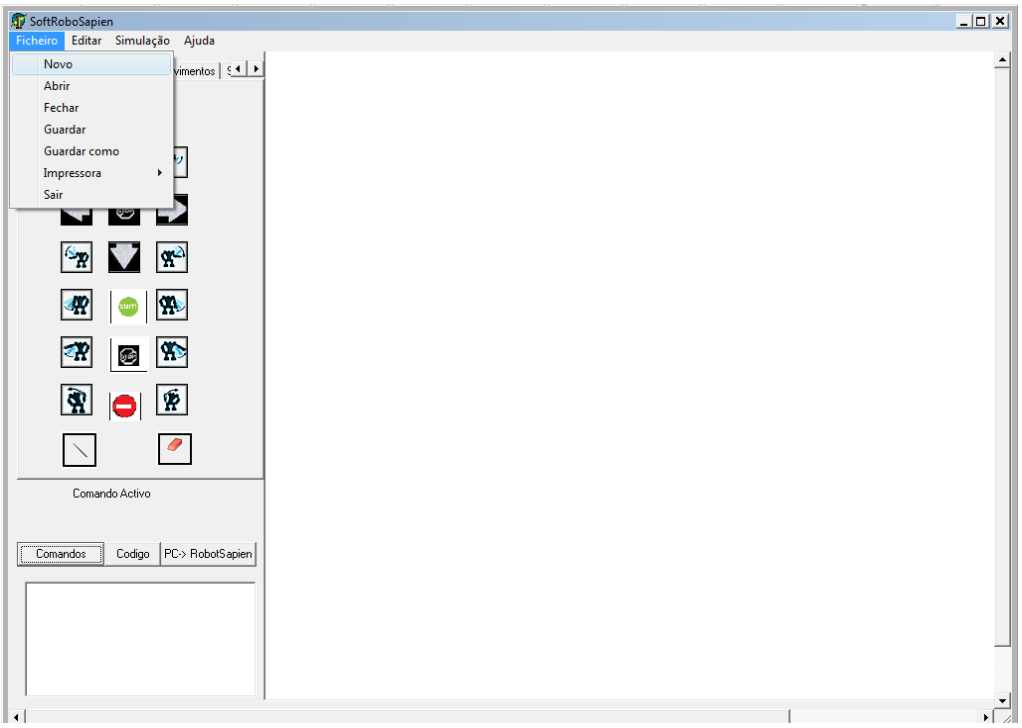


**Figura 27- Menu ficheiro Novo**

Para abrir um ficheiro que já existe no disco, ou noutra unidade de armazenamento, pode-se escolher a opção *Ficheiro Abrir* localizada no menu *ficheiro*.

O comando *fechar* permite fechar a folha de programação. Devendo o utilizador ter o cuidado de primeiro guardar o programa no disco do computador. O utilizador ao guardar o trabalho desenvolvido pode voltar a abrir o documento a qualquer momento. A diferença entre o comando *guardar* e o *guardar como*, é que o comando *guardar como* permite gravar o ficheiro com outro nome diferente do primeiro, caso ainda não exista no disco um ficheiro com o mesmo nome. O comando *guardar* também permite gravar com um nome de acordo com a preferência do utilizador.

Tal como se ilustra na figura 28 é possível carregar um programa previamente elaborado e que esteja salvaguardo em memória.

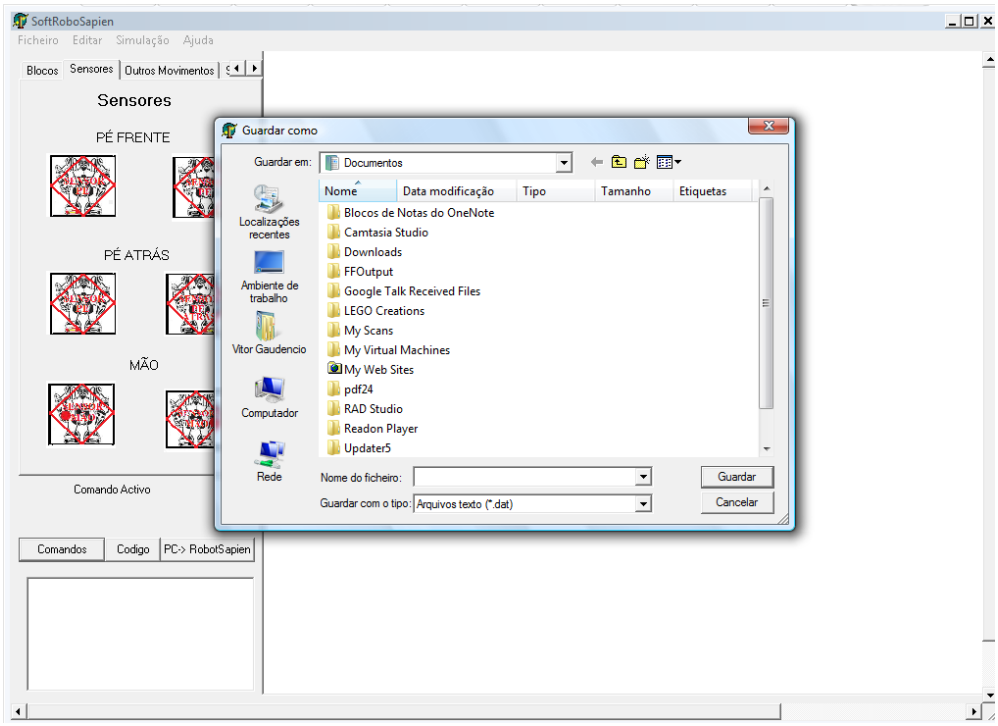


**Figura 28- Menu guardar ficheiro**

Para imprimir o programa basta seleccionar o menu *ficheiro* e posteriormente dar ordem de impressão.

O comando *sair* permite fechar a aplicação SoftRobosapien.

O Menu Editar está vocacionado para auxiliar as tarefas de edição do programa: copiar, colar e apagar, tal como se ilustra na figura 30.

Para isso, é necessário seleccionar o comando pretendido e posteriormente seleccionar a instrução que se pretende editar, ver figura 29.

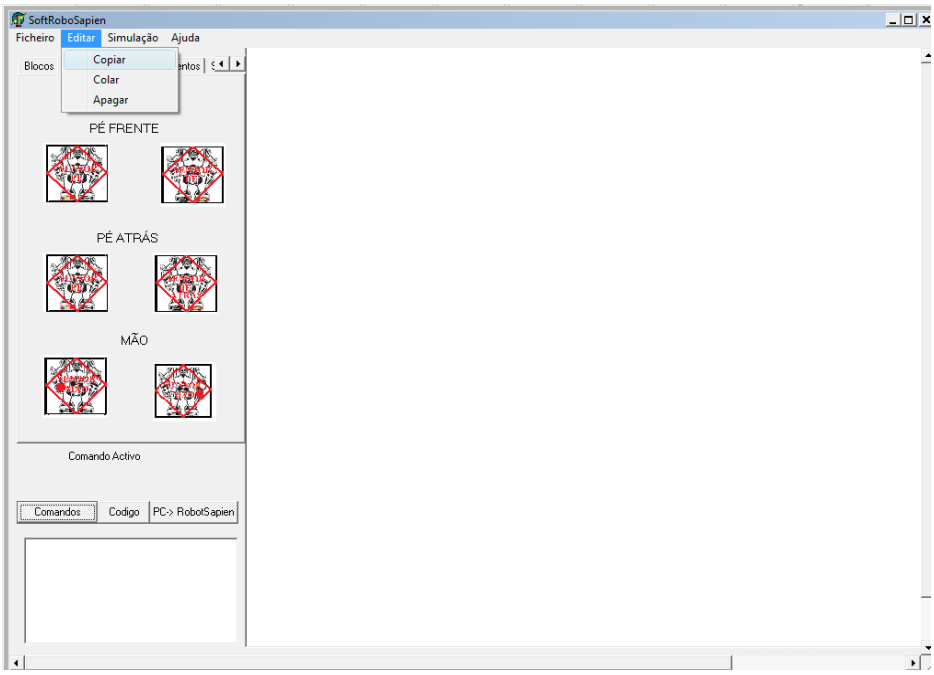


**Figura 29- Menu permite copiar, colar e apagar instrução.**

Os comandos deste menu são:

Cortar - Função que permite cortar instruções do programa após a selecção.

Copiar – Função que permite copiar instruções após seleccionadas.

Colar – Função que permite colar instruções que já tenham sido copiadas.

O Menu *Simulação PC-Robosapien* permite ao utilizador executar a transferência do programa para o Robosapien (ver figura 30).

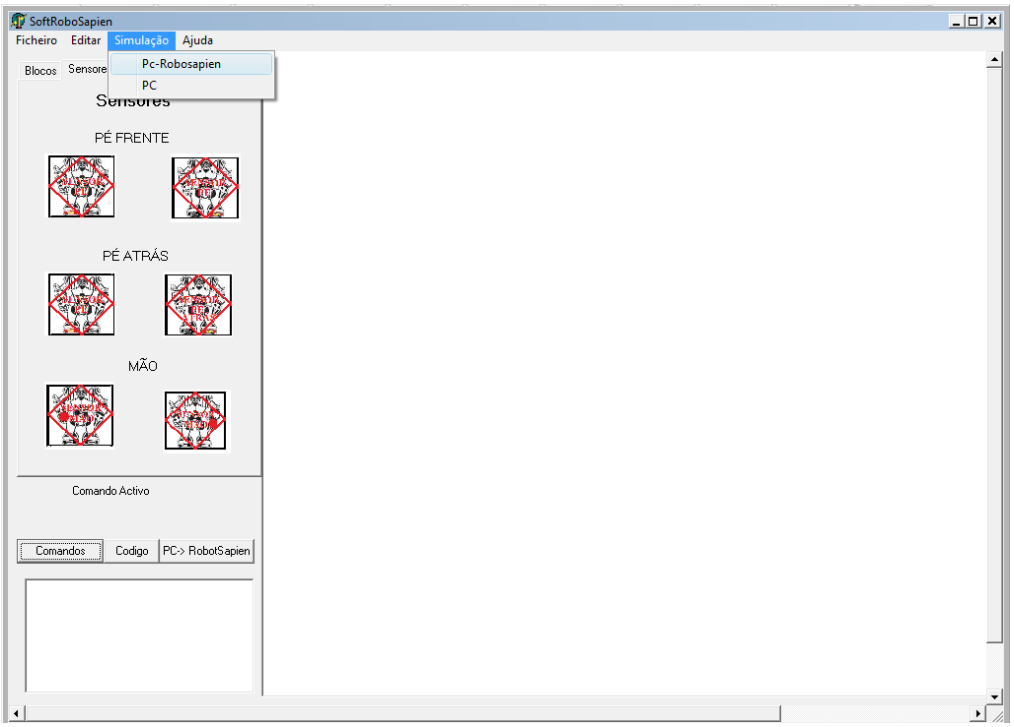


**Figura 30 - Menu de Simulação do programa para o Robot.**

Os comandos deste menu são:

PC - Robosapien – Função que permite enviar o programa para o Robosapien.

PC – Função que permite simular o programa no computador.

No Menu *Ajuda* o programador tem acesso a informação relativa ao funcionamento da aplicação, às suas características e limitações, ver a figura 31.

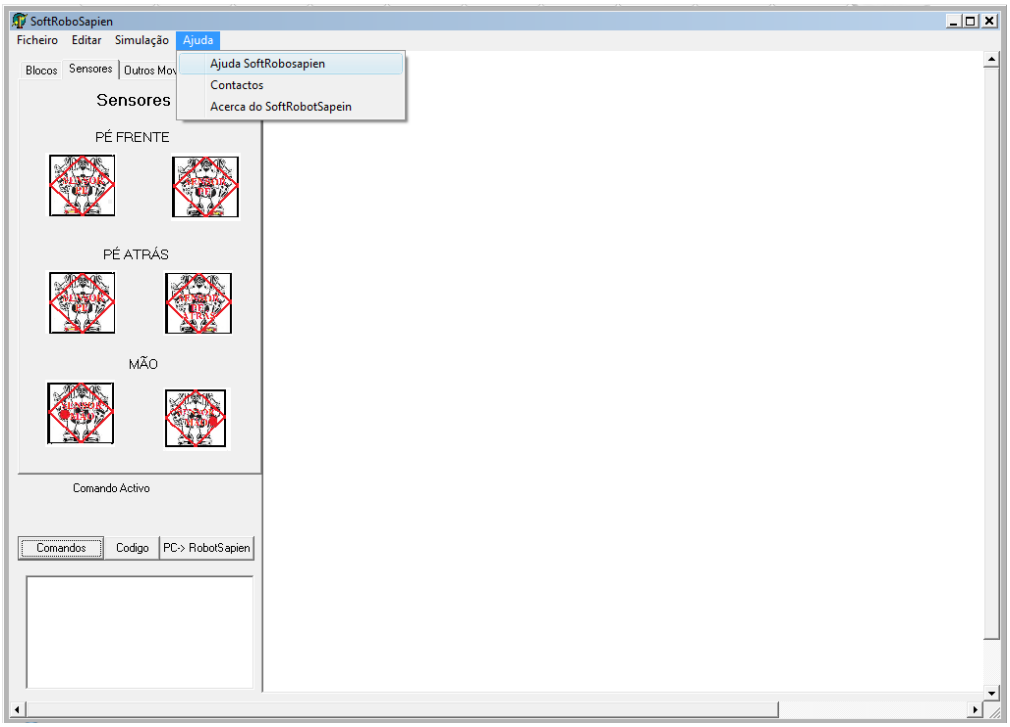


**Figura 31- Menu Ajuda**

Os comandos envolvidos neste menu são:

Ajuda SoftRobosapien – Função que permite obter ajuda sobre determinada instrução.

Acerca do SoftRobosapien – Informação da versão do software.

# Capítulo 4 - Robosapien na sala de aula

## 4.1 Fundamentação teórica

A utilização do Robosapien em sala de aula pode ser explorada de diferentes modos:

- Método expositivo;
- Aprendizagem baseada na resolução de problemas;
- Aprendizagem baseada em projecto.

### 4.1.1 Método expositivo

Os fundamentos teóricos do método expositivo provêm de três concepções contemporâneas: a estrutura e organização do conhecimento; a psicologia da aprendizagem verbal significativa e a psicologia cognitiva (Arends, 2008).

O conhecimento está organizado em diversas áreas temáticas e é designado por disciplinas. Estas constituem os recursos nas quais os docentes decidem os conteúdos que deverão ser ensinados e que obedecem a uma estrutura lógica.

Na psicologia da aprendizagem verbal significativa, David Ausubel (1963), apresenta a sua teoria tendo como ponto de partida a organização hierárquica dos conteúdos (Gonçalves, 2008). Segundo Ausubel, cada aluno detém uma organização de conhecimento já existente, relativamente a uma determinada área temática. Desta forma, o desenvolvimento de novos conteúdos só poderia surgir se houvesse ligação entre as estruturas cognitivas anteriores (Arends, 2008).

No que concerne à psicologia cognitiva da aprendizagem, ela permite aos docentes compreender o modo de funcionamento da mente, como o conhecimento é adquirido e representado e como os indivíduos percebem, aprendem, recordam e pensam sobre determinada informação. O conhecimento pode ser dividido em quatro: Factual, conhecer os elementos básicos de algo; Conceptual, conhecer as relações entre os elementos básicos; Processual, saber fazer algo, e Metacognitivo, saber quando utilizar determinados conhecimentos e a consciência que um indivíduo tem da sua própria cognição (Arends, 2008). O novo conhecimento entra na mente dos alunos através de um dos sentidos, audição ou visão, sendo detectado pela memória a curto prazo. Posteriormente esse conhecimento é armazenado na memória de longo prazo, pronta a ser recuperada quando necessário (Arends, 2008). O método expositivo é importante para os docentes por quatro motivos: primeiro, permite conhecer a estrutura e a organização do conhecimento em torno de proposições básicas e de ideias unificadoras; segundo, as aprendizagens de novos conhecimentos dependem do seu conhecimento prévio e das estruturas cognitivas existentes; terceiro, permite reflectir sobre quais as tarefas do docente que auxiliam as aprendizagens e, por último, reconhecer que as estruturas cognitivas sofrem alterações devido aos novos conhecimentos, sendo uma nova base para o desenvolvimento de novas estruturas cognitivas (Arends, 2008).

O método expositivo é centrado no professor, pode ser aplicado em todas as salas de aula, nomeadamente em sala de aula do ensino básico, secundário, superior e grupos de grande audiência (seminá-

rios, conferências e palestras), mas requer uma estrutura firme do ambiente de aprendizagem (Arends, 2008), condições de apresentação e de audição, visibilidade e que inclua instrumentos necessários para o aproveitamento de recursos de mutimédia.

### 4.1.2 Aprendizagem baseada em problemas

Os princípios da aprendizagem baseada em problemas podem ser encontrados nas teorias de Dewey, Bruner, Auzube, Rogers (Ribeiro, 2008), estando o seu suporte teórico na psicologia cognitiva, isto na forma como os indivíduos pensam durante uma determinada tarefa.

Na teoria de John Dewey, a aprendizagem baseada em problemas refere que a sala de aula deveria ser como um laboratório de pesquisa e de resolução de problemas da vida real e que esta aprendizagem obteria melhores resultados com pequenos grupos, a qual constitui a base filosófica da aprendizagem baseada em Problemas (Arends, 2008). O filósofo salienta que em situações que intencionalmente geram dúvidas ou perturbações intelectuais, o método valoriza as experiências com forte motivação prática e estímulo cognitivo, visão apoioada pela teoria Piaget (Cyrino & Pereira, 2004). A base teórica da aprendizagem assente em problemas advém das teorias de construtivismo (Piaget e Lev Vygotsky) (Arends, 2008), alteração do conhecimento evolui e muda à medida que os discentes se confrontam com novas experiencias, obrigando-os a construir ou a modificar os conhecimentos anteriores. A diferença entre Piaget e Vygotsky reside no facto de que o primeiro se centrou nos estádios de desenvolvimento intelectual pelos quais os indivíduos passam, independentemente das relações sociais e culturais, enquanto o segundo refere que a aprendizagem ocorre também através da interacção social, estimulando a construção de novas ideias e contribuindo para o desenvolvimento intelectual. Jerome Bruner (1962) realçou a importância para a aprendizagem pela descoberta como forma de ajudar os alunos a construir o seu próprio conhecimento. Noutra teoria de Bruner, o apoio de andaimes conceptuais, é um processo no qual o aprendente é auxiliado pelo docente a resolver um problema superior à sua capacidade intelectual. Tal como Lev Vygotsky, Jerome Bruner refere que a interacção social era responsável por grande parte da aquisição de linguagem e comportamentos para a resolução de problemas. Além disso, Richard Suchman (1962), desenvolveu uma abordagem na qual os docentes colocam os discentes perante situações confusas e os encorajam a pesquisar e a solucionar os problemas (Arends, 2008).

Na aprendizagem baseada em problemas, os discentes trabalham normalmente em pares ou em pequenos grupos, servindo também de motivação para o envolvimento apoiado em tarefas complexas.

### 4.1.3 Aprendizagem baseada em projecto

A aprendizagem baseada em projectos é um método de ensino-aprendizagem inovador em que o discente trabalha autonomamente a investigação na resolução de um ou mais problemas da vida real, quer económico, político, científico ou social. Este método de aprendizagem permite ao aluno construir o seu conhecimento. A aprendizagem baseada em projecto é uma mais-valia para o ensino dado que permite tanto ao docente como ao discente alargar os seus conhecimentos sobre um determinado assunto. Esta situação não é observada nos métodos de ensino tradicionais. A aprendizagem Baseada em Projectos é

um método de aprendizagem mais próximo do real.

O aluno adquire competências de escrita criativa, capacidade de reflexão e de síntese, responsabilidade, gestão de tempo, aprender a aprender, gosto pela descoberta, cooperação, auto-gestão, capacidade de investigação e argumentação e a oportunidade de participação na resolução de problemas da vida real (Aprendizagem Baseada em Projectos).

## 4.2 Organização de uma aula modelo

No decorrer desta secção é apresentada a estrutura de uma aula modelo onde o Robosapien é explorado do ponto de vista pedagógico.

### 4.2.1 Conteúdos/Aprendizagens a adquirir pelos alunos:

- história do Robosapien;
- principais instruções;
- como funciona o SofRosapien;
- demonstração do funcionamento do programa SoftRobosapien;
- colocação de problema aos alunos.

### 4.2.2 Estratégias/Actividades a desenvolver

No início da aula, o docente faz um breve resumo da história do Robosapien e dos seus constituintes. Posteriormente, o docente explica aos alunos, através do método expositivo, como funciona o programa SoftRobosapien, quais os principais comandos e sensores existentes no robot. Seguidamente, o professor expõe um labirinto, para o qual os alunos têm que elaborar um algoritmo. O programa deve permitir ao robot ultrapassar os obstáculos. Este exercício será resolvido em pares.

No final da aula, o docente vai proceder à auto-avaliação, entregando uma ficha de auto-avaliação individual, onde cada aluno irá reflectir sobre as suas atitudes e aprendizagens no decorrer da formação.

### 4.2.3 Diferenciação pedagógica

É comum estarmos perante uma turma heterogénea e como tal, nessa situação, o docente deverá apoiar os alunos com mais dificuldades na realização das tarefas, de forma individual e colectiva sempre que necessário.

#### 4.2.4 Recursos a utilizar

- computadores portáteis;
- Robosapien;
- aplicação informática SotRobosapien;
- projector de vídeo.

#### 4.2.5 Avaliação

Os alunos serão avaliados relativamente ao interesse, empenho, participação, comportamento, criatividade e quanto às tarefas realizadas.

#### 4.2.6 Sumário da aula

História do Robosapien. Principais comandos. Demonstração do funcionamento do programa. Colocação de um problema onde os alunos têm que contornar um obstáculo com o robot.

# Capítulo 5 - Síntese

O trabalho descrito ao longo deste documento apresenta uma ferramenta com um levado potencial didáctico. Foi ilustrado como a introdução de dispositivos como o robotsapien pode contribuir para motivar o aluno, ao mesmo tempo que auxilia o professor na tarefa de ensinar.

Tornou-se também claro que a inclusão deste tipo de actividades, que requerem conhecimentos de programação, é muitas vezes dificultada pelos escassos conhecimentos dos alunos no domínio da informática. Nomeadamente, no que concerne à robótica, a elaboração de um trabalho funcional requer conhecimentos avançados de programação, muitas vezes recorrendo a linguagens de programação como C/C++ ou mesmo Assembly.

A identificação desta dificuldade esteve na origem deste trabalho. Procurou-se desenvolver uma plataforma que permitisse realizar actividades de programação a um nível gráfico evitando-se, assim, a necessidade de possuir conhecimentos avançados de programação.

Podemos constatar que uma ferramenta como a que é descrita ao longo deste trabalho pode servir como plataforma de iniciação à programação noutras linguagens, já que nela o estudante tem a oportunidade de explorar conceitos associados à programação.

Os resultados obtidos permitem concluir que a estrutura da aplicação desenvolvida pode ser aplicada noutras situações para além do domínio educacional, nomeadamente, no controlo e automação de equipamentos industriais (linhas de montagem, robot, etc…), simulação de processos, sistemas CAD/CAM, não sendo a sua aplicação restrita ao Robosapien.

O desenvolvimento de novas instruções pode ser realizado de forma fácil, desde que se respeite a compatibilidade com as instruções existentes. Esta característica da ferramenta desenvolvida permite que possa ser utilizada no controlo de outros robots criados pelos próprios alunos.

Como ficou demonstrado na parte final da tese, a aplicação desenvolvida poderá ser uma ferramenta de introdução à programação com um elevado potencial de sucesso. Este tipo de ferramenta é muito atractivo para os alunos. Deste modo, considera-se que o trabalho desenvolvido no decorrer desta dissertação constitui um excelente contributo para a promoção do ensino da robótica junto dos mais jovens.

Este projecto pode ter uma continuação em diferentes vertentes das quais se deixa uma lista de sugestões:

- avaliar o impacto em sala de aula da ferramenta desenvolvida;
- desenvolver robots de baixo custo que possam ser programados pela ferramenta apresentada;
- incluir funcionalidades de comunicação para que a programação seja feita por rádio frequência;
- desenvolver um simulador que permita ao utilizador avaliar o desempenho da aplicação que desenvolveu sem necessidade de um robot real.

# Bibliografia

*Aprendizagem Baseada em Projectos.* Buck Institut of Education.

Arends, R. I. (2008). *Aprender a Ensinar (7ª Edição). Tradução de A. Faria.* Lisboa: McGraw-Hill.

Caci, B., & D' Amico, A. (2002). *Children's Cognitive Abilities in Construction and Programming Robots.* Palermo: University Of Palermo.

Côrtes, P. L., & Shiraishi, K. (2004). *Conhecendo e Trabalhando com Delphi 8.* São Paulo: Editora Érica Ltda.

Cyrino, E. G., & Pereira, M. L. (2004). *Trabalhado com estratégias de ensino-aprendizado por descoberta na área da saúde: a problematização e a aprendizagem baseada em problemas.* Rio de Janeiro: Cadernos de Saúde Pública.

Gonçalves, S. (2008). *Pedagogia no Ensino Superior.* Coimbra: Escola Superior de Educação de Coimbra.

Grodzik, R. (2009). *Pic Cookbook for Virtual Instrumentation.* Susteren: Elektor International Media BV.

JC. (24 de 12 de 2004). *Ciberia.* Obtido em 18 de 04 de 2011, de Ciberia: http://ciberia.aeiou.pt/gen.pl?p=stories&op=view&fokey=id.stories/877

Kereki, I. F. (2008). *Scratch: Applications in Computer Science 1.* Uruguay: Uruguay University.

Knudsen, J. B. (1999). *The Unofficiall Guide to Lego Mindstorms Robots.* Sebastopol: O' Reilly.

Maloney, J., Burd, L., & Kafai, Y. (2004). *Scratch: A Sneak Preview.* Massachusetts: Mit Media Laboratory.

Marsh, T. (07 de 12 de 2010). The Evolution of a Roboticist. United States, United States. Obtido em 07 de 2011, de Robot.

Nagchaudhuri, A., Singh, G., Kaur, M., & George, S. (2002). *Lego Robotics products boosts student creativity in pre-college programs at Umes.* Boston.

Oliveira, E. P. *Actividades diversificadas em sala de aula: Construção do conhecimento na Escola Indígenea Tuxuaa Albino Moraes.* Universidade Federal de Roraima.

Papert, S. (1980). *Mindstorms Children, Computers and Powerfull Ideas.* New York: Basic Book.

Ribeiro, L. R. (2008). Aprendizagem Baseada em Problemas na Educação em Engenharia. Revista de Engenharia. *Revista de Engenharia* , 23-32.

*Robosapein - User's Manual.* (2004). Quebec: WowWee Ltd.

Rocha, R. (2006). *Utilização da Robótica Pedagogica no Processo de Ensino-Aprendizagem de*

*Programação de Computadores*. Belo Horizonte: Centro Federal e Educação Tecnológica de Minas Gerais - CEFET.

Rocha, R. (25 de 06 de 2011). Utilização da Robótica Pedagógica no processo de Ensino-Aprensizagem de programação de computadores. Belo Horizonye, Brasil.

You, Z.-J., Shen, C.-W., Liu, B.-J., & Chen, G.-D. (2006). *A Robot as a Teaching Assistant in an English Class*. Yuan Ze: Computer Science & Enginnering Yuan Ze University.

# Anexos

# Lista de Instruções

| N.º | Instrução | Tipo de instrução | Ícone | Nome do ficheiro | Variável | Código binário |
|---|---|---|---|---|---|---|
| 1 | Levantar Braço Direito | Movimento de Blocos |  | levantarbraçodireito | lbd | 000001 |
| 2 | Levantar Braço esquerdo | Movimento de Blocos |  | levantarbracoesquerdo | lbe | 000010 |
| 3 | Baixar Braço direito | Movimento de Blocos |  | baixarbracodireito | bbd | 000011 |
| 4 | Baixar Braço esquerdo | Movimento de Blocos |  | baixarbracoesquerdo | bbe | 000100 |
| 5 | Mover Braço Direito para dentro | Movimento de Blocos |  | bracodireitodentro | bdd | 000101 |
| 6 | Mover Braço esquerdo para dentro | Movimento de Blocos |  | bracoesquerdodentro | bed | 000110 |
| 7 | Mover Braço direito para fora | Movimento de Blocos |  | Abrirbracodireito | abd | 000111 |
| 8 | Mover Braço esquerdo para fora | Movimento de Blocos |  | Abrirbracoesquerdo | abe | 001000 |

| N.º | Instrução | Tipo de instrução | Ícone | Nome do ficheiro | Variável | Código binário |
|---|---|---|---|---|---|---|
| 9 | Inclinar corpo para o Lado direito | Movimento de Blocos |  | Inclinarcorpodireito | icd | 001001 |
| 10 | Inclinar corpo para o Lado esquerdo | Movimento de Blocos |  | Inclinarcorpoesquerdo | ice | 001010 |
| 11 | Andar para frente | Movimento de Blocos |  | marchafrente | mf | 001011 |
| 12 | Andar para trás | Movimento de Blocos |  | marchaatras | ma | 001100 |
| 13 | Virar para a direita | Movimento de Blocos |  | moverobotdireita | mrd | 001101 |
| 14 | Virar para a esquerda | Movimento de Blocos |  | moverobotesquerda | mre | 001110 |
| 15 | Parar | Movimento de Blocos |  | Stop | sp | 001111 |
| 16 | Start | Movimento de Blocos |  | start | sta | 010000 |
| 17 | Halt | Movimento de Blocos |  | halt | hal | 010001 |

| N.º | Instrução | Tipo de instrução | Ícone | Nome do ficheiro | Variável | Código binário |
|---|---|---|---|---|---|---|
| 18 | Pancada com a mão direita | Outros movimentos |  | pancadamaodireita | pmd | 010010 |
| 19 | Pancada com a mão esquerda | Outros movimentos |  | pancadamaoesquerda | pme | 010011 |
| 20 | Apanhar com a mão direita | Outros movimentos |  | apanharmaodireita | apmd | 010100 |
| 21 | Apanhar com a mão esquerda | Outros movimentos |  | apanharmaoesquerda | apme | 010101 |
| 22 | Inclinar para trás | Outros movimentos |  | inclinartras | int | 010110 |
| 23 | Inclinar para frente | Outros movimentos |  | inclinarfrente | inf | 010111 |
| 24 | Avançar com a mão direita | Outros movimentos |  | avancarmaodireita | amd | 011000 |
| 25 | Avançar com a mão esquerda | Outros movimentos |  | avancarmaoesquerda | ame | 011001 |
| 26 | Mover pé direito | Outros movimentos |  | moverpedireito | mpd | 011010 |
| 27 | Mover pé esquerdo | Outros movimentos |  | moverpesquerdo | mpe | 011011 |
| 28 | Murro 1 - Mão direita | Murro |  | murromaodireita | mmd | 011100 |

| N.º | Instrução | Tipo de instrução | Ícone | Nome do ficheiro | Variável | Código binário |
|---|---|---|---|---|---|---|
| 29 | Murro 1- Mão esquerda | Murro |  | murromaoesquerda | mme | 011101 |
| 30 | Murro 2 -Mão Direita | Murro |  | murroamaodireita | mamd | 011110 |
| 31 | Murro 2 -Mão esquerda | Murro |  | murroamaoesquerda | mbme | 011111 |
| 32 | Sensor pé direito frente | Sensores |  | sensorpedireitofrente | spdf | 100000 |
| 33 | Sensor pé esquerdo frente | Sensores |  | sensorpesquerdofrente | spef | 100001 |
| 34 | Sensor pé direito atrás | Sensores |  | sensorpedireitotras | spdt | 100010 |
| 35 | Sensor pé esquerdo atrás | Sensores |  | sensorpesquerdotras | spet | 100011 |
| 36 | Sensor mão direita | Sensores |  | sensormaodireita | smd | 100100 |
| 37 | Sensor mão esquerda | Sensores |  | sensormaoesquerda | sme | 100101 |
| 38 | Dormir | Sentimentos |  | dormir | dorm | 100110 |
| 39 | Escutar | Sentimentos |  | escutar | escu | 100111 |
| 40 | Acordar | Sentimentos |  | acordar | acor | 101000 |

| N.º | Instrução | Tipo de instrução | Ícone | Nome do ficheiro | Variável | Código binário |
|---|---|---|---|---|---|---|
| 41 | Bulldozer | Sentimentos |  | bulldozer | bull | 101001 |